Culture Shift

How DevOps changed the way we think about IT

I like to read Wikipedia for fun (joke)

Favorite articles are the "lists of things"—cognitive biases; latin phrases; global metropolitan areas by population

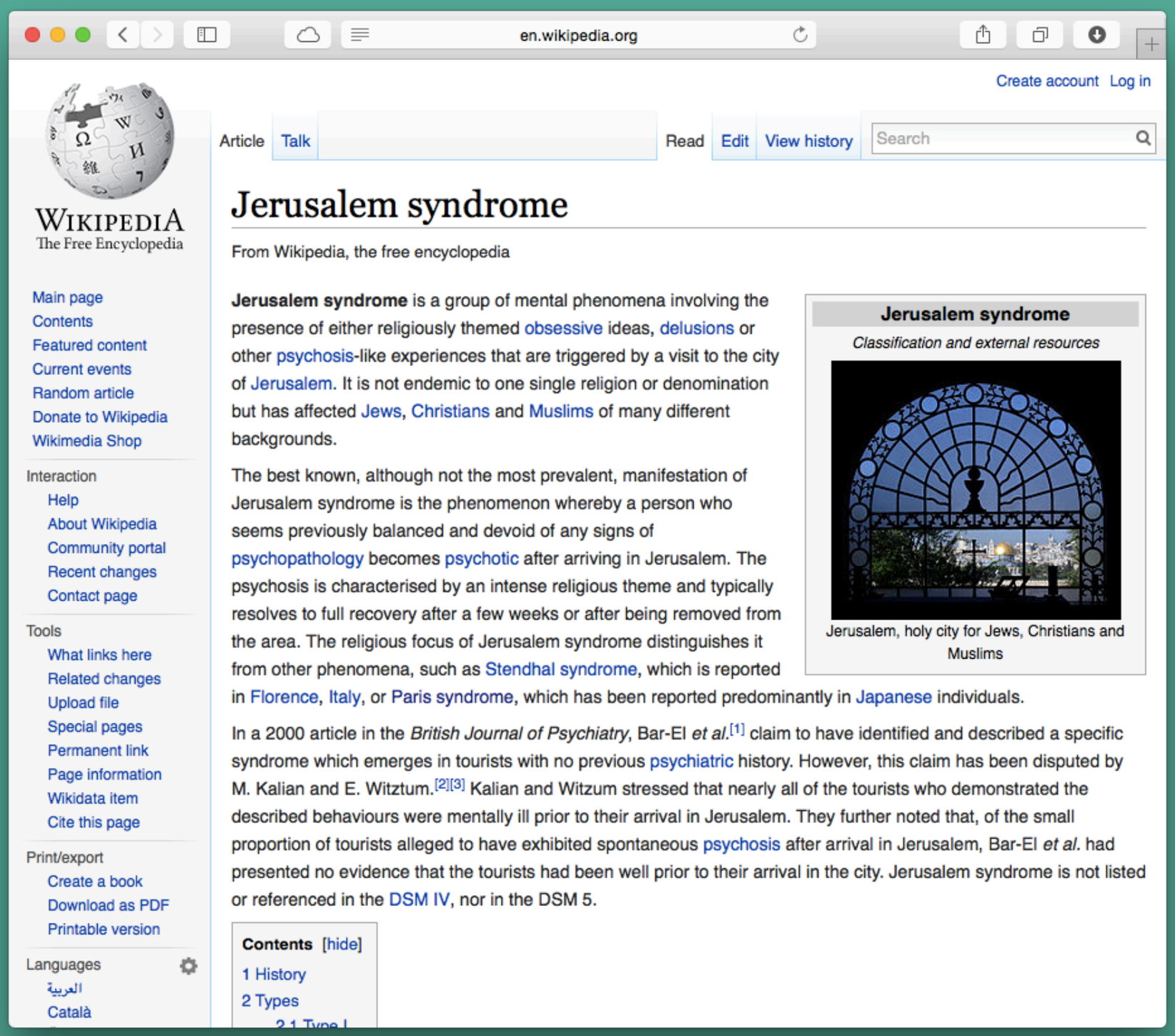
Found a new favorite: Culture-bound syndromes

Defined as—a combination of psychiatric and somatic symptoms that are considered to be a recognizable disease only within a specific society or culture.

Think about that for a minute

Paris syndrome: a transient psychological disorder encountered by some individuals visiting or vacationing in Paris. It is characterized by a number of psychiatric symptoms such as acute delusional states, hallucinations, feelings of persecution...

According to several medical journals, Japanese visitors are observed to be especially susceptible.



My favorite: Jerusalem Syndrome—phenomenon whereby a person who seems previously balanced and devoid of any signs of psychopathology becomes psychotic after arriving in Jerusalem.

Types (types!)

Type I—affected person believes themself to be a Messiah

Type III—variety of behaviors, including the preparation—often with the aid of hotel bed-linen—of a long, ankle-length, toga-like gown, which is always white

Culture Shift

How DevOps ~~changed~~ changes the way we think about IT

Brings us back to our topic today—essentially the story of our journey at the College of Architecture over the past year or so

CHANGED implies finality—we are certainly not finished our journey. We are still learning how to apply these ideas and improve

Roadmap

1. Why Culture?
2. Some Context
3. Meet DevOps
4. DevOps, meet Academia
5. Open Discussion

Culture

The outlook, attitudes, values, morals, goals, and customs shared by a group.

We've all seen this definition before

Culture (more simply)

Culture is the way you think, act, and interact

IMPORTANT: much of culture is invisible—assumptions—background process

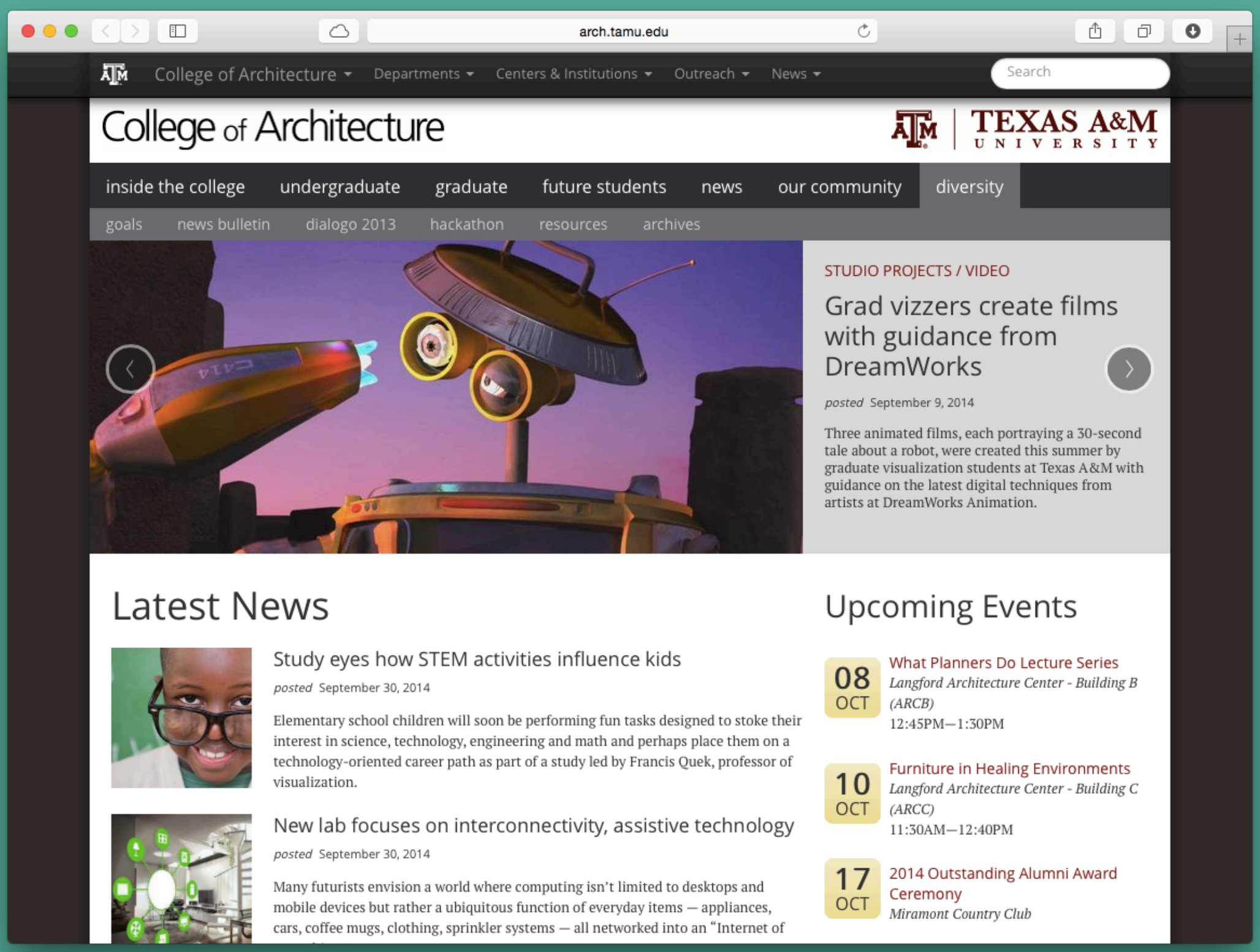
"Culture eats strategy for lunch"

The Context

The College of Architecture at Texas A&M

College of Architecture

— 4 departments
— 8 research centers and institutes
— 500 employees
— 10,000 (3,000) students
— locations across Texas and the globe



Work we do: web; infrastructure; business applications; automated tools to manage student accounts and data (!!); classroom technology; rendering pipeline and infrastructure; research computing services (Iaas, Paas)

The Problem

Do we have a problem?

IT has always been like this

No dramatic stories of epic failure, just regular IT woes—fighting fires, can't seem to get ahead, can't seem to effectively coordinate operations and development; no clear picture of work capacity

~3 years ago, one of my employees (UNIX admin) started looking at automating parts of his job

Several attempts, some false starts, then he came into my office one day and said --

The Solution

"Dude—you HAVE to read this book"
So I did—in one sitting. It just clicked.
Then I bought copies for all my team, and made them read it too (Really. I assigned it.)

The Phoenix Project

A management fable in the style of E. Golrdratt's *The Goal*.

Theory of Constraints — Systems management

The Goal was a book published in 1984 about systems management and his Theory of Constraints.

The surprise is that a 30-y-o business theory about manufacturing has so many things to teach us about IT.

Summarize TPP—Bill takes over failing project, cast of characters (archetypes); you'll all recognize at least a few

Meets a mentor that helps him understand why his inherited IT project is failing and how to fix it—DevOps

What is DevOps?

DevOps is a service delivery concept that stresses communication, collaboration, and integration between software developers and IT professionals.

DevOps, from Wikipedia

What is DevOps (alternative edition):

Developers need to understand infrastructure. Operations people need to understand code. People need to fking work with each other and not just occupy space next to each other.**

— John Vincent

DevOps - the Title Match from blog dot lusis

John Vincent is a systems engineer who worked for Dell (currently with MailChimp)

DevOps : Operations :: Agile : Development

In fact, someone reminded me that early on DevOps was even called *Agile operations*

DevOps seeks to maximize:

— predictability
— efficiency
— security
— maintainability

PRED: you want to be sure that every time a system is changed you know what state it is in

EFF: dev cycles are faster and faster—very long build times are no longer reasonable

SEC: if you have no control over the changes, you have lack security

MAIN: 80% of a system's lifecycle is in maint., not dev—any model that doesn't address this FAILS

DevOps emphasizes:

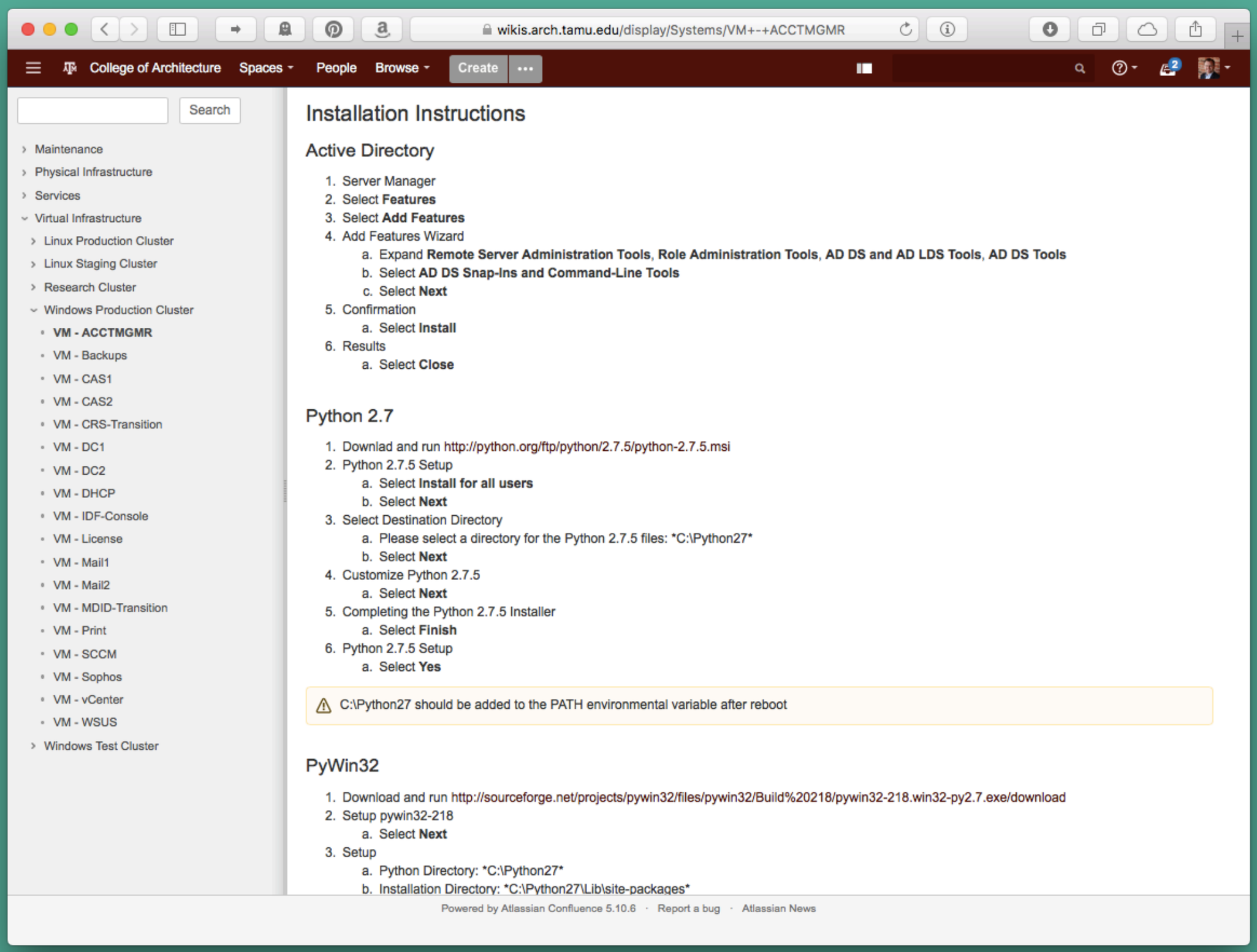
- automation whenever possible
- infrastructure as code
- continuous integration / delivery
- collaborative teams with shared responsibility

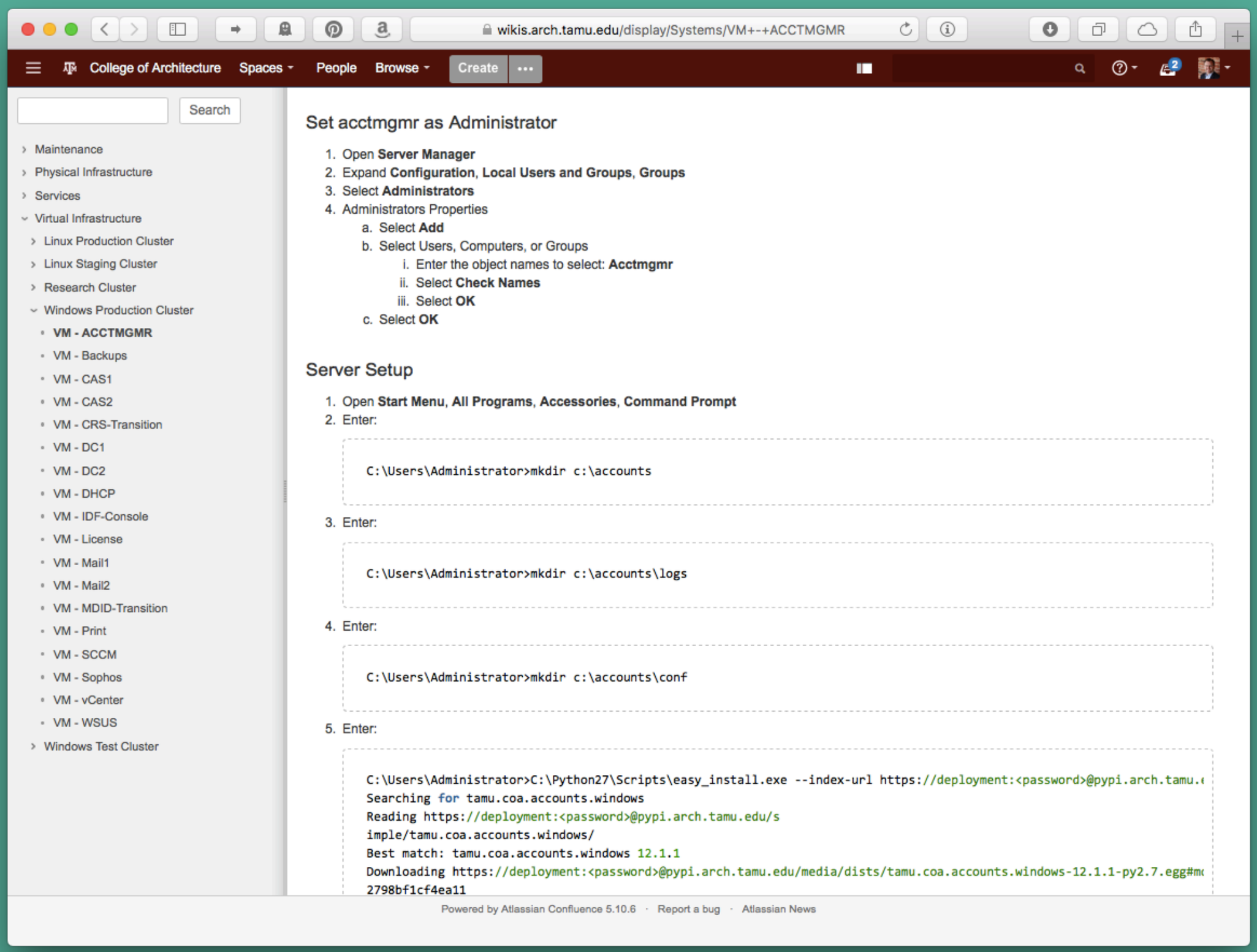
A fundamental tenet of the DevOps way is *automation*—automation is always preferable, no matter how small

Automate all the things! ;-)

One result of automation and IaC is that we this less about disaster *recovery* and more about disaster *indifference*

So our systems are not defined like this—


wikis.arch.tamu.edu/display/Systems/VM+-+ACCTMGMR
College of Architecture
Spaces
People
Browse
Create
Search
Maintenance
Physical Infrastructure
Services
Virtual Infrastructure
Linux Production Cluster
Linux Staging Cluster
Research Cluster
Windows Production Cluster
VM - ACCTMGMR
VM - Backups
VM - CAS1
VM - CAS2
VM - CRS-Transition
VM - DC1
VM - DC2
VM - DHCP
VM - IDF-Console
VM - License
VM - Mail1
VM - Mail2
VM - MDID-Transition
VM - Print
VM - SCCM
VM - Sophos
VM - vCenter
VM - WSUS
Windows Test Cluster
Installation Instructions
Active Directory
1. Server Manager
2. Select Features
3. Select Add Features
4. Add Features Wizard
a. Expand Remote Server Administration Tools, Role Administration Tools, AD DS and AD LDS Tools, AD DS Tools
b. Select AD DS Snap-Ins and Command-Line Tools
c. Select Next
5. Confirmation
a. Select Install
6. Results
a. Select Close
Python 2.7
1. Downlad and run http://python.org/ftp/python/2.7.5/python-2.7.5.msi
2. Python 2.7.5 Setup
a. Select Install for all users
b. Select Next
3. Select Destination Directory
a. Please select a directory for the Python 2.7.5 files: *C:\Python27*
b. Select Next
4. Customize Python 2.7.5
a. Select Next
5. Completing the Python 2.7.5 Installer
a. Select Finish
6. Python 2.7.5 Setup
a. Select Yes
C:\Python27 should be added to the PATH environmental variable after reboot
PyWin32
1. Download and run http://sourceforge.net/projects/pywin32/files/pywin32/Build%20218/pywin32-218.win32-py2.7.exe/download
2. Setup pywin32-218
a. Select Next
3. Setup
a. Python Directory: *C:\Python27*
b. Installation Directory: *C:\Python27\Lib\site-packages*
Powered by Atlassian Confluence 5.10.6 · Report a bug · Atlassian News



We've all seen this—we've probably asked for documentation like this to be created before.

But (almost) no one likes to do this—almost impossible.

And no one reads it. And it doesn't stay in sync with reality—configuration drift

But rather, like this—

```
include_recipe 'nginx'
include_recipe 'git'

directory '/var/www' do
 action :create
 owner 'www-data'
 group 'www-data'
end

git node['magic']['directory'] do
 user 'www-data'
 group 'www-data'
 reference 'master'
 repo 'git://github.com/TAMUArch/magic'
end

...
```


If the only way to build this server is to run this code, then you are guaranteed that the documentation (the CODE) is in sync with the running service

Main lessons from The Phoenix Project

There are 4 types of work

Work-in-progress is the silent killer

Unplanned work accrues technical debt

Bottlenecks constrain work

After I "assigned" the book to my staff, we held a 1-day retreat—left the office, and spent all day working through the book.

We'll look at the 4 types in a moment

What is WIP? Current work in the pipeline—backlog is committed work

Technical debt should be avoided whenever possible, but certainly managed

Any improvement that is not at the constraint is illusory

Four types of work

1. Business projects
2. Infrastructure projects
3. Changes
4. Unplanned work

Explain the 4 types

We then spent almost another day in a "war room" with stacks of colored index cards

Trying to get a handle on the amount of work we had in progress

The Three Ways

1. Systems thinking
2. Amplify feedback loops
3. Culture of continual experimentation and learning

The "ways" were the lessons imparted to the main character in the book by the mentor

1st way emphasizes the performance of the entire system—everyone must be concerned with the final product, and take ownership (no throw over)

2nd way talks about feedback from ops back to dev so that improvements can be made

3rd way is about creating a CULTURE of continual experimentation and improvement

Related emphases

Business value

Collaboration requires empathy and trust

Blame-free thinking

This relates back to "Systems thinking"—the actual value to the customer is always the ultimate focus

There is a strong emphasis in the DevOps movement on building strong teams with affinity—no collaboration without trust

Extending the emphasis on teams and improvement—growth of the team is more important than temporary catharsis of assigning blame

Back to Culture

Changes we made in Architecture were mostly about culture

Better communication—Got people talking

Weekly change management meeting so any changes to systems were broadcast to all members of the team

Better team organization—We've experimented with various Kanban software tools—still learning here

Better handle on work capacity than ever before

DevOps syncs well with a framework like ITIL because DevOps is primarily a way to get your team thinking about the final **service** that is delivered, and not just the narrow slice they see from their desk.

Caveats

No silver bullet

Staying the course is hard

Need commitment from the *whole* team

There are no "tools" that will magically give you DevOps—if a salesmans offers *DevOps in a box*, run away

Many of us have been doing IT the same way for a long time—easy to slip back into old ways of thinking

If the engineers aren't invested, the collaboration will not happen

If the manager is not convinced, the team will struggle to align priorities

More reading

On DevOps and organizational culture:

Empathy: The Essence of DevOps
http://blog.ingineering.it/post/72964480807/empathy-the-essence-of-devops

DevOps Culture (Part 1)
http://itrevolution.com/devops-culture-part-1/

Continuously Deploying Culture: Scaling Culture at Etsy
http://www.slideshare.net/mcdonnps/continuously-deploying-culture-scaling-culture-at-etsy-14588485

More reading

On DevOps methodologies:

The Phoenix Project: A novel about IT, DevOps, and helping your business win (IT Revolution Press)
http://itrevolution.com/books/

Where To Learn More About Concepts In "The Phoenix Project" (Part 1)
http://itrevolution.com/learn-more-about-concepts-in-phoenix-project/

About me

http://github.com/amikeal
http://linkedin.com/in/amikeal

adam@tamu.edu

Open discussion