JavaOne

Democratizing Development Metrics

9/30/14 - Christopher W.H. Davis

Agenda

Thesis: Development teams should be responsible for tracking themselves through metrics that are easy to obtain and communicate

What We'll Talk About

- Types of data you should be looking for
- How to get metrics to measure your team
- A proposed system for analytics
- Communicating up the chain

About The Presenter

Christopher W H Davis

- Hacking/Deving professionally since the 90's
- Leading teams at Nike in awesomeness
- @techchrisdavis
- Author: Measure, React Repeat

A Few Truths

Someone Wants You To Develop Faster

- Software development is NOT a slow industry

Agile Is The Rule, Not The Exception

- If you're not practicing agile, I'm surprised
- There is no standard agile practice

Everyone Wants Happy Developers

- DevOps, Continuous Delivery (CD), Agile all focus on the developer
- Developers get the job done and are expensive


OVER PRESSURE
PULSE FREE MECHANICS
CLIPLIGHT FIREFLY

SPEED
96

Water Speed
Presure (PSI)
Temp


Bottom
Middle
Top


AMPERES
DISCHARGE
CHARGE
OIL


MPH


MPH
TEMP
OIL

00:12
FLT TIME
01:16
TIME REM
RPM
15.0
18
48
15
AUX LTR
MAIN LTR
ONE
--:--
TIME REM
14.7
22
VOLTS
AMPS
AUX LTR
68
83

Agile Definitions Are Not Straightforward

Working software is the primary measure of progress

- What do you mean?

Any measurement you're currently using has to be cheap

- Is the value from the improvement associated with a metric higher then the cost of collecting it

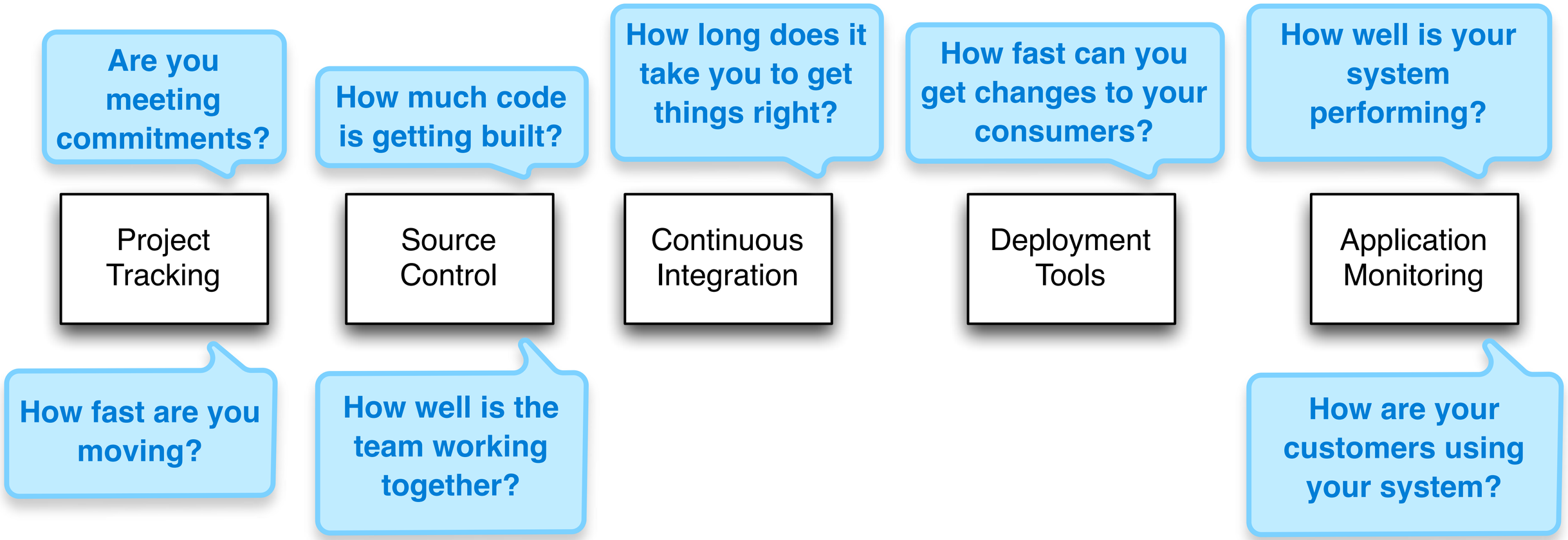
Only Measure What Matters

- So what matters?

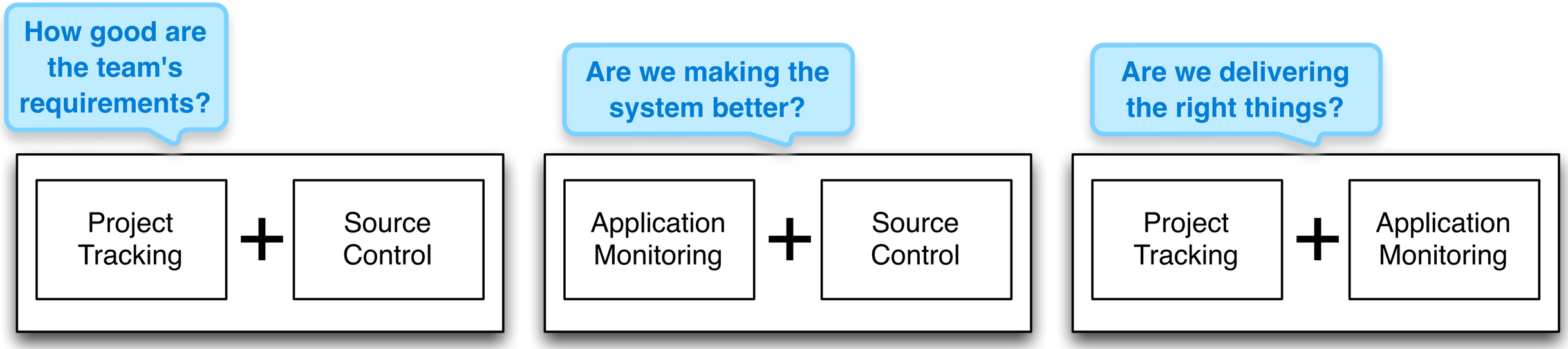
Data Is Everywhere…

Manage tasks & bugs	Manage code & collaboration	Generate builds, & run tests	Move code across environments	Ensure everything is working
Project Tracking	Source Control	Continuous Integration	Deployment Tools	Application Monitoring

It Answers Questions...



And Combined Can Give Lots Of Insight

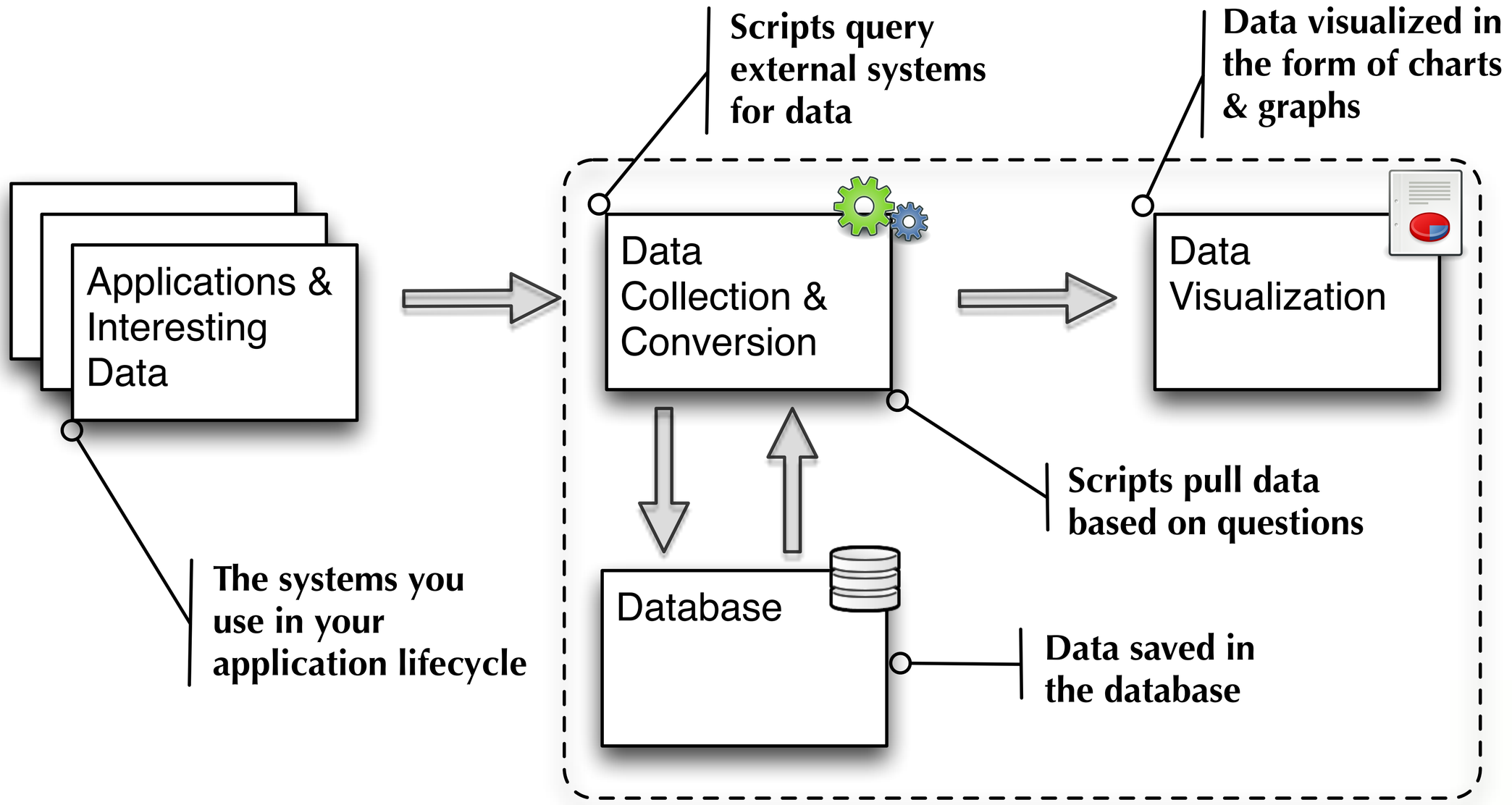


And It's Not That Hard

Components Of Data Analytics

- APIs to get data out of source systems
  - You probably work with APIs all the time
- Database to store data in
  - You probably know how to work with a database
- Application to normalize & transform data
  - Current frameworks make getting data REALLY EASY
  - Charting frameworks are common and really good

Architecture Of An Analytics System



Whoa, What?

Hang on…

- You're talking about a data analytics system
- I don't have time for that!

Actually You Do

Why Would You Not Measure Yourself?

- Development teams are closer to the data
  - You understand it, you have the context
  - Spotting trends is easier when you're living it
- Developing with today's tools makes building this stuff easy
- The only way to have a truly autonomous team is to measures & report on yourself

And If You Don’t, Who Will?

4 Key Data Sources You Want

If you don't have this stuff, get it!

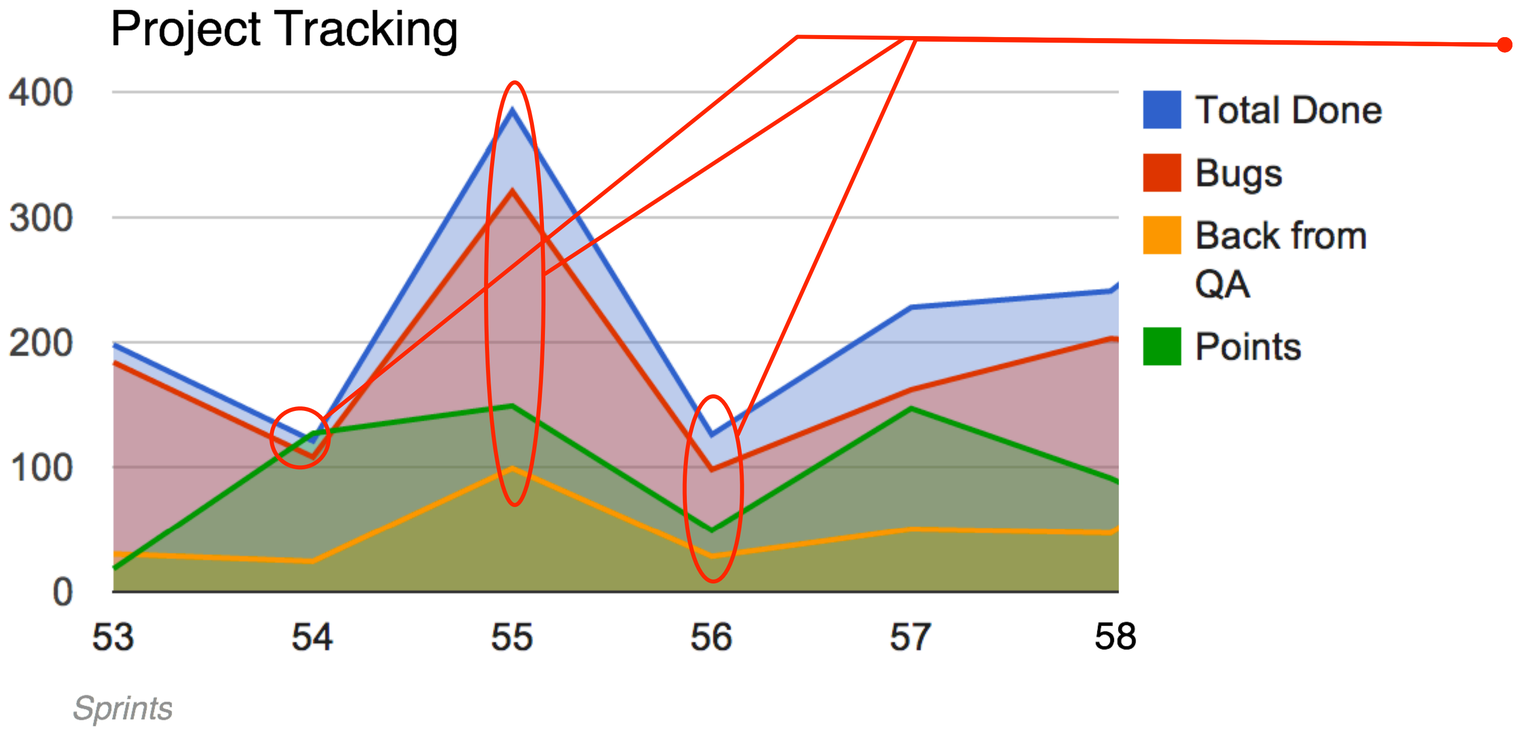
- Project Tracking System (PTS) Data
  - JIRA, Rally, Stuff like that

- Source Control (SCM) Data
  - Hopefully you're using Git

- Continuous Integration/Delivery CI/CD Data
  - Jenkins, TeamCity, Bamboo, Go-CD

- Production Monitoring Data
  - New Relic, Graphite

PTS Data Looks Bad Outside The Team

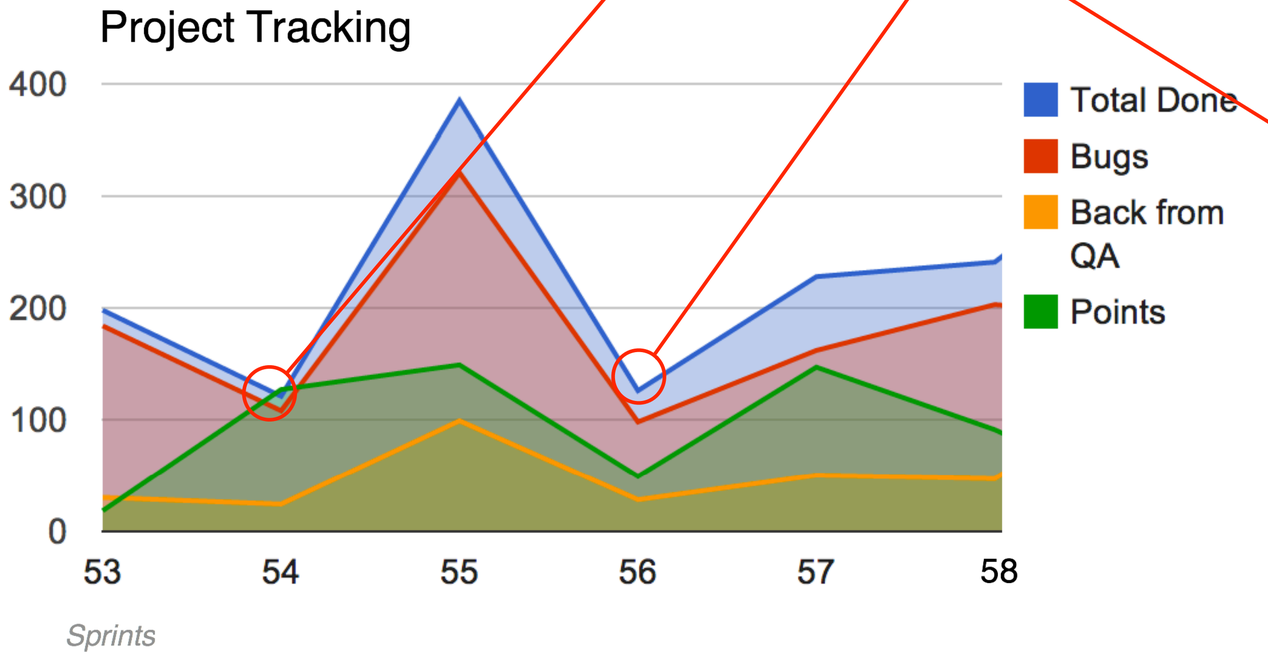


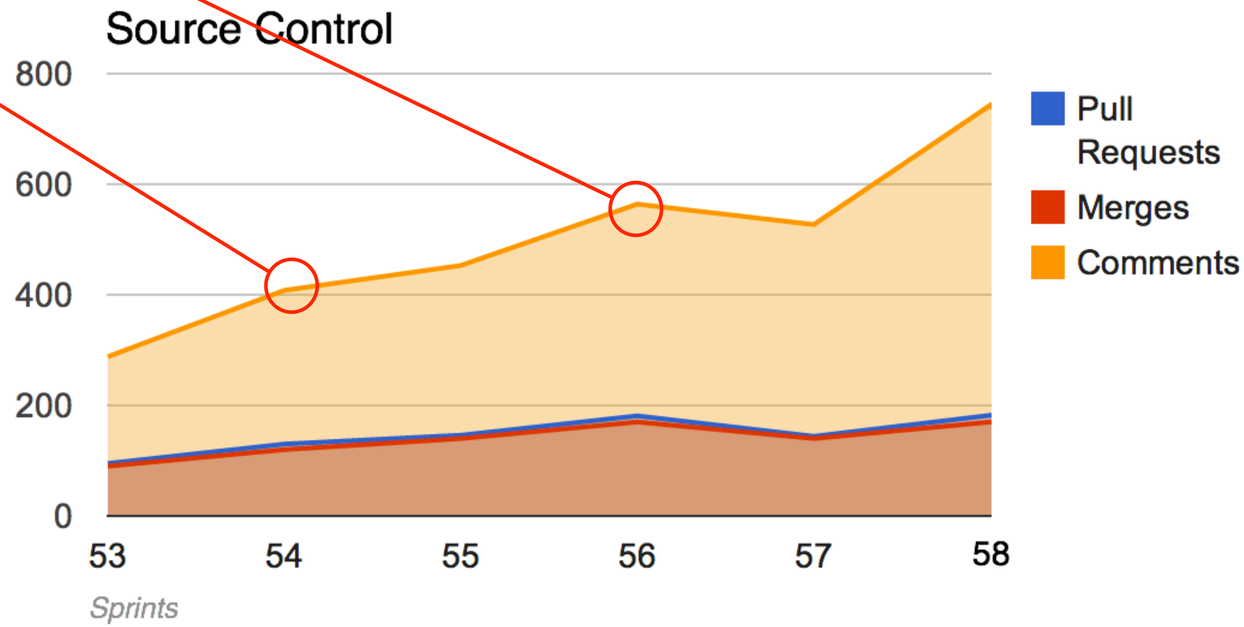
1) Sprint points don't match tasks complete

2) Most of the tasks complete are bugs

SCM + PTS: Closer Look

Source control use is trending up, yet there are large dips in completed





Narratives

Occured	Story
Tue Sep 17 00:00:00 PDT 2013	Release 4.3.1, iOS7 compatibility
Fri Sep 13 00:00:00 PDT 2013	Release 4.4 was changed from Coach to Photo Sharing

Let's Get To It

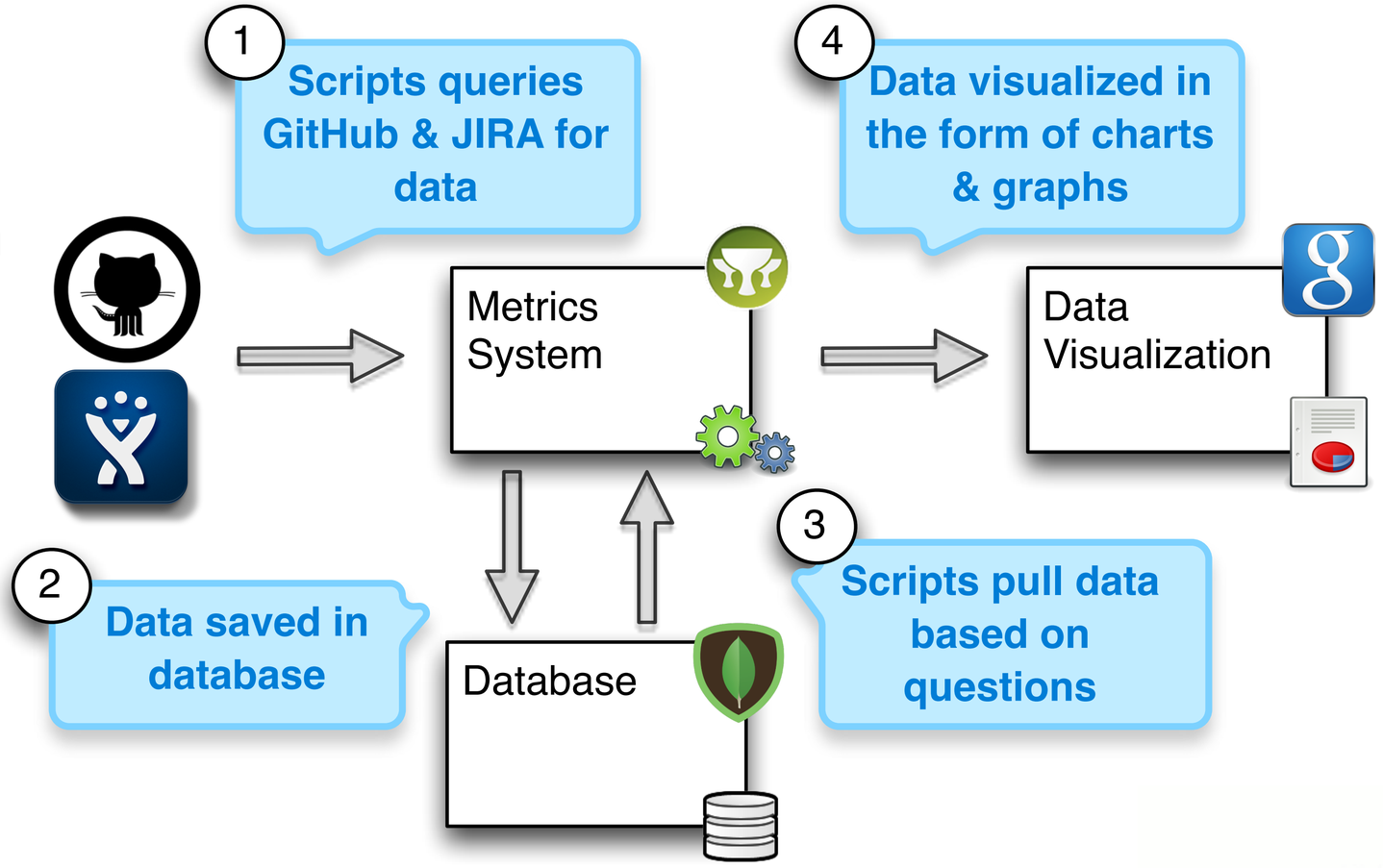
Here's what we're going to do

- Get data from an API
- Get it into a DB
- Get it into some nice charts

Technologies Used

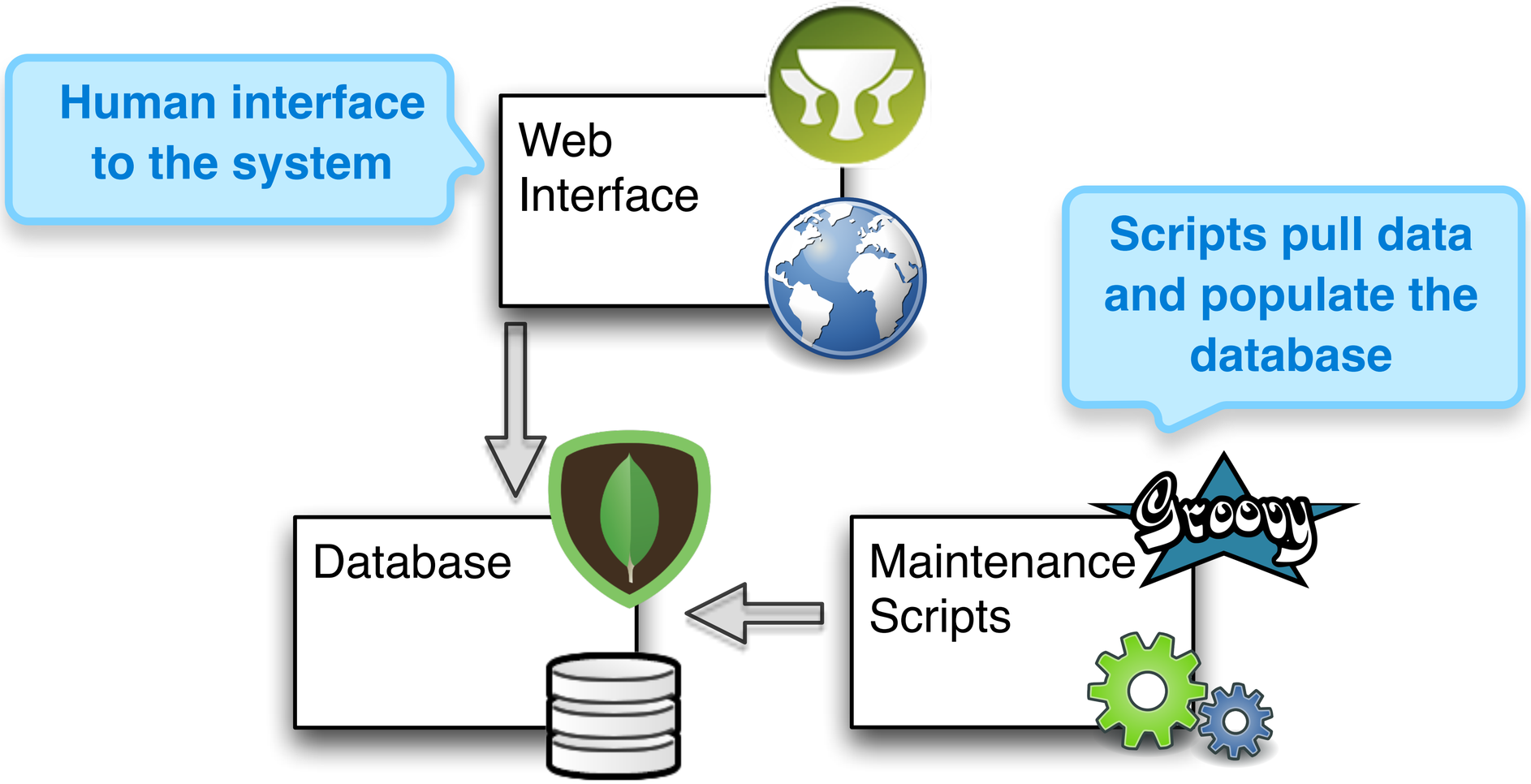
Here's what we're going to use to do it

- Grails - because it's easy
  - Lots of great plug-ins
  - Convention over configuration
- MongoDB - also easy
  - Plugs in to Grails
  - Has built in map-reduce
- Google Charts - yea, easy
  - Lots of different charts
  - Well supported



Technologies Used

Double Click On The System



Project Set Up

```
> grails create-app project-lifecycle-intelligence #A


> cd project-lifecycle-intelligence/ #B
> grails create-domain-class com.blastamo.sourceControlData #C
> grails create-domain-class com.blastamo.projectData #D
```

#A Create a new web application called project-lifecycle-intelligence
#B Once your project is created move into the new directory to update it
#C Create a domain object for source control data
#D Create a domain object for project data

Project Set Up – Add Plug Ins

```
plugins {
    // plugins for the build system only
    build ":tomcat:7.0.52.1"

    // plugins for the compile step
    compile ":scaffolding:2.0.3"
    compile ':cache:1.1.2'
    compile ':rest:0.8'
    compile ':mongodb:2.0.1'
    compile ":google-visualization:0.7"

    // plugins needed at runtime but not for compilation
    //runtime ":hibernate:3.6.10.13" // or ":hibernate4:4.3.5.1"
    //runtime ":database-migration:1.4.0"
    runtime ":jquery:1.11.0.2"
    runtime ":resources:1.2.7"
```

Project Set Up – Create Domain

```
class SourceControlData {
    int pullRequests
    int reviews
    int deniedPullRequests
    int reviewsDenied
    int sprintNumber
    //String id

    static mapWith = "mongo"

    static constraints = {
    }

    static mapping = {
        sort "sprintNumber"
    }

}
```

```
class ProjectData {
    int totalDone
    int bugsDoneCount
    int points
    int movedBackFromQA
    int sprintNumber
    String sprintDescription
    String id

    static mapWith = "mongo"

    static constraints = {
    }

    static mapping = {
        sort "sprintNumber"
    }
}
```

Maps this object to Mongo through MongoDB GORM

When we get data pre-sort it by sprintNumber

Scaffolding Is Awesome

```
> grails compile

> grails generate-all com.blastamo.ProjectData

> grails generate-all com.blastamo.SourceControlData
```

- Compile the app
- Generate scaffolding

HTTP Requests Are Easy

```
static def getApiData(url, path, query) {

    println url
    println path
    println query
    def http = new HTTPBuilder(url)

    http.request( GET, JSON ) {
        uri.path = path
        uri.query = query
        println uri

        headers.'User-Agent' = 'Mozilla/5.0 Ubuntu/8.10 Firefox/3.0.4'
        headers.'Accept' = 'application/json'
        headers.'Content-Type' = 'application/json'

        response.success = { resp, json ->
            return json
        }

        response.failure = { resp ->
            println "Unexpected error: ${resp.statusLine.statusCode} : ${resp.statusLine.reasonPhrase}"
        }
    }
}
```

Get Data From SCM

```
SourceControlData getData(startDate, endDate, sprintNumber) {
    def sourceControlData = new SourceControlData()

    def url = 'https://api.github.com'
    def path = '/repos/yourrepo'
    def query = [state: "closed"]

    def jsonR = APIRequest.getApiData(url, path, state)
    def commentCount = 0
    def mergedCount = 0

    for(def i : jsonR) {
        if(i["updated_at"] >= startDate && i["updated_at"] <= endDate) {
            def jsonRes = APIRequest.getApiData(url, i._links["comments"].href.replaceAll(url,""), null)
            commentCount += jsonRes.size()
            if(i["merged_at"] != null) {
                mergedCount++
            }
        }
    }

    sourceControlData.pullRequests = jsonR.size()
    sourceControlData.comments = commentCount
    sourceControlData.merges = mergedCount
    sourceControlData.sprintNumber = sprintNumber

    println new JsonBuilder(sourceControlData).toPrettyString()
}
```

Parse the response from GitHub

Update the domain object

Get Data From PTS

```
ProjectData getData(sprintNumber) {
    def projectData = new ProjectData()

    def url = 'https://jira.blastamo.com'
    def path = '/rest/api/2/search/'
    def doneQuery = [jql: "status changed TO (\"Deploy Ready\", Done) and Sprint = " + sprintNumber]

    def jsonR = APIRequest.getApiData(url, path, doneQuery)

    def bugCounter = 0
    def pointCounter = 0

    for(def i : jsonR.issues) {
        switch(i.fields.issuetype.name.toUpperCase()) {
            case("BUG"):
                bugCounter++
        }
        if(i.fields.customfield_10013) {
            pointCounter += i.fields.customfield_10013.toInteger()
        }
    }
    projectData.sprintNumber = sprintNumber
    projectData.bugsDoneCount = bugCounter
    projectData.totalDone = jsonR.total
    projectData.points = pointCounter

    def backFromQAQuery = [jql: "status changed TO (Dev, \"Dev Ready\", \"Needs Definition\") FROM (QA, \"QA Ready\", Done)
    def jsonRes = APIRequest.getApiData(url, path, backFromQAQuery)

    projectData.movedBackFromQA = jsonRes.total

    println new JsonBuilder(projectData).toPrettyString()
    return projectData
}
```

Parse the response
& set the object properties

Add It To The Page

```
<gvisualization:columnCoreChart
   elementId="source_control_data"
   title="Source Control Data”
   width="${485}" height="${250}"
   hAxis="${new Expando(title: ‘Source Control Data’, titleColor:'A0A0A0')}"
   columns="graphColumns " data="graphData" />
```

```
def index(Integer max) {

    def graphColumns = [
            ['string', 'Sprint Number'],
            ['number', 'Pull Requests'],
            ['number', 'Reviews'],
            ['number','Denied Pull Requests'],
            ['number','Reviews Denied']]
    def graphData = []

    SourceControlData.getAll().each() { s ->
        def newElement = [p.sprintNumber, p.pullRequests, p.reviews, p.deniedPullRequests, p.reviewsDenied]
        graphData.add(newElement)
    }

    params.max = Math.min(max ?: 10, 100)
    respond SourceControlData.list(params), model:[
            sourceControlDataInstanceCount: SourceControlData.count(),
            graphColumns:graphColumns,
            graphData:graphData]
}
```

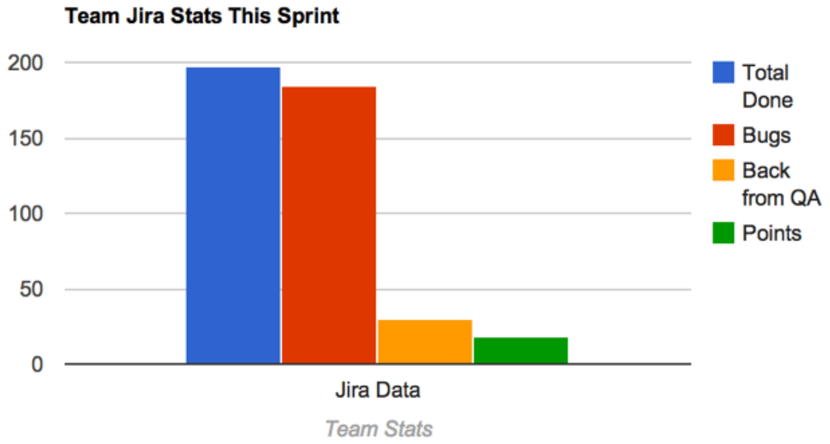
OMG It's Beautiful

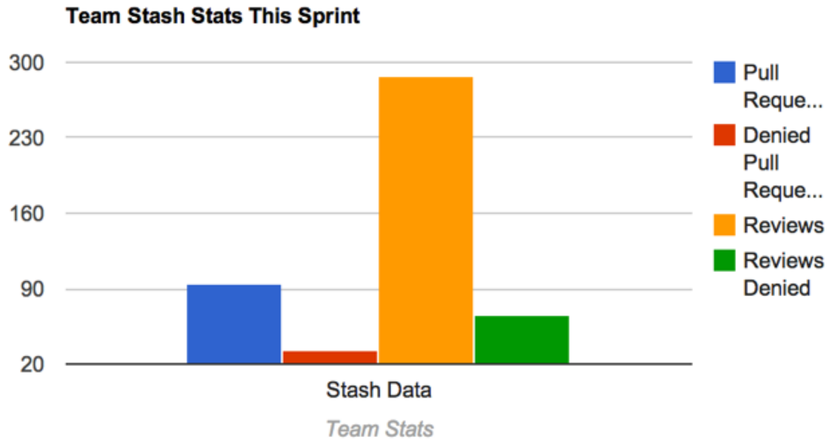
HOME | Admin

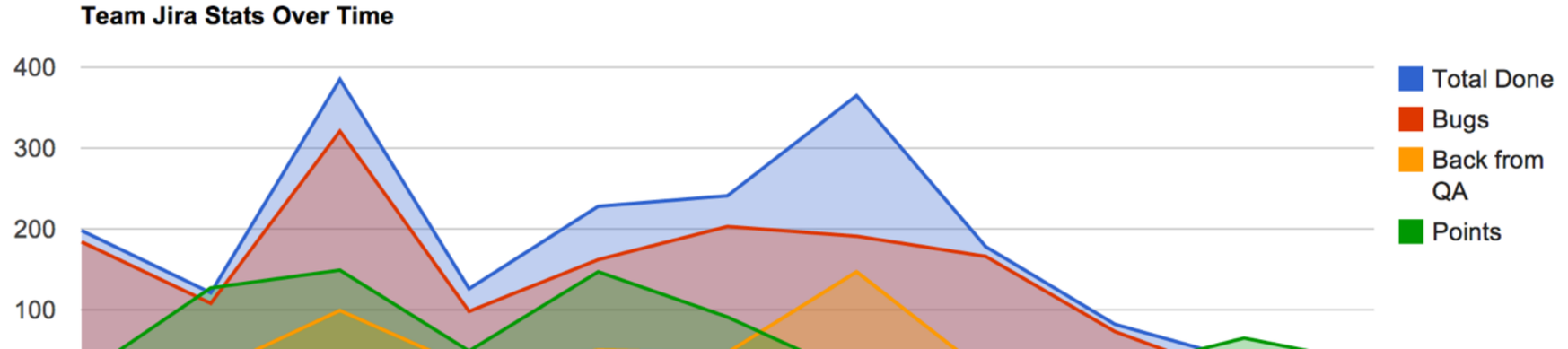
Stats for Sprint 53; aka Jira Sprint 431 for team Nike+ Running iOS

Sprint Goal: 18/42

Sonar | Automation Results





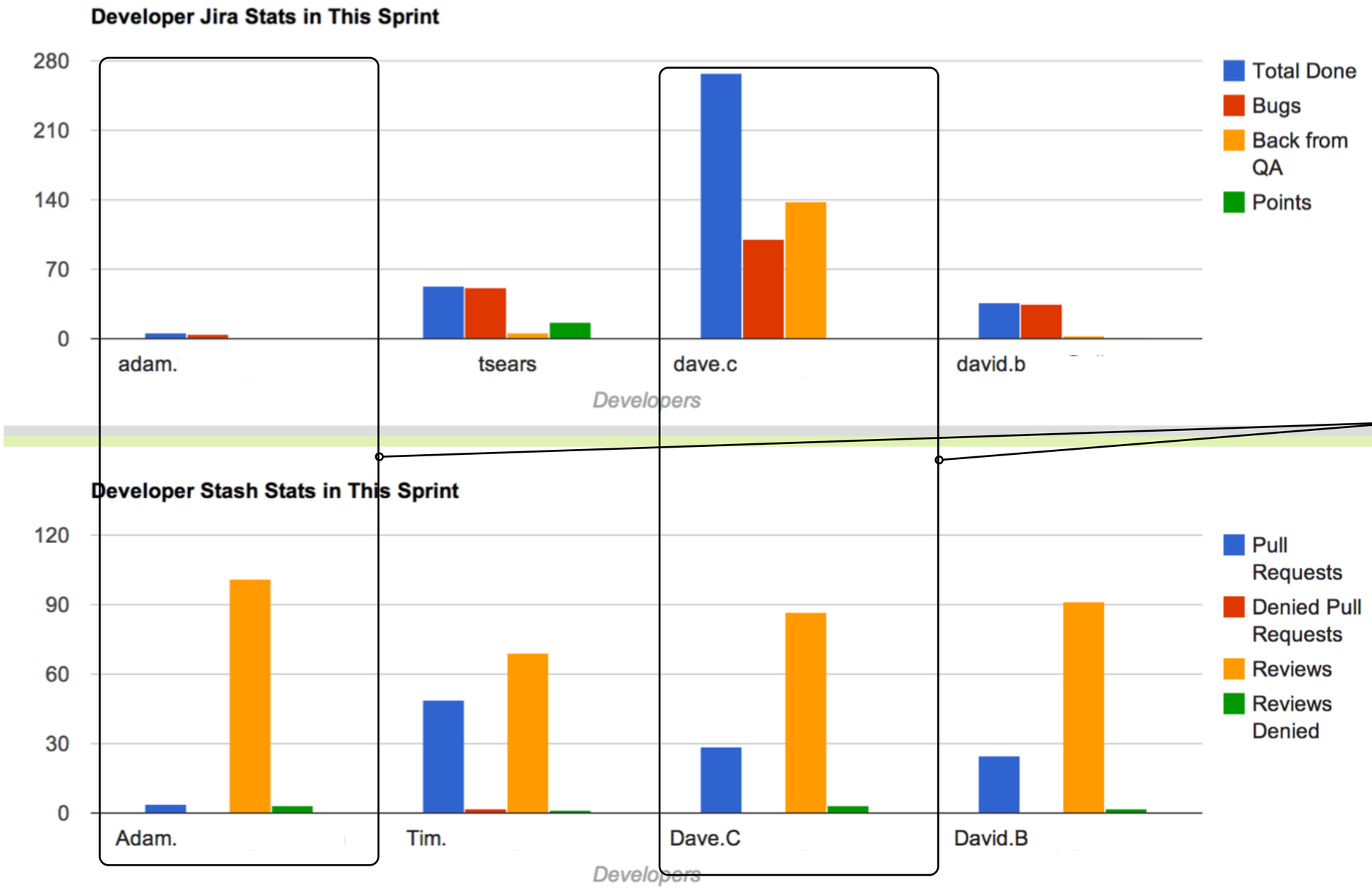


What Does It Mean?

Breaking Down Data

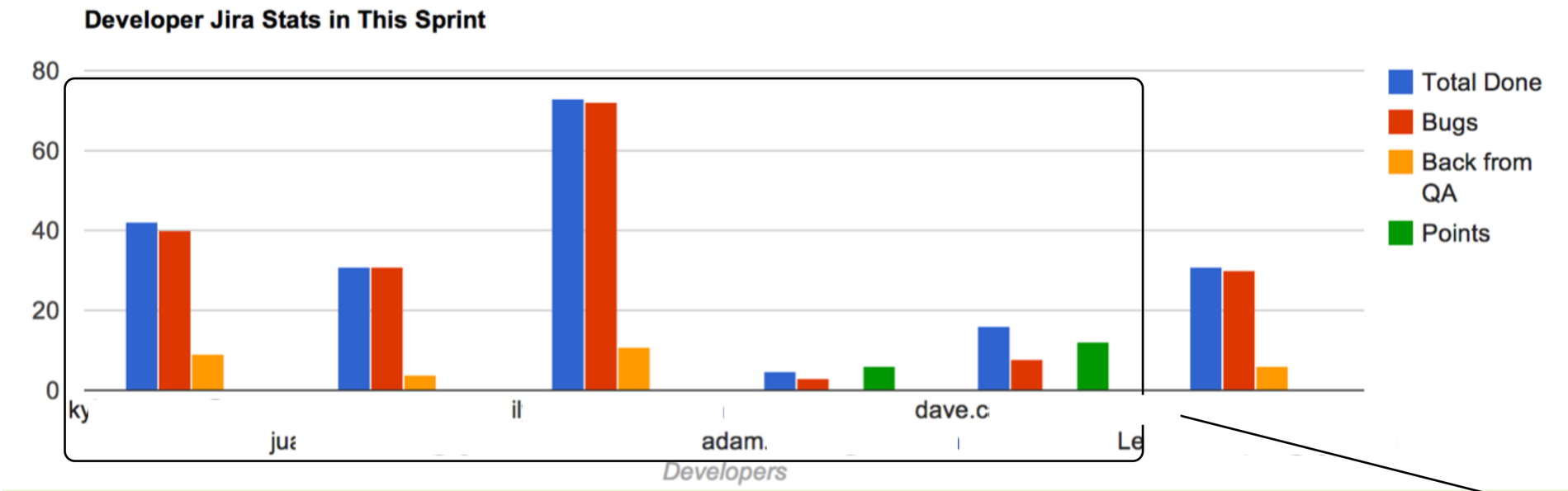
- Let's look at some examples
  - iOS Sprint 53 - Developer Productivity
  - Adding more developers to get more done

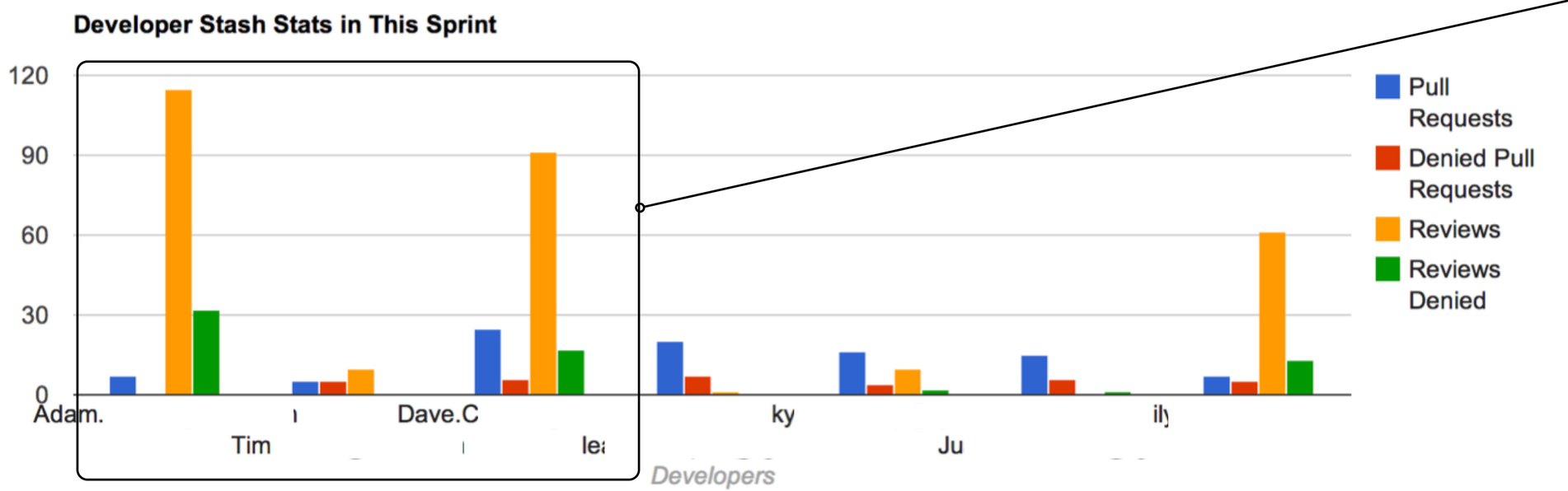
Developer Productivity: PTS + SCM



Combination of data paints the whole picture

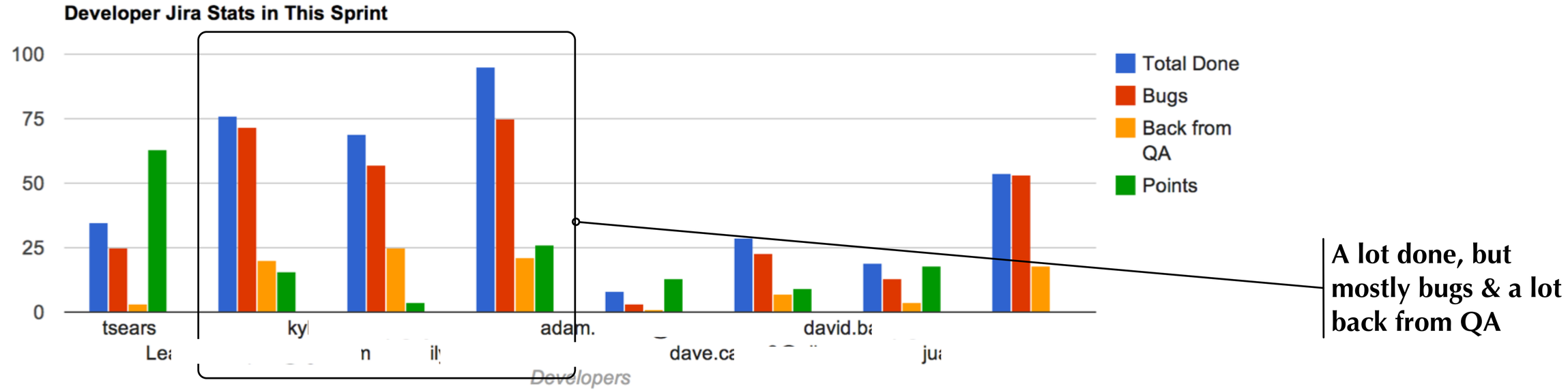
Developer Productivity “More Devs = Better”



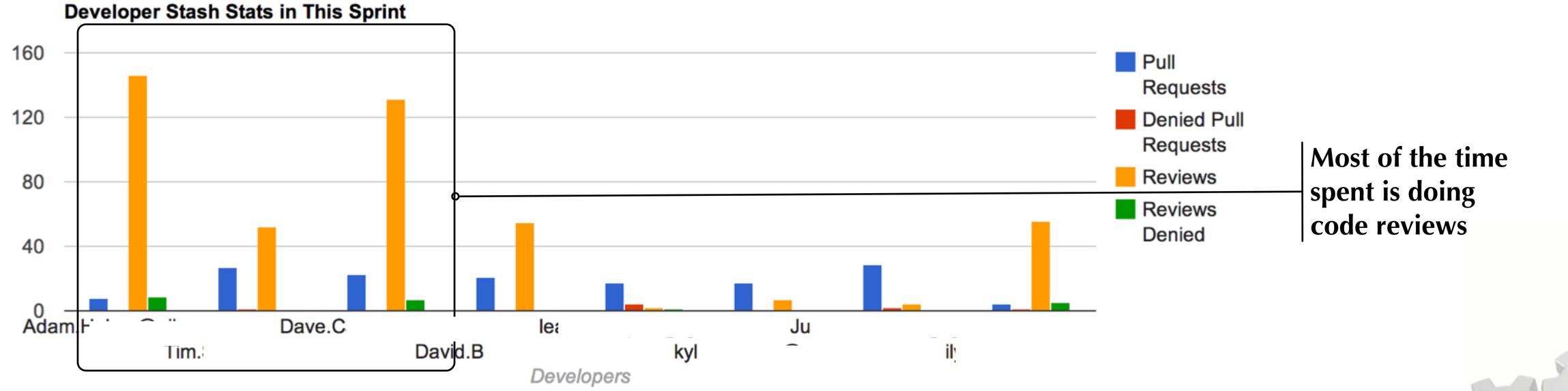


Productivity of new team hurts productivity of existing team

Developer Productivity “More Devs = Better 2”



A lot done, but mostly bugs & a lot back from QA



Most of the time spent is doing code reviews

Communication Advice

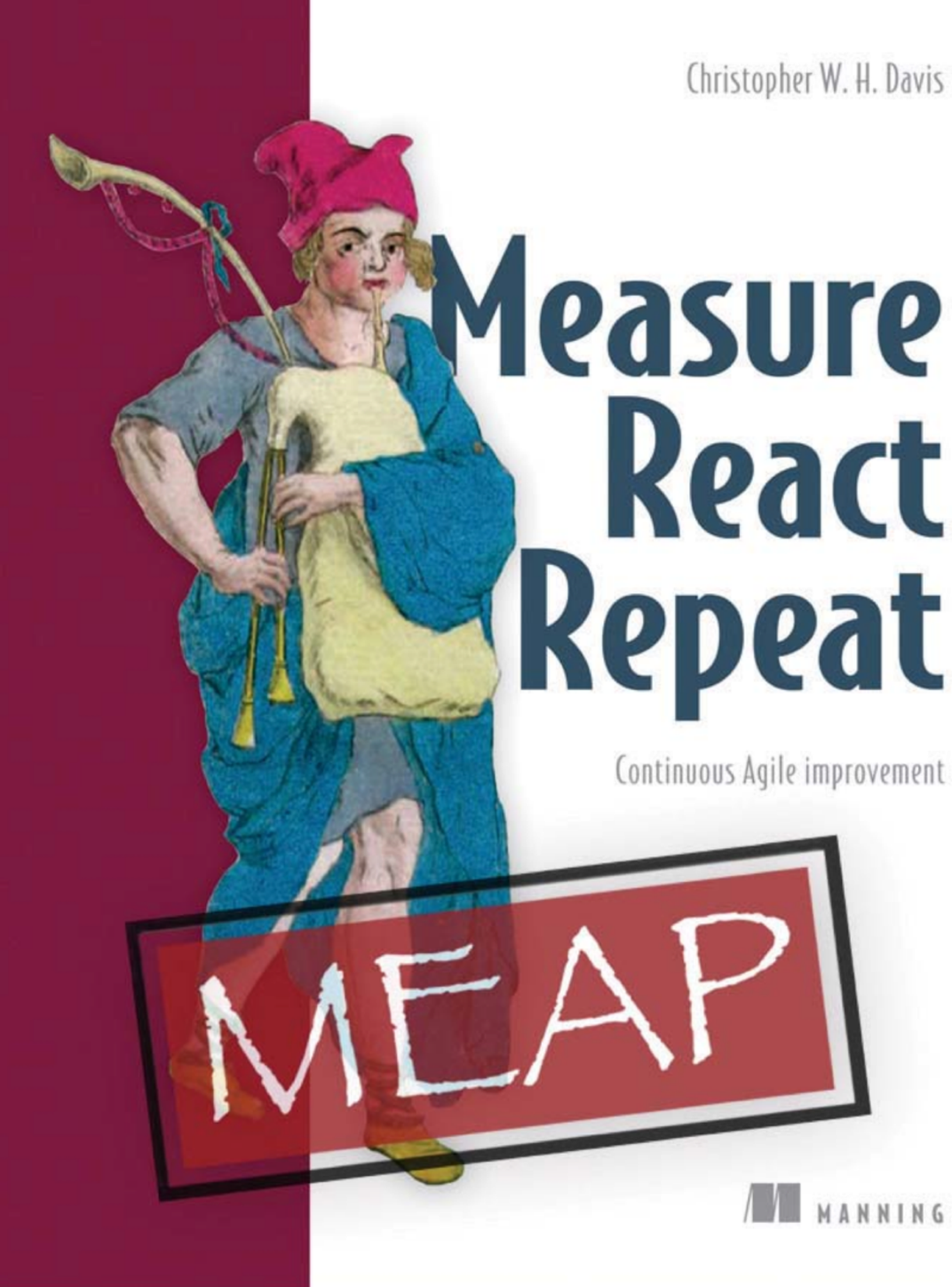
Build out the dashboards

- The higher the link in the chain, the simpler the data has to be
  - Aggregate to show trends
  - Have a point, don't just show a bunch of lines
- Allow for detail at the level of the team
- Ensure that data is actionable; use it in your planning sessions
- Meritocracies are OK

References

If you want to read more:

- Grails: https://grails.org/
- GVM : http://gvmtool.net/
- Google Charts: https://developers.google.com/chart/
- MongoDB: http://www.mongodb.org/
- Measure, React, Repeat MEAP: http://www.manning.com/davis2/



For more info, check out the book

javaonecftw (44% off all Manning books)