Agile, Secure Cloud Application Development & Management

Adam Getchell

acgetchell@ucdavis.edu

https://github.com/acgetchell

IT Security Symposium, June 15-16, 2011

University of California Davis

How to make Software

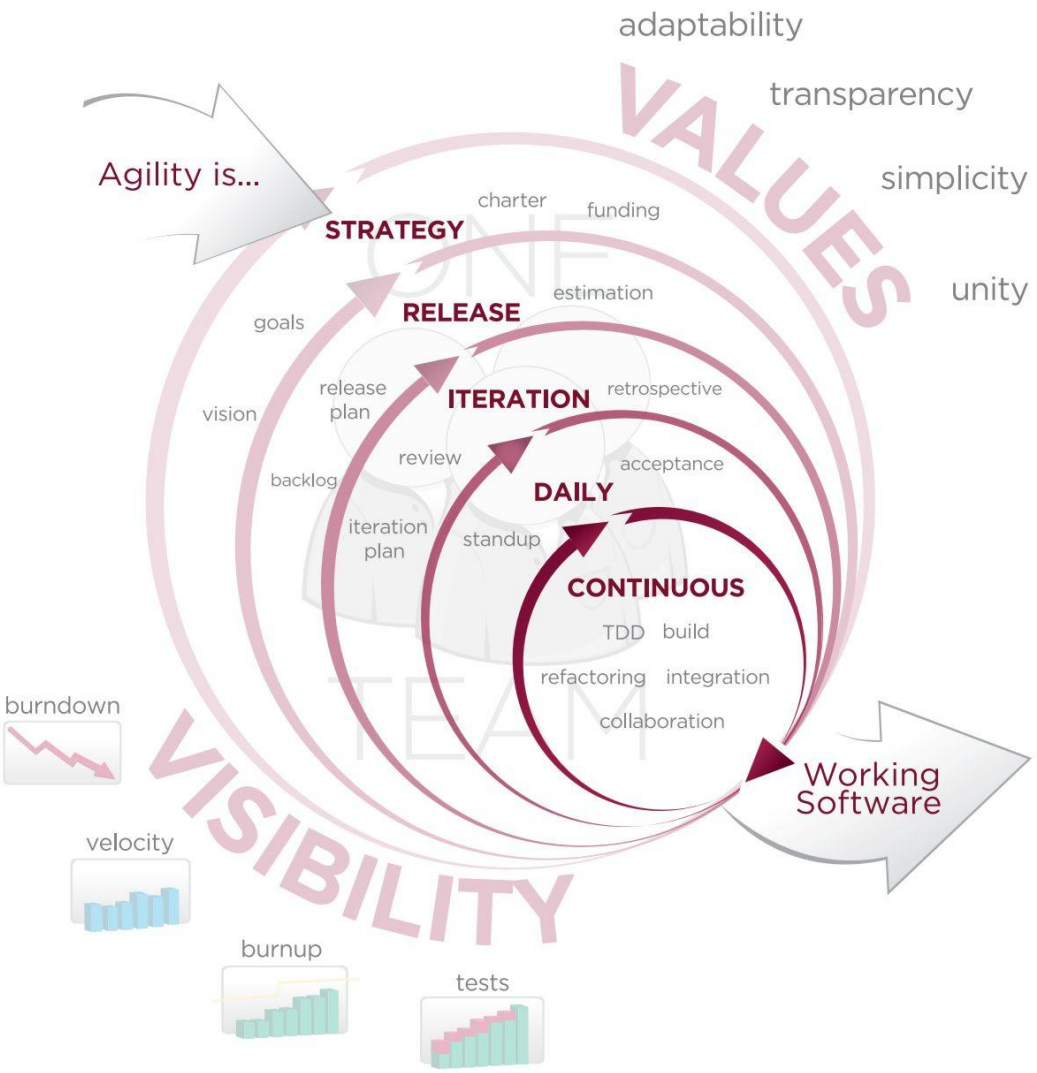
- Better*
- Faster
- Cheaper

* Since this is a security conference, one of our definitions of better = less bugs

Agile Development

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

AGILE DEVELOPMENT



ACCELERATE DELIVERY

Agile Development Principles

- Simplicity
- Self-organizing teams
- Regular adaption to changing circumstances
- Projects are built around motivated, trusted individuals
- Sustainable development, able to maintain a constant pace
- Welcome changing requirements, even late in development
- Continuous attention to technical excellence and good design
- Working software is the principal measure of progress
- Customer satisfaction by rapid delivery of useful software
- Close co-operation between business people and developers
- Face-to-face conversations as the best form of communication
- Working software is delivered frequently (weeks rather than months)

Agile Development Principles

- Simplicity
- Self-organizing teams
- Regular adaption to changing circumstances
- Projects are built around motivated, trusted individuals
- Sustainable development, able to maintain a constant pace
- Welcome changing requirements, even late in development
- Continuous attention to technical excellence and good design
- Working software is the principal measure of progress
- Customer satisfaction by rapid delivery of useful software
- Close co-operation between business people and developers
- Face-to-face conversations as the best form of communication
- Working software is delivered frequently (weeks rather than months)

Processes

Agile Development Principles

- Simplicity
- Self-organizing teams
- Regular adaption to changing circumstances
- Projects are built around motivated, trusted individuals
- Sustainable development, able to maintain a constant pace
- Welcome changing requirements, even late in development
- Continuous attention to technical excellence and good design
- Working software is the principal measure of progress
- Customer satisfaction by rapid delivery of useful software
- Close co-operation between business people and developers
- Face-to-face conversation is the best form of communication
- Working software is delivered frequently (weeks rather than months)

Processes

Feedback

Cloud Computing

Having secure access to all your applications and data from any network device

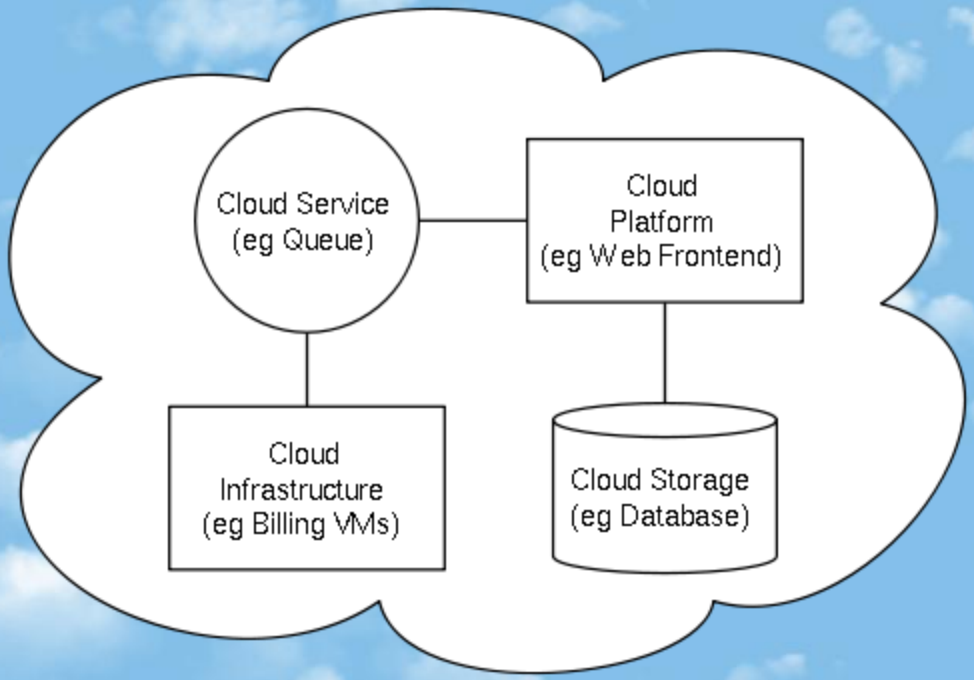
Agile Development in the Cloud

Agile and the Cloud: Perfect Together*

- Agile software: grant users of applications new capabilities as they need them
- Cloud computing: Pervasive, on-demand computation as a utility (elastic provision, pay-as-you-go, online, illusion of infinite supply)

*Jim Ensell, “Agile Development in the Cloud”, http://www.agilejournal.com/articles/columns/column-articles/6018

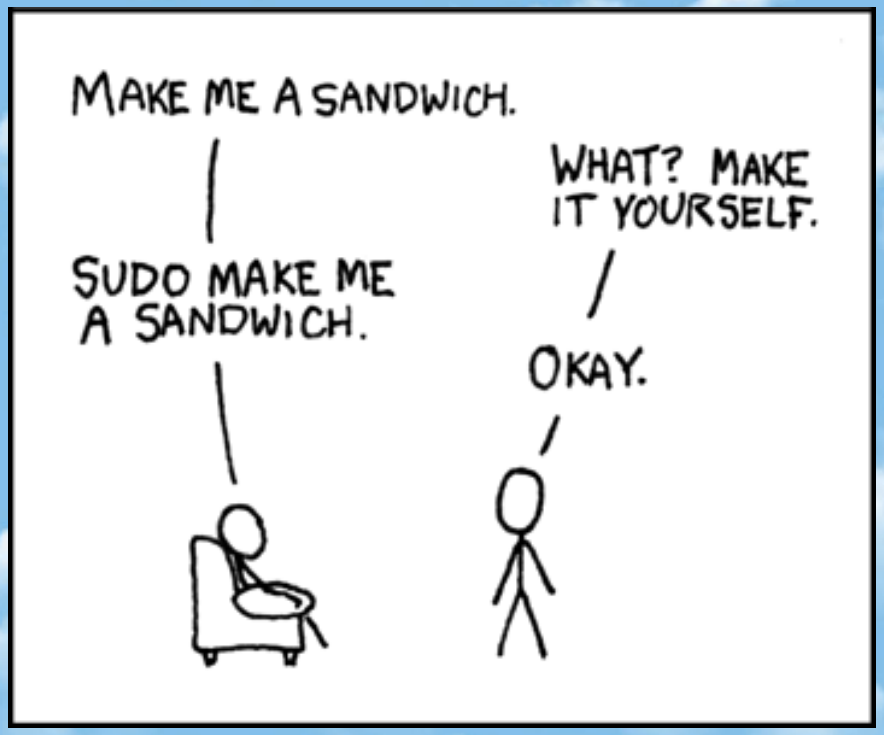
Cloud Applications



Stages of Agile Cloud Development

1. Developing using cloud-based source control management (SCM) repositories
2. Build in the cloud using virtual images, provide feedback to continuous integration (CI) servers
3. Testing in the cloud by building test machines
4. Production in the cloud, automated deployment, load-balancing, analytics

Programmer (noun): An organism that can turn caffeine and alchohol into code*



http://xkcd.com/149/

http://xkcd.com/303/

* http://uncyclopedia.wikia.com/wiki/Programmer

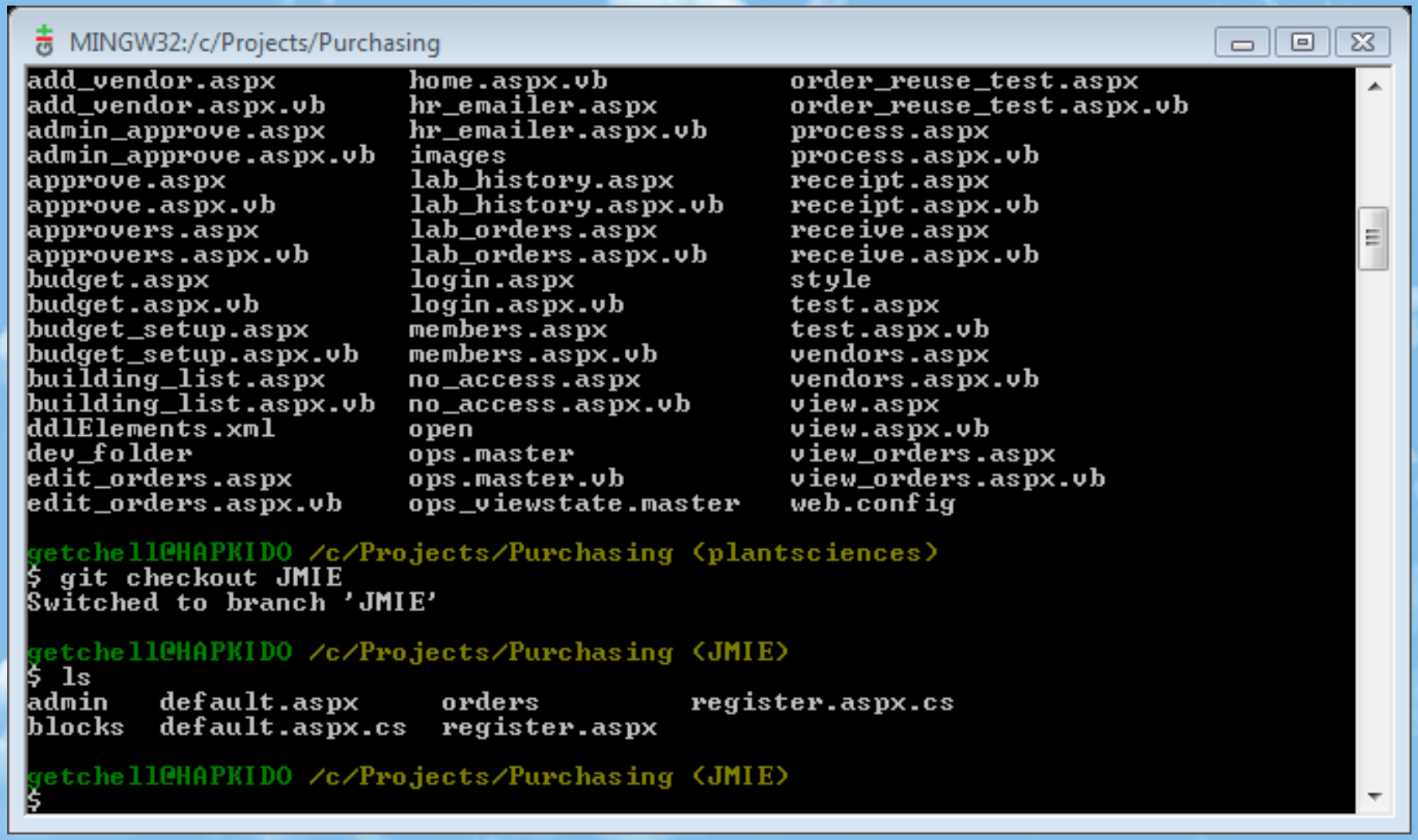
How to save caffeine by-products?

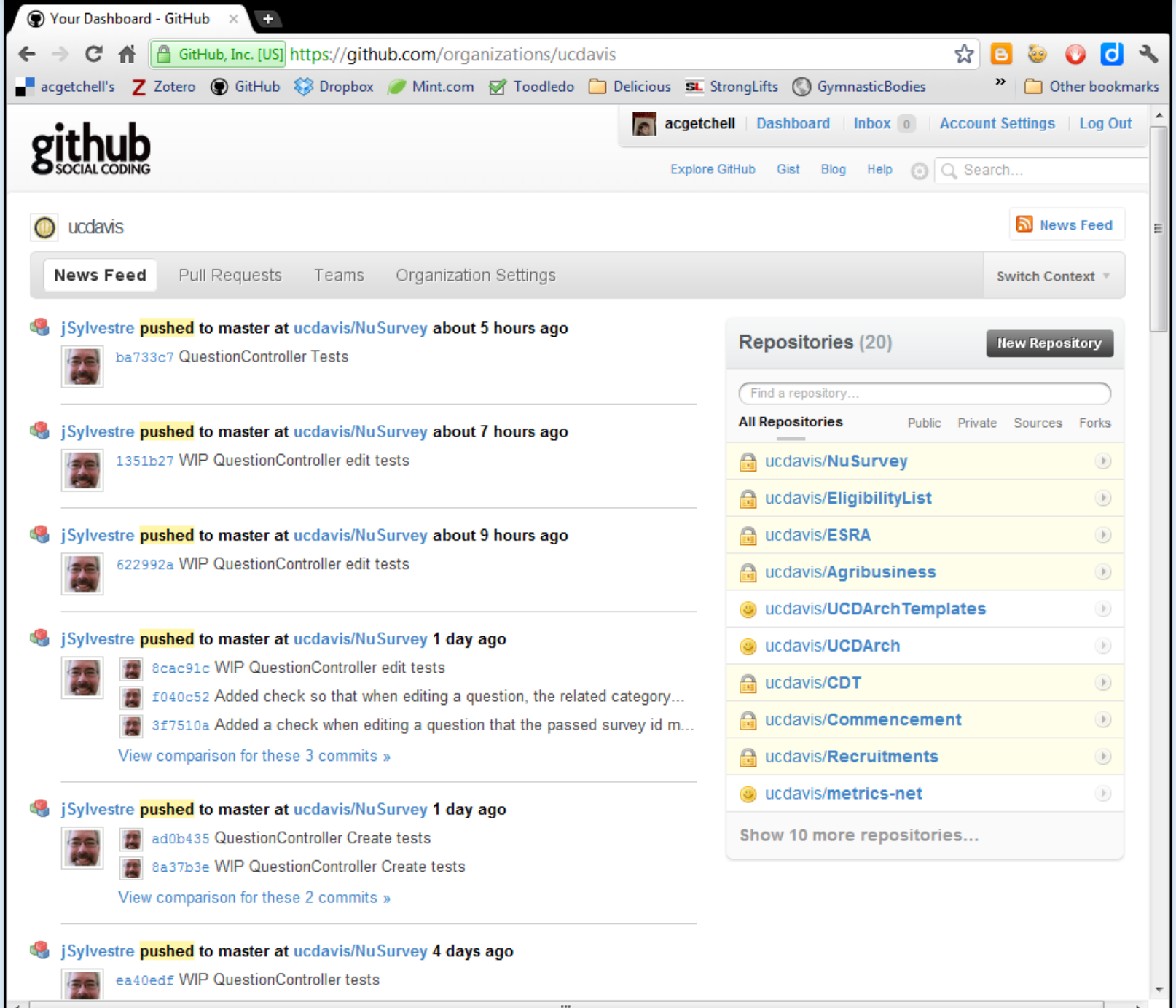
Source Control! But …

- It should be available, always
- It should be secure
- It should checksum files and keep readable version histories
- It should be easy to manage projects, collaborate with folks, setup teams/perms
- It should be ~~easy to use~~ able to do hard things like fork/branch/merge
- It should allow for distributed development

Git

- Cross Platform/IDE friendly: Integrates into Visual Studio, Eclipse, and Xcode 4.x
- Distributed Development: Each developer gets a local copy of the entire development history, and changes are copied from one repo to another. Git protocol is fast/efficient and SSH friendly
- Non-linear development: Rapid, convenient branching and merging
- Efficient handling of large projects: Very fast, scales well, typically from one to several orders of magnitude faster than other version control systems. Efficient packed format for long-term revision storage.
- Cryptographic authentication of history: History is stored such that a particular revision/commit depends upon the complete development history leading up to that commit. It's therefore not possible to change old versions without being noticed. Tags can be cryptographically signed.
- Toolkit: Git is a collection of many small tools plus scripts as convenient wrappers. Easily used by humans and automation.





GitHub@UCDavis

Organization: https://github.com/ucdavis

- Public and private repositories
- Anyone with a free GitHub account can become a team member
- We currently have 10 Private and unlimited Public repositories that allow for unlimited collaborators (team members)
- $18.75/month (25% educational discount) for 10 Private repos

Repository:

- A location storing a snapshot of a particular project, plus commit history and subsequent changes.

GitHub processes

Fork + Pull: Contributors fork the code (clone) and work on their own repository, then submit pull requests

- Unlike previous version control systems, every repository can be a master ← by convention only

Push: Agree upon a master repository, then push changes to it

- Use branch/merge to minimize conflicts

Some Caveats

Don’t store sensitive configuration info (LDAP/API keys, database passwords, etc.)

Scott Kirkland’s method (.NET)

1. All sensitive information should be stored in the <appSettings> section.
2. Create a file in your website root called AppSettings.config.

AppSettings.config:

```
<appSettings>
    <add key="WOPRpassword" value="Joshua"/>
    <add key="MapsKey" value="1234567890-AA"/>
</appSettings>
```

Some Caveats

In your web.config, add the following property to your <appSettings> element: **file=”AppSettings.config”**

Web.config:

```
<appSettings file=”AppSettings.config”>
    ...
</appSettings>
```

3. At this point, the appSettings values in your web.config will be overwritten by those in AppSettings.config. Remove any sensitive information from the web.config file now, and you can also remove non-sensitive settings from AppSettings.config if desired (optional) because anything in AppSettings.config will overwrite matching values (or non-existent values) in your web.config section.

4. The last step is to make sure that your new secret AppSettings.config file is not checked into source control, and for that you can just add **AppSettings.config** to your **.gitignore** file. Also, please think of the children and be sure to copy your AppSettings.config file to the project share so others can find it (ex: W:\Devel\Projects\Recruitment).

Some Caveats

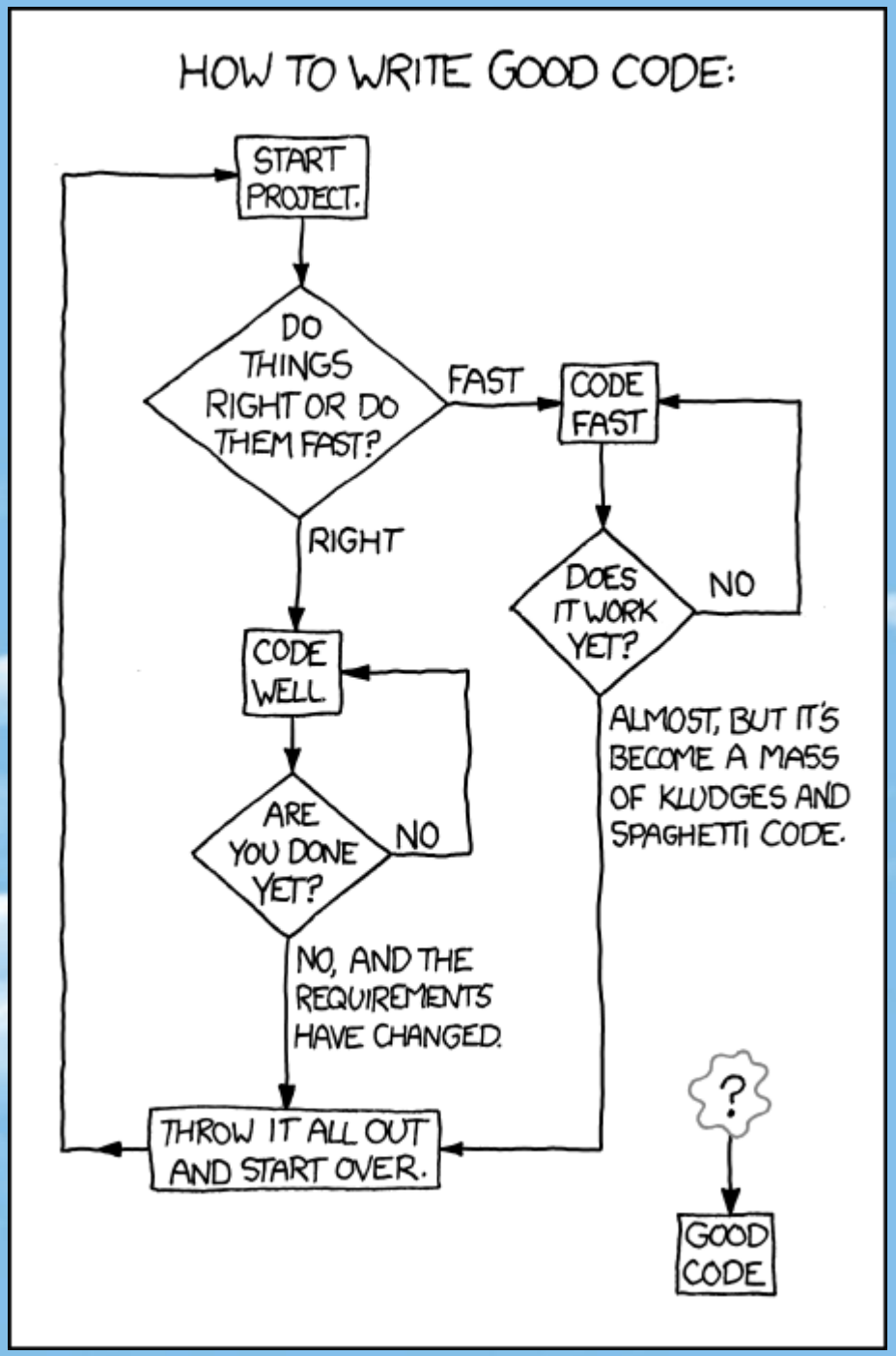
If you accidentally committed sensitive info to a repo:

1. Change the password
2. Purge file containing sensitive info from repo
3. Cleanup and reclaim space
4. Tell collaborators to use *git rebase*

Full details here: http://help.github.com/remove-sensitive-data/

What have we accomplished?

- Simplicity
- Self-organizing teams
- Regular adaption to changing circumstances
- Projects are built around motivated, trusted individuals
- Sustainable development, able to maintain a constant pace
- Welcome changing requirements, even late in development
- Continuous attention to technical excellence and good design



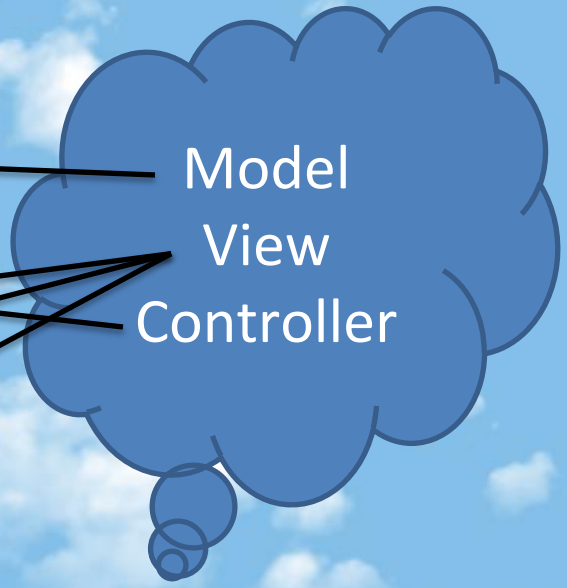
http://xkcd.com/844/

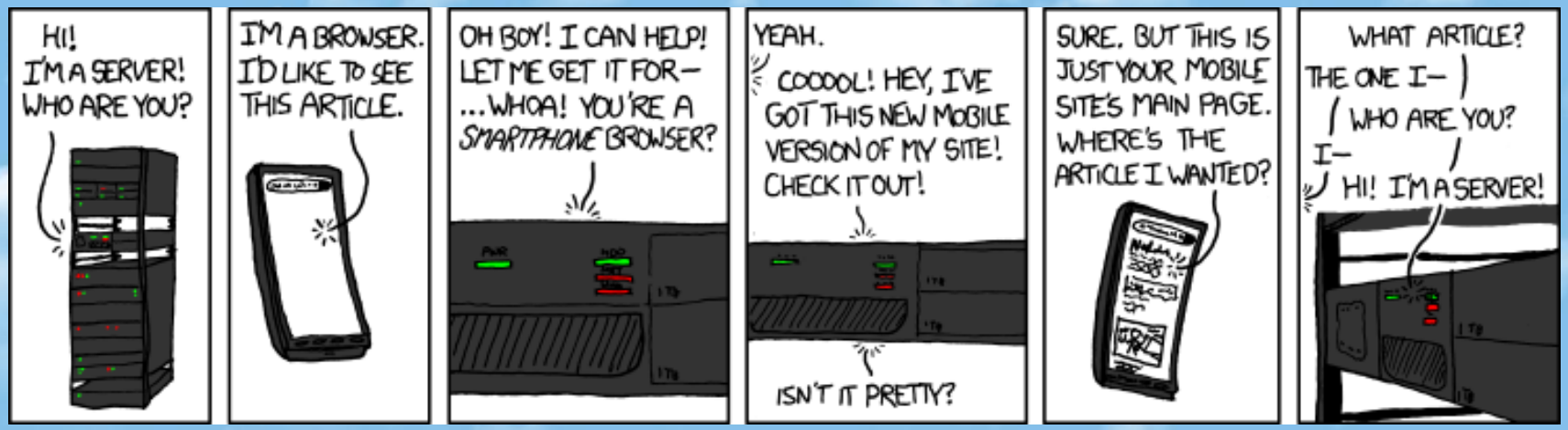
Secure Agile Cloud Development

- We've been talking Agile, and Cloud, but this is the IT ***Security*** Symposium after all
- Security comes from correct code
- Correct code comes from validation
- Validation comes from testing and measurement

Interlude: REST and MVC

- Client-server
- Layered
- Stateless
- Cacheable
- Uniform Interface





http://xkcd.com/869/

Test-Driven Development

Big topic!

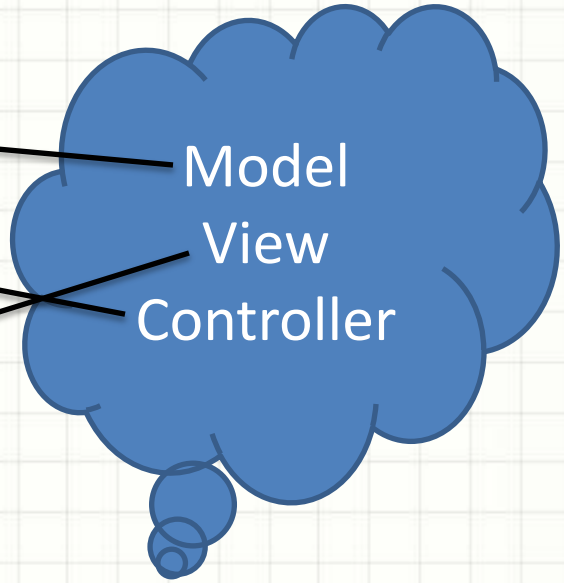
Basic idea: write functional tests that verify that things work as expected!

For a very thorough presentation, see Jason Sylvestre's "How To Write Unit Tests":

https://github.com/jSylvestre/ResharperTemplates/blob/master/How%20To%20Write%20Unit%20Tests.pptx

Types Of Unit Tests

- Repository Tests
- Controller Tests
- Interface Tests



Jason Sylvestre, How to Write Unit Tests,
https://github.com/jSylvestre/ResharperTemplates

Repository Tests

- Test Database actions
- Done on an in memory SQL Lite database created from the mapping file.
- This is done by using UCDArch's base class in testing " FluentRepositoryTestBase"

Jason Sylvestre, How to Write Unit Tests, https://github.com/jSylvestre/ResharperTemplates

Controller Tests

- Mapping Tests
- Boundary Tests of all public methods
- Reflection Tests

Jason Sylvestre, How to Write Unit Tests,
https://github.com/jSylvestre/ResharperTemplates

Interface Tests

- Sometimes these need to be run within the structure of a Controller test
- Check that expected parameters are passed
- Check that any actions or return values are what is expected
- Even if you don't write tests for your interfaces, you just have one place to review the logic.

Jason Sylvestre, How to Write Unit Tests,
https://github.com/jSylvestre/ResharperTemplates

Stages of Agile Cloud Development

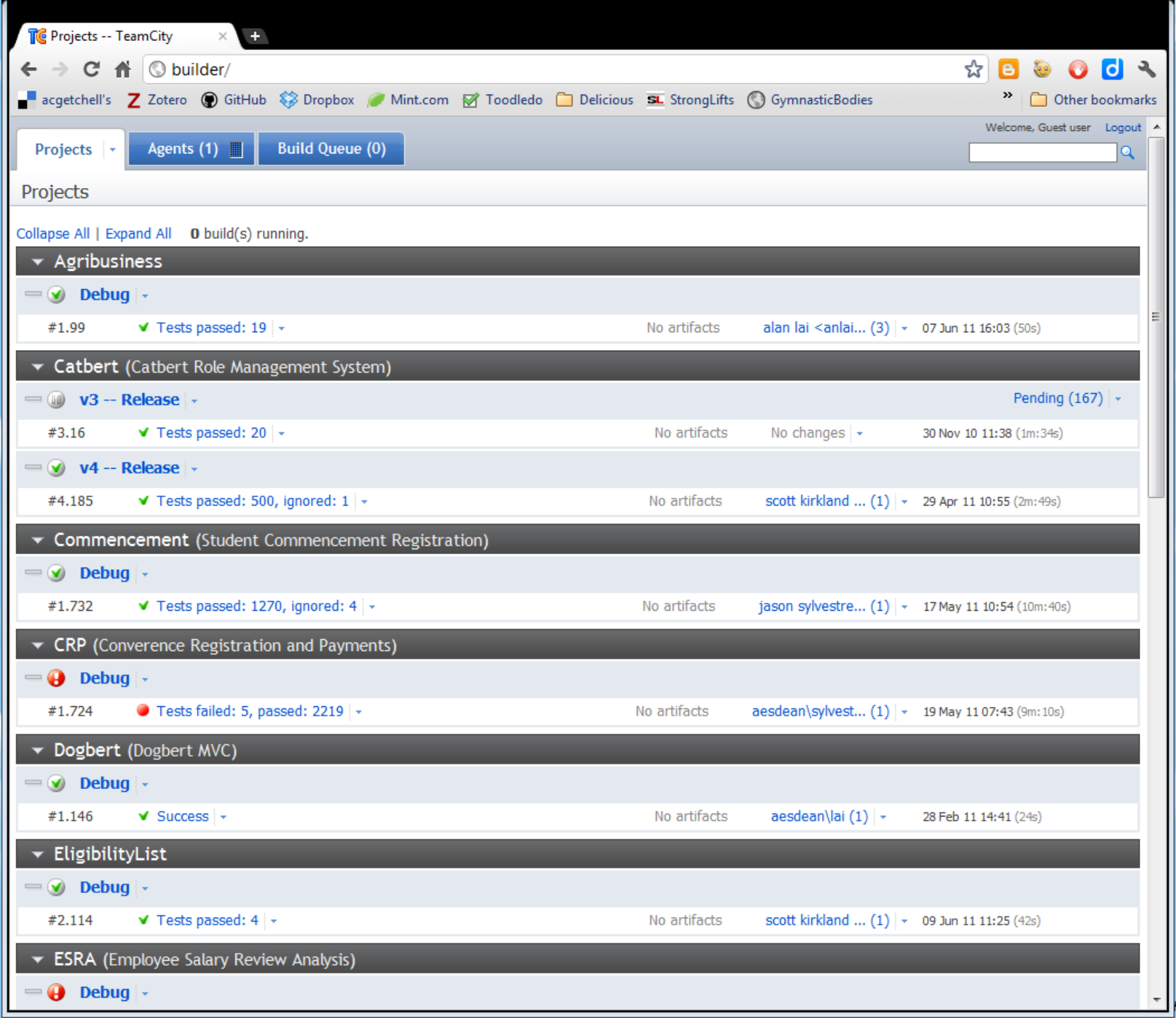
1. Developing using cloud-based source control management (SCM) repositories
2. Build in the cloud using virtual images, provide feedback to continuous integration (CI) servers
3. Testing in the cloud by building test machines
4. Production in the cloud, automated deployment, load-balancing, analytics

Continuous Integration with Team City

http://www.jetbrains.com/teamcity/

"Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily – leading to multiple integrations per day. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly."

– Martin Fowler


Projects -- TeamCity
builder/
acgetchell's
Zotero
GitHub
Dropbox
Mint.com
Toodledo
Delicious
StrongLifts
GymnasticBodies
Other bookmarks
Welcome, Guest user
Logout
Projects
Agents (1)
Build Queue (0)
Projects
Collapse All | Expand All
0 build(s) running.
Agribusiness
Debug
#1.99
Tests passed: 19
No artifacts
alan lai <anlai... (3)
07 Jun 11 16:03 (50s)
Catbert (Catbert Role Management System)
v3 -- Release
Pending (167)
#3.16
Tests passed: 20
No artifacts
No changes
30 Nov 10 11:38 (1m:34s)
v4 -- Release
#4.185
Tests passed: 500, ignored: 1
No artifacts
scott kirkland ... (1)
29 Apr 11 10:55 (2m:49s)
Commencement (Student Commencement Registration)
Debug
#1.732
Tests passed: 1270, ignored: 4
No artifacts
jason sylvestre... (1)
17 May 11 10:54 (10m:40s)
CRP (Converence Registration and Payments)
Debug
#1.724
Tests failed: 5, passed: 2219
No artifacts
aesdean\sylvest... (1)
19 May 11 07:43 (9m:10s)
Dogbert (Dogbert MVC)
Debug
#1.146
Success
No artifacts
aesdean\lai (1)
28 Feb 11 14:41 (24s)
EligibilityList
Debug
#2.114
Tests passed: 4
No artifacts
scott kirkland ... (1)
09 Jun 11 11:25 (42s)
ESRA (Employee Salary Review Analysis)
Debug

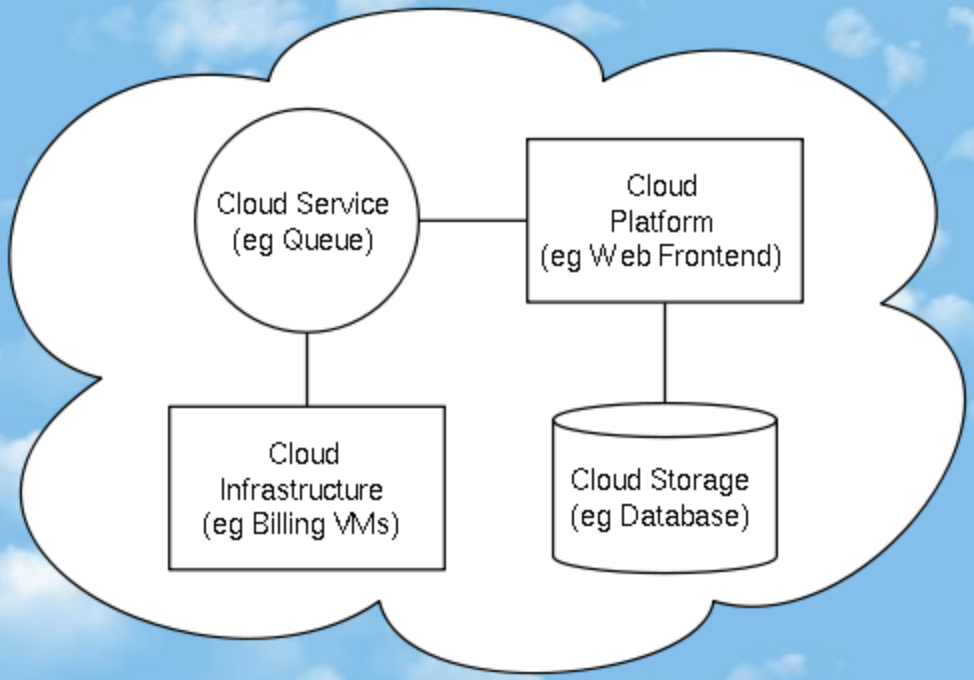
TeamCity

- Free for small/medium teams (20 build configs) or open source projects
- Works with .NET, Java, and Ruby
- Integrates with GitHub, Perforce, Subversion, Team Foundation Server, etc.
- Easy to configure for CI (e.g. every time code is checked into a repo)
- History builds → quickly identify unsafe modifications, restore lost artifacts (files, etc.)
- Build infrastructure scales using Amazon's EC2 (configure virtual build agents to start instances using AMIs)

What have we accomplished?

- Simplicity
- Self-organizing teams
- Regular adaption to changing circumstances
- Projects are built around motivated, trusted individuals
- Sustainable development, able to maintain a constant pace
- Welcome changing requirements, even late in development
- Continuous attention to technical excellence and good design

Cloud Applications



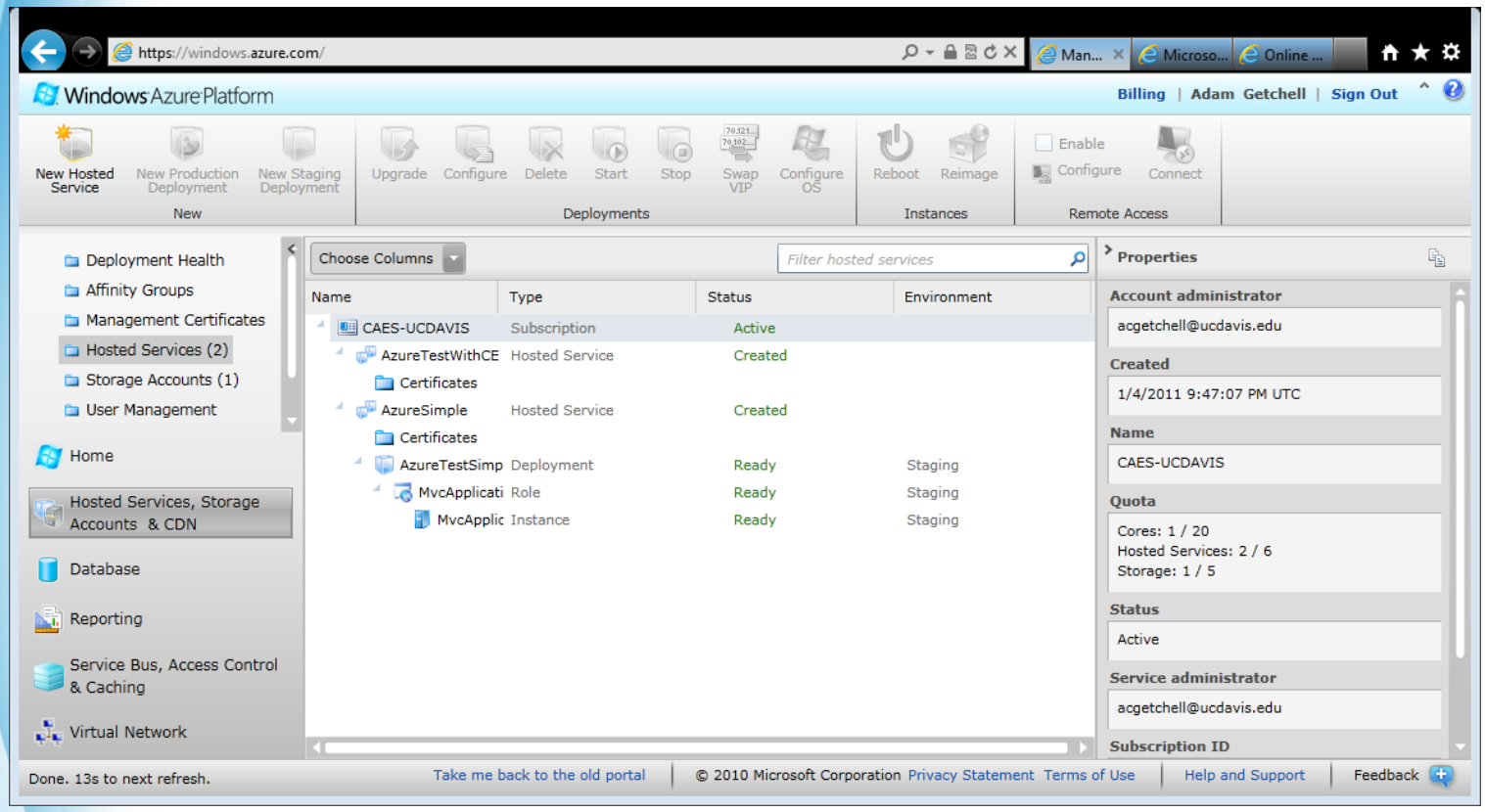
Stages of Agile Cloud Development

1. Developing using cloud-based source control management (SCM) repositories
2. Build in the cloud using virtual images, provide feedback to continuous integration (CI) servers
3. Testing in the cloud by building test machines
4. Production in the cloud, automated deployment, load-balancing, analytics

Azure

http://www.microsoft.com/windowsazure/

- .NET, Java (Hadoop), PHP and Ruby
- Used by Apple's iCloud*

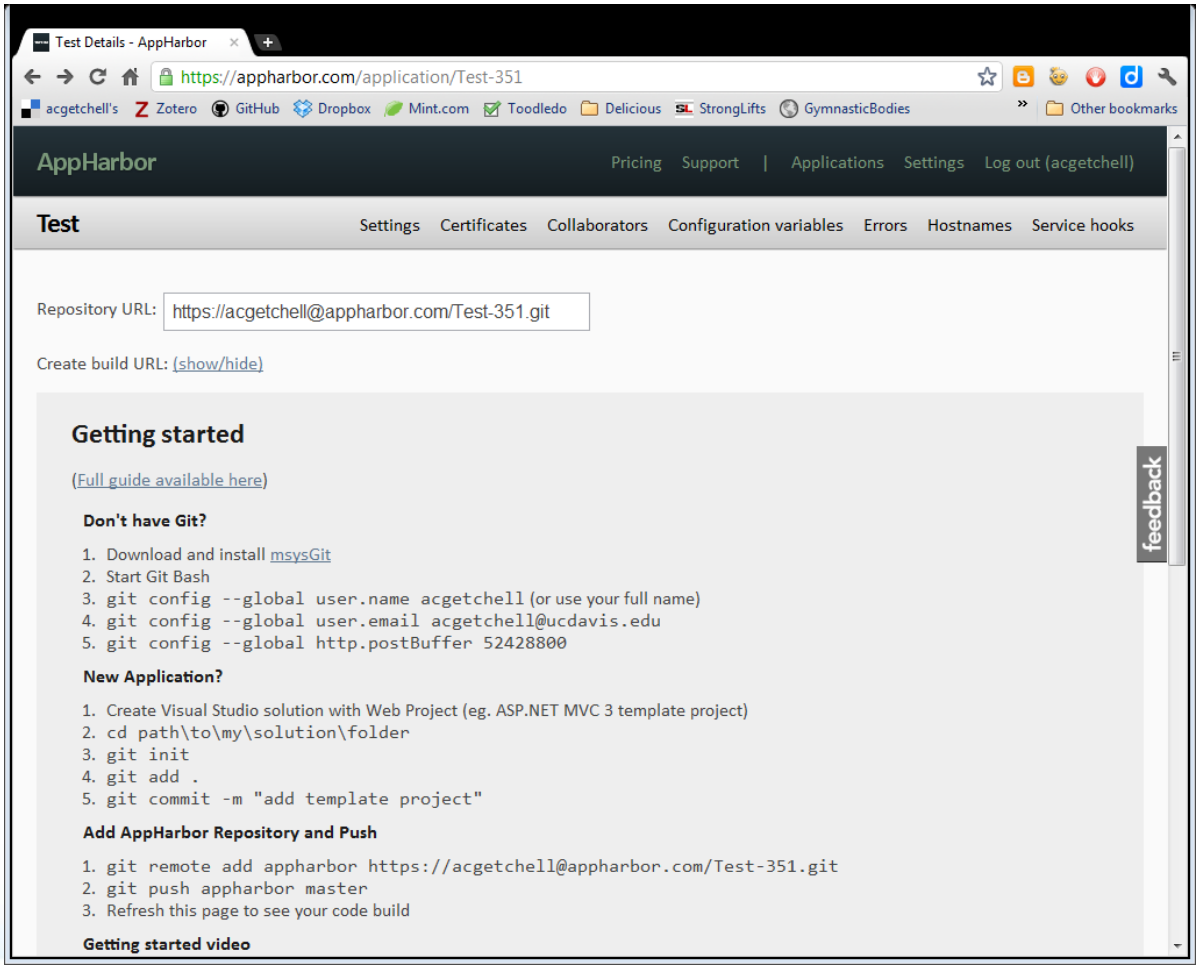


* http://www.zdnet.com/blog/microsoft/is-apple-really-using-windows-azure-to-power-icloud/9687

AppHarbor

https://appharbor.com/

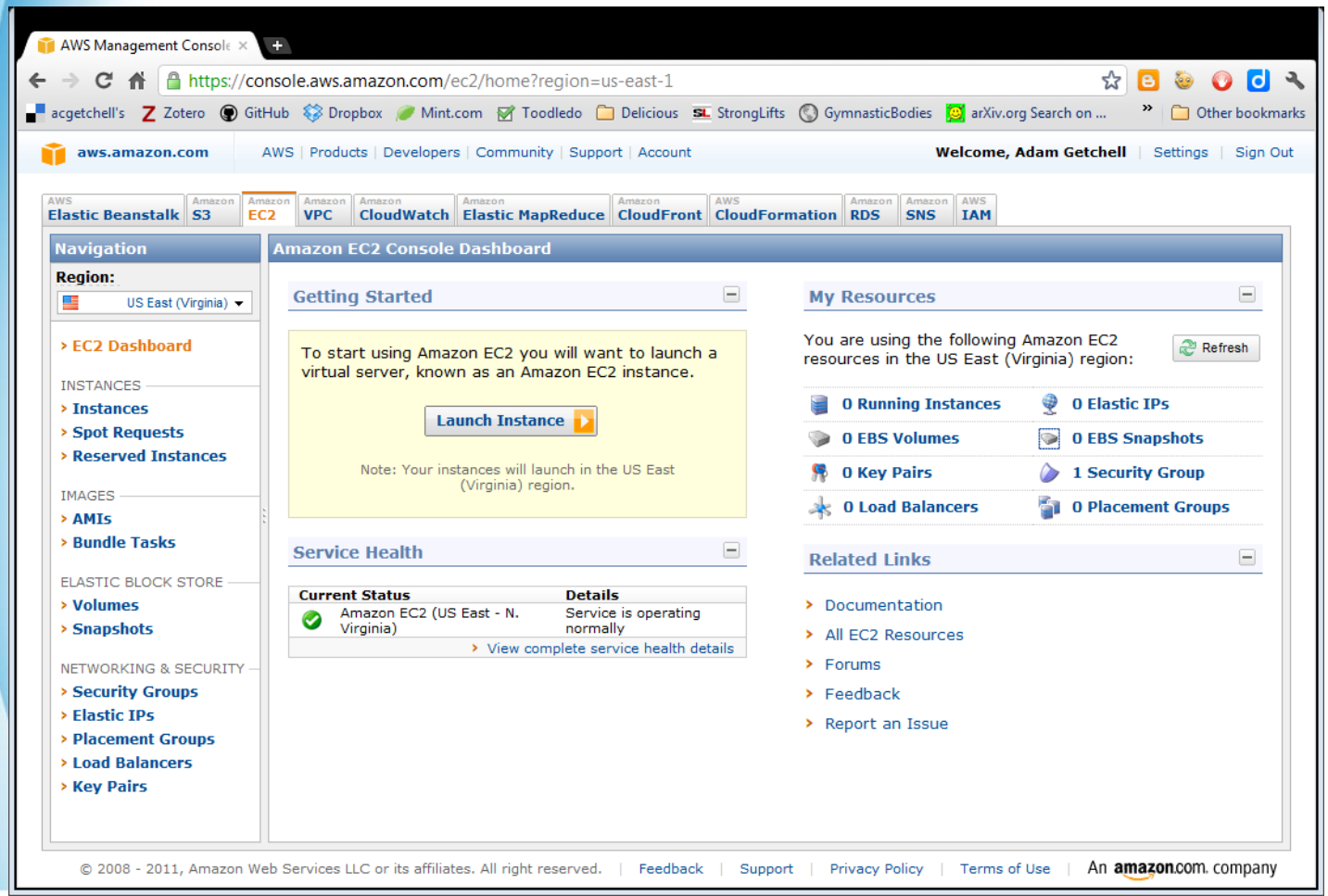
- "Azure done right"
- .NET deployment integrated with Git



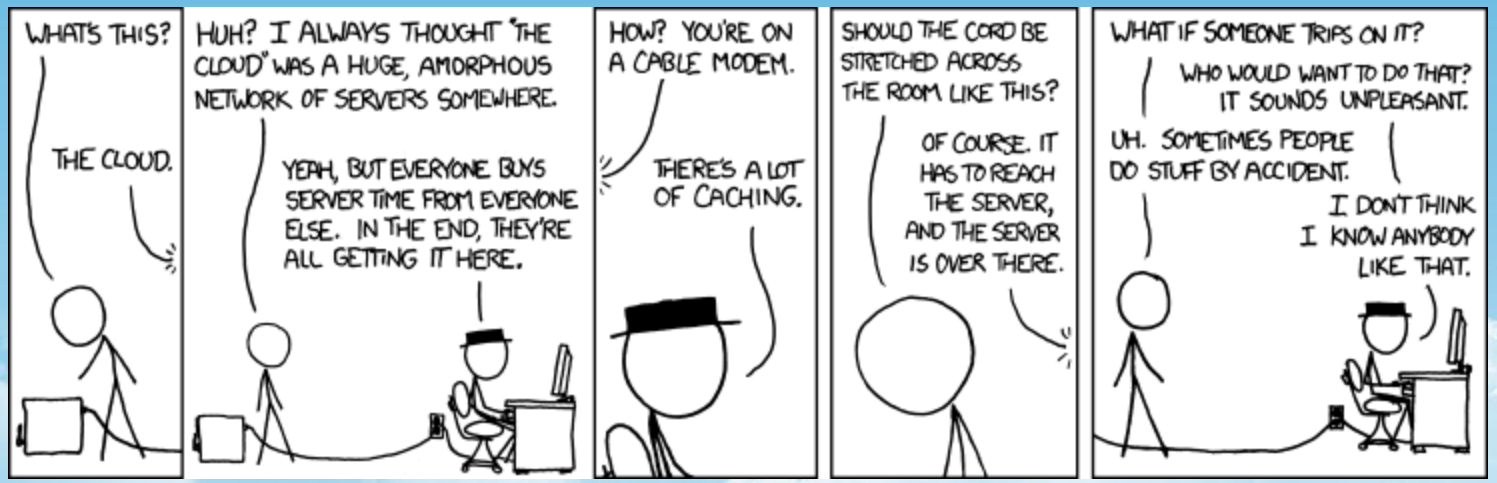
Amazon EC2

http://aws.amazon.com/about-aws/

- High-profile failure
- Known-good redundancy methods, better monitoring now



The Real Reason Amazon EC2/S3 Went Down



http://xkcd.com/908/

Other Cloud Application hosting

- Heroku (http://www.heroku.com)
  - Ruby, Node.js
  - Git friendly
  - $0.05/hour/web process ($36/month)
  - $.05/hour/background process
- Engine Yard (http://www.engineyard.com/)
  - Ruby
  - GitHub friendly
  - Production Application (2 medium load-balance app instances, 1 medium database instance) ~$534.00/month
- Rackspace (http://www.rackspace.com/)
  - Cloud Servers (virtual instance)
  - $0.08/hour/1GBRAM & 40GB disk
  - OpenStack for private clouds
    - http://www.rackspace.com/cloudbuilders/openstack/

Nice, but what about the data?

Can we store public data in the public cloud?

Can we store business data in the public cloud?

Can we store Restricted Information* in the public cloud?

"Restricted information describes any confidential or personal information that is protected by law or policy and that requires the highest level of access control and security protection, whether in storage or in transit. The term "restricted" should not be confused with that used

by the UC managed national laboratories where federal programs may employ a different classification scheme."

* http://www.ucop.edu/ucophome/policies/bfb/is3.pdf

Nice, but what about the data?

~~Can~~ May we store public data in the public cloud?

~~Can~~ May we store business data in the public cloud?

~~Can~~ May we store Restricted Information in the public cloud?

Policy Exception so far (March-September 2011):

- No Restricted Information shall be remotely available in Azure (cloud services)

Cloud Data hosting

Relational Database Management Systems RDBMS vs. Key-Value ("NoSQL")

Why do Clouds prefer NoSQL?

RDBMS

Provides:

- Atomicity: All operations in a transaction will complete, or none will
- Consistency: The database will be in a consistent state when the transaction begins and ends
- Isolation: The transaction will behave as if it is the only operation being performed upon the database
- Durability: Upon completion of the transaction, the operation will not be reversed

Brewer's CAP Theorem*

Web services cannot ensure all three of the following properties at once:

- Consistency: The client perceives that a set of operations has occurred all at once
- Availability: Every operation must terminate in an intended response
- Partition tolerance: Operations will complete even if individual components are unavailable

In order to scale, we must have partitioning and availability. Thus we must give up consistency.

* http://citeseer.ist.psu.edu/544596.html

BASE vs. ACID*

BASE is Basically Available, Soft State, Eventually Consistent

- Essentially, you must find ways to relax your constraints
- For example, if you have a ticket selling application, you must assume that there may be tickets that have been bought but not yet recorded as purchased
- Application errors = data errors
  - RDBMS are internally consistent, by contrast

* http://queue.acm.org/detail.cfm?id=1394128

Why NoSQL?*

- BASE allows scalability for clouds
- No Entity Joins (data is denormalized), so faster
- Data model matches data structures in code better

* http://www.readwriteweb.com/enterprise/2009/02/is-the-relational-database-doomed.php

Think carefully about whether you truly need Relational Data

It will be more expensive and slower

But, so far we have not needed to scale out drastically and have stuck with RDBMS

Cloud NoSQL data sources

- Amazon SimpleDB (http://aws.amazon.com/simpledb/)
  - Automatic geo-redundant replication within Region
  - Automatic indexing
  - Simple API
- Redis (http://redis.io/)
  - Open-source (powers GitHub) (http://code.google.com/p/redis/)
  - Excellent API for programmer
  - Dozens of clients for most programming languages

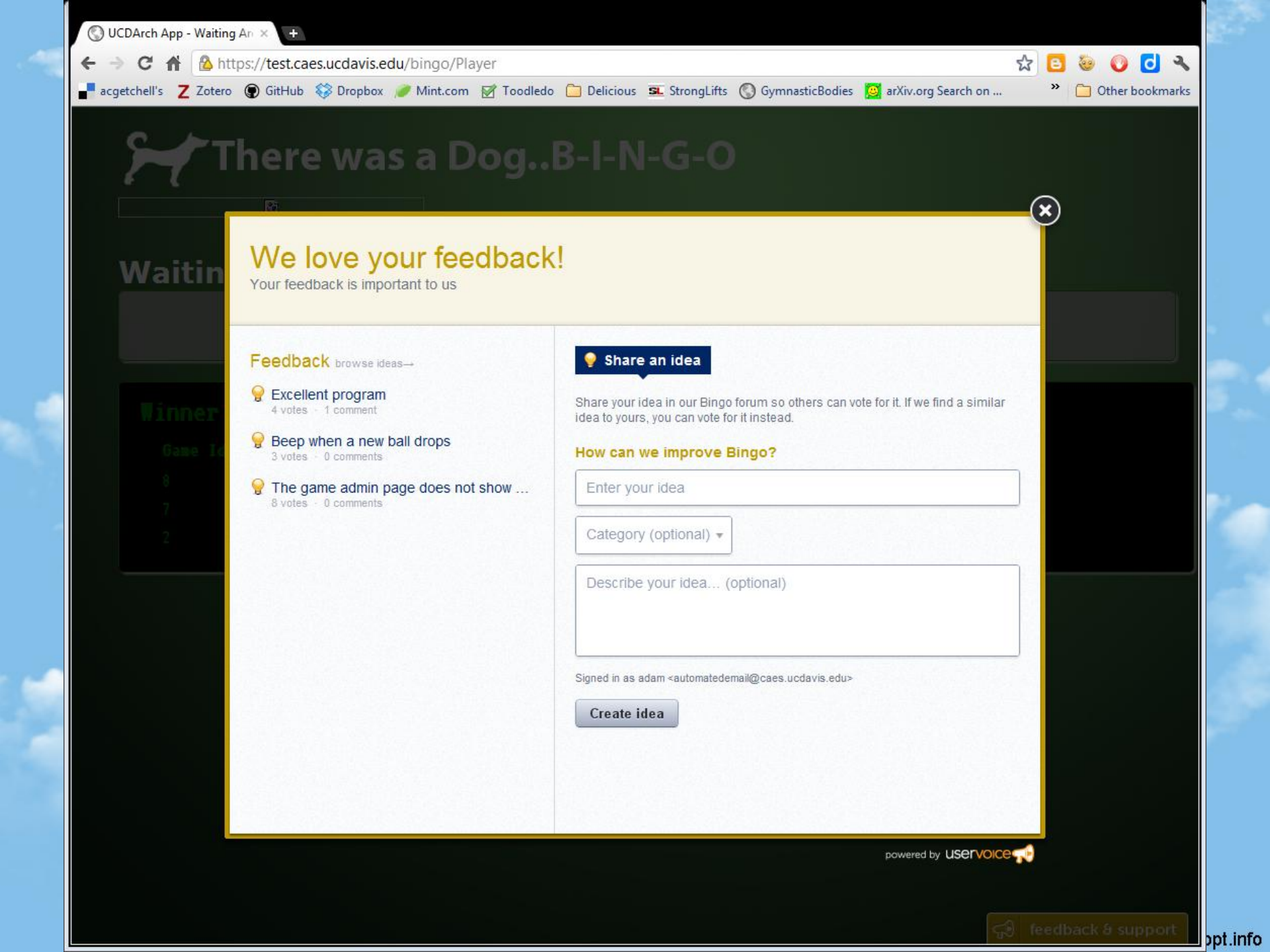
- (Didn't mention Google AppEngine Data Store, because you can't use it outside of AppEngine.)

Cloud SQL data sources

- Microsoft SQL Azure (http://www.microsoft.com/windowsazure/sqlazure/database/)
  - High-availability and fault tolerance
  - T-SQL syntax
- AppHarbor MySQL or Microsoft SQL Server
  - Shared servers $10-10GB/month, dedicated server costs extra
- Amazon EC2 Running Microsoft Windows Server & SQL Server
  - No change for SQL/.NET developers
- Rackspace Cloud Server running Microsoft SQL Server 2008 R2

Agile Development Feedback

- Working software is the principal measure of progress
- Customer satisfaction by rapid delivery of useful software
- Close co-operation between business people and developers
- Face-to-face conversations as the best form of communication
- Working software is delivered frequently (weeks rather than months)


UCDArch App - Waiting Ar
https://test.caes.ucdavis.edu/bingo/Player
acgetchell's
Zotero
GitHub
Dropbox
Mint.com
Toodledo
Delicious
StrongLifts
GymnasticBodies
arXiv.org Search on ...
Other bookmarks
There was a Dog..B-I-N-G-O
Waitin
Winner
We love your feedback!
Your feedback is important to us
Feedback
browse ideas→
Excellent program
4 votes · 1 comment
Beep when a new ball drops
3 votes · 0 comments
The game admin page does not show ...
8 votes · 0 comments
Share an idea
Share your idea in our Bingo forum so others can vote for it. If we find a similar idea to yours, you can vote for it instead.
How can we improve Bingo?
Enter your idea
Category (optional)
Describe your idea... (optional)
Signed in as adam <automatedemail@caes.ucdavis.edu>
Create idea
powered by uservoice
feedback & support

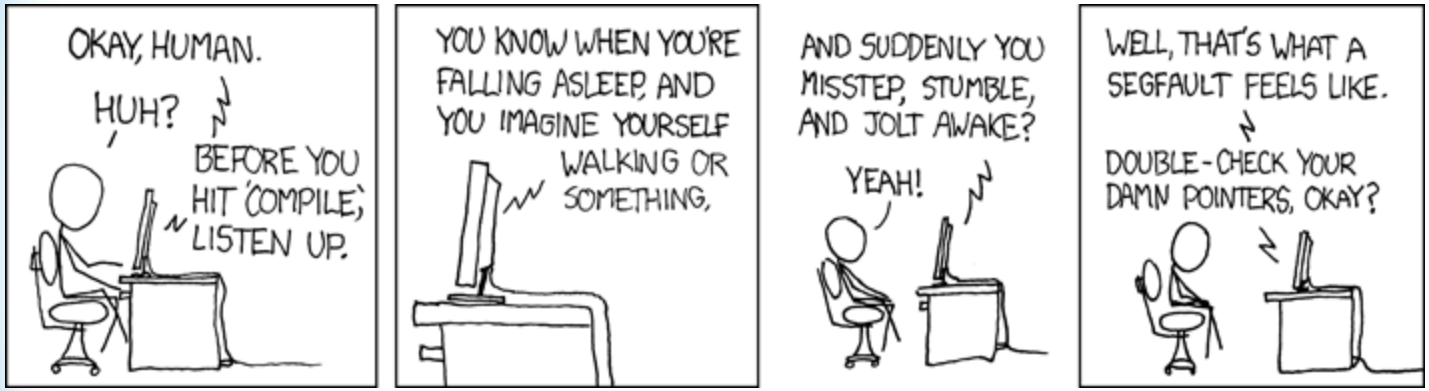
UserVoice

- Single Sign-on (we integrated it with CAS)
- Public API
- iPhone SDK
- Facebook Pages
- Domain aliasing
- Custom logo & colors
- CSS customization
- 10 forums
  - File attachment
  - Private forums
- Feedback Widget
- Excel export
- $45/month

Measure Your Code

See Coda Hale's "Metrics, Metrics Everywhere"

- http://codahale.com/codeconf-2011-04-09-metrics-metrics-everywhere.pdf



http://xkcd.com/371/

Coda Hale's Rap

- Code is business value
- Business value is new features, improved features, fewer bugs, making future changes easier, adding unit tests
- Make better decisions about code
- Code generates business value when it *runs*, not when it is *written*

Coda Hale's Rap

- Thus, we need to know what code is doing when it is running
- We need to measure it when it is running and generating business value in Production
- We have a mental model of the code, but it is often wrong

Coda Hale's Rap

- Thus, we need to know what code is doing when it is running
- We need to measure it when it is running and generating business value in Production
- We have a mental model of the code, but it is often wrong
- We also have to measure the right things

Yammer Metrics for Java/Scala

- Gauges
  - Instantaneous readings
- Counters
  - 64-bit integer with increment/decrement
- Meters
  - Increment-only counters which track mean rates and exponentially-weighted moving average (same formula as UNIX la)
- Histograms
  - Distribution measurements: count, max, min, mean, standard deviation, median, 75/95/98/99/99.9 %
- Timers
  - Duration and rate information, plus histogram info

Yammer Metrics for Java/Scala

- https://github.com/codahale/metrics

.NET/CLR version

- https://github.com/danielcrenna/metrics-net

Local fork

- https://github.com/ucdavis/metrics-net

Ruby version

- https://github.com/johnewart/ruby-metrics

References

- http://en.wikipedia.org/wiki/Agile_development
- Image on Slide 4 was uploaded to Wikipedia on 2010-07-02 18:40 by Dbenson 2700×3600× (906118 bytes) *This poster provides a good visual of the standard Agile Software Development methodology. This original work was created by VersionOne, Inc. and is licensed under the Creative Commons Attribution CC-BY-SA license*
- http://en.wikipedia.org/wiki/Cloud_computing
- Image on Slide 8 was uploaded to Wikipedia and the public domain on 2011-02-15 by Benjamin P. Griner and Philip J. Butler.
- http://git-scm.com/about

Agile, Secure Cloud Application Development & Management

Adam Getchell

acgetchell@ucdavis.edu

https://github.com/acgetchell

IT Security Symposium, June 15-16, 2011

University of California Davis