Slide 1

DevOps is Bigger than IT: Driving Digital Transformation in Libraries

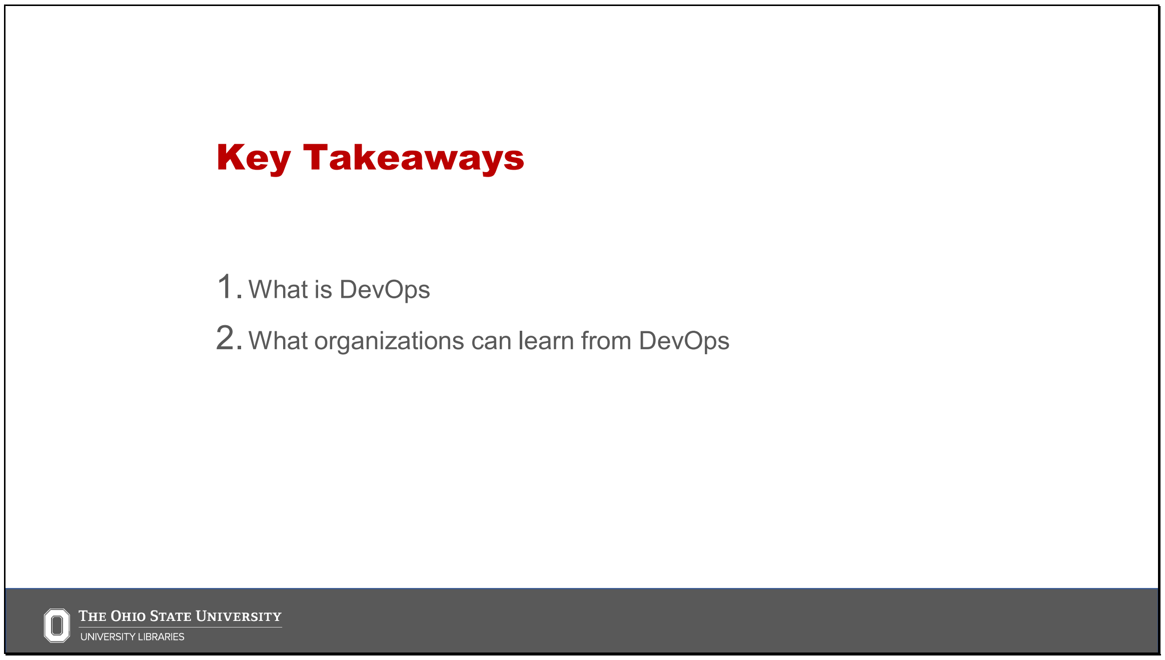
Beth Snapp (snapp.6@osu.edu)
Jennifer Vinopal (vinopal.5@osu.edu)

THE OHIO STATE UNIVERSITY
UNIVERSITY LIBRARIES

1

Presented at: Coalition for Networked Information (CNI) Spring 2020 Membership Meeting, April 21, 2020

Slide 2



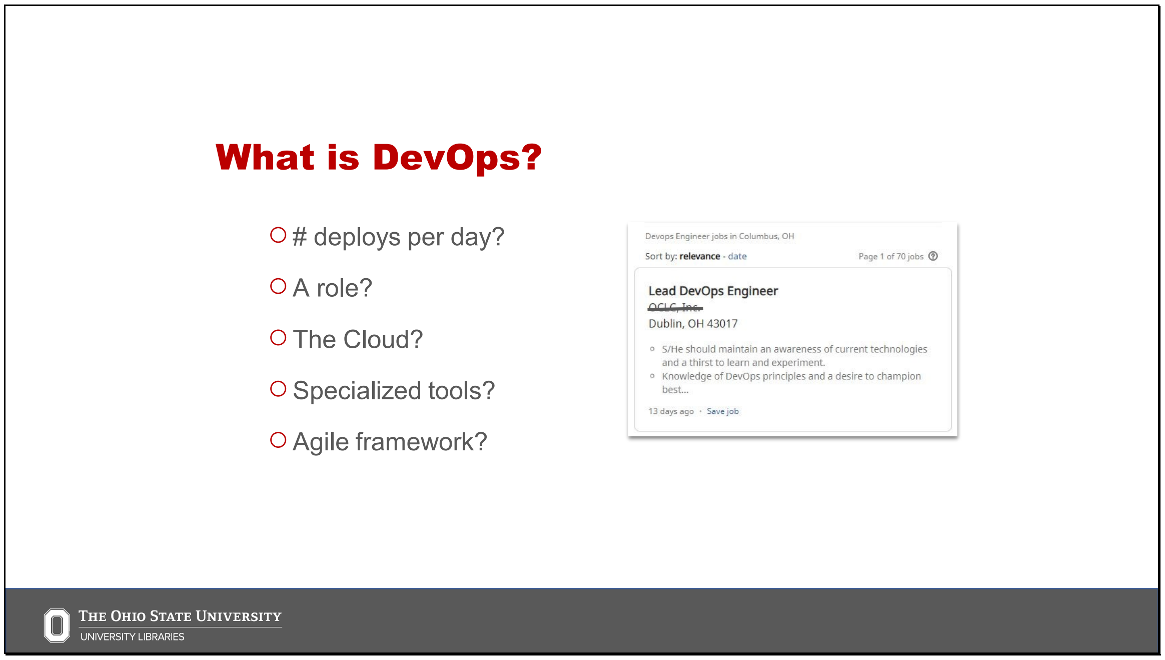
Key Takeaways

1. What is DevOps
2. What organizations can learn from DevOps

THE OHIO STATE UNIVERSITY
UNIVERSITY LIBRARIES

In our presentation today, Jennifer and I will focus on two topics: 1) what is DevOps, in particular what does a DevOps culture look like; and 2) we'll take a case study of the digital preservation program in the Ohio State University Libraries to illustrate what the organization can learn from the DevOps movement.

Slide 3



What is DevOps?

- # deploys per day?
- A role?
- The Cloud?
- Specialized tools?
- Agile framework?

Devops Engineer jobs in Columbus, OH

Sort by: **relevance** - date Page 1 of 70 jobs

Lead DevOps Engineer
~~OCLC, Inc.~~
Dublin, OH 43017

- S/He should maintain an awareness of current technologies and a thirst to learn and experiment.
- Knowledge of DevOps principles and a desire to champion best...

13 days ago · Save job

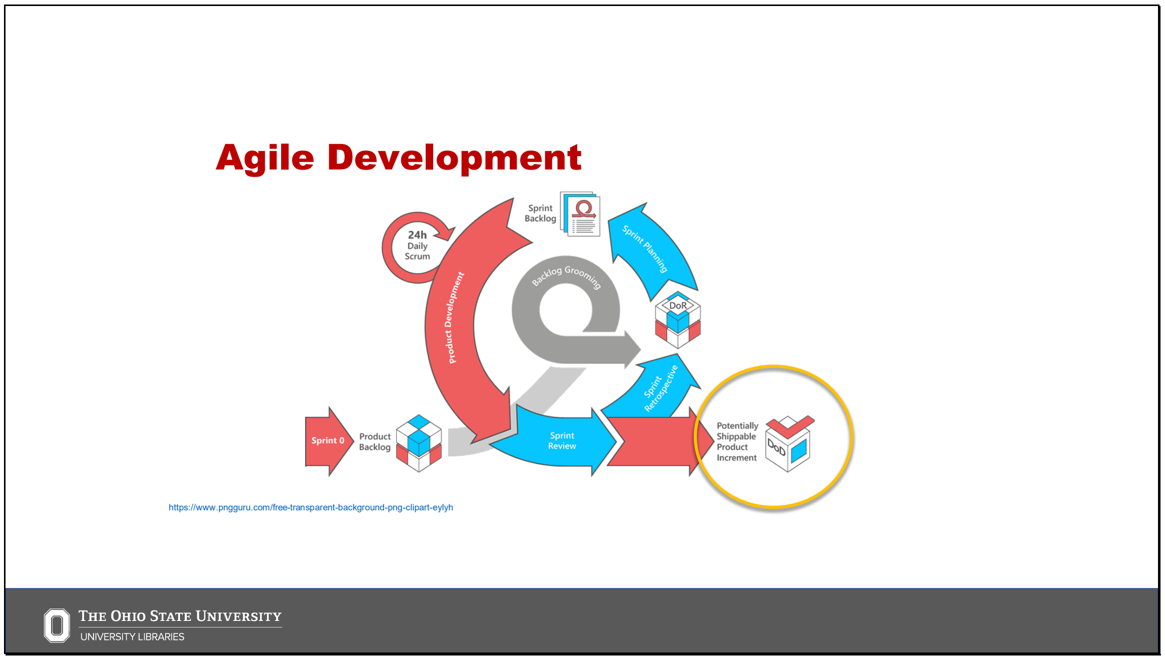
The Ohio State University
University Libraries

First, what is DevOps? There's quite a bit of misunderstanding around DevOps, and if you Google DevOps, finding a useful, concrete and concise definition is challenging. Let's take a look at these common associations with DevOps:

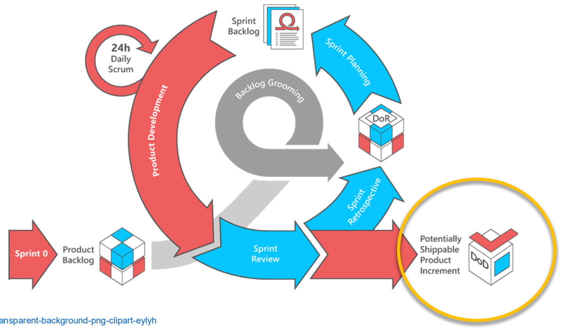
- You could be deploying code multiple times per week, or even day. For sure, frequent and fast delivery is a hallmark of DevOps.
- You might have someone on the team with the title of DevOps engineer. (I found quite a few posted on Indeed in Columbus.)
- Your systems could all be in the cloud.
- You could be using a DevOps toolchain, such as CI/CD, Puppet, Ansible, Docker, Kubernetes, etc.
- Your developers could be practicing an agile framework like Scrum or Kanban.

Are you DevOps? Perhaps but not necessarily. Because practices, though certainly valuable, don't automatically lead to a DevOps culture--which I believe is actually most important.

Slide 4



Agile Development

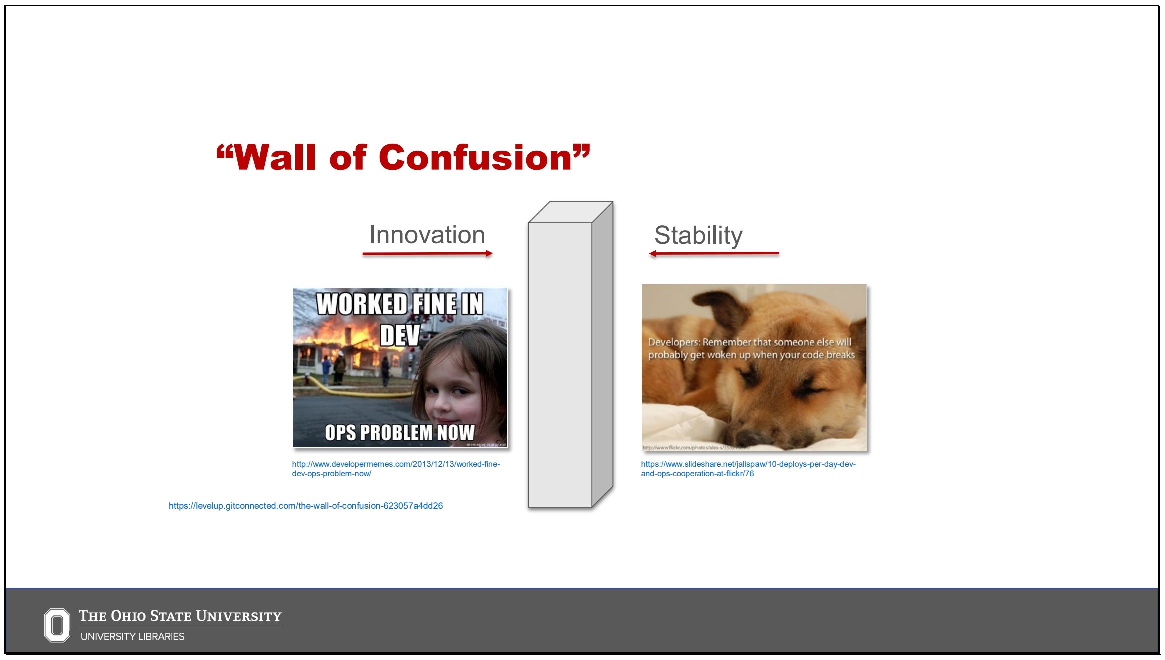


https://www.pngguru.com/free-transparent-background-png-clipart-eylyh

The Ohio State University
University Libraries

Let's drill down a bit on Agile software development, which I'm sure you are all familiar with. Here is a typical diagram of Scrum, one of the most popular Agile frameworks. Our team practices a modified version of Scrum plus Kanban in 2 week sprints. I don't want to spend too much time on Scrum itself but rather would like you to notice the output of this iterative cycle at the bottom right. --the "potentially shippable product increment". In Scrum, that increment should meet your team's D.O.D.-- "Definition of Done." Definition of Done could include things like…there are unit tests and there is sufficient test coverage; the code integrates successfully; and the product owner has approved for release. The choice of language here--"shippable product" and "done"--suggest—and what I have experienced and unfortunately, have encouraged--is that developers tend to consider their jobs to be finished when their code lands in production. Our code is deployed, we celebrate, let's leave! This attitude can be very problematic and a source of angst for some infrastructure and operations teams—and a major reason why DevOps came about. Agile simply didn't go far enough to include all members of the delivery team.

Slide 5



"Wall of Confusion"

Innovation

Stability

WORKED FINE IN DEV
OPS PROBLEM NOW
Developers: Remember that someone else will probably get woken up when your code breaks

http://www.developermemes.com/2013/12/13/worked-fine-dev-ops-problem-now/

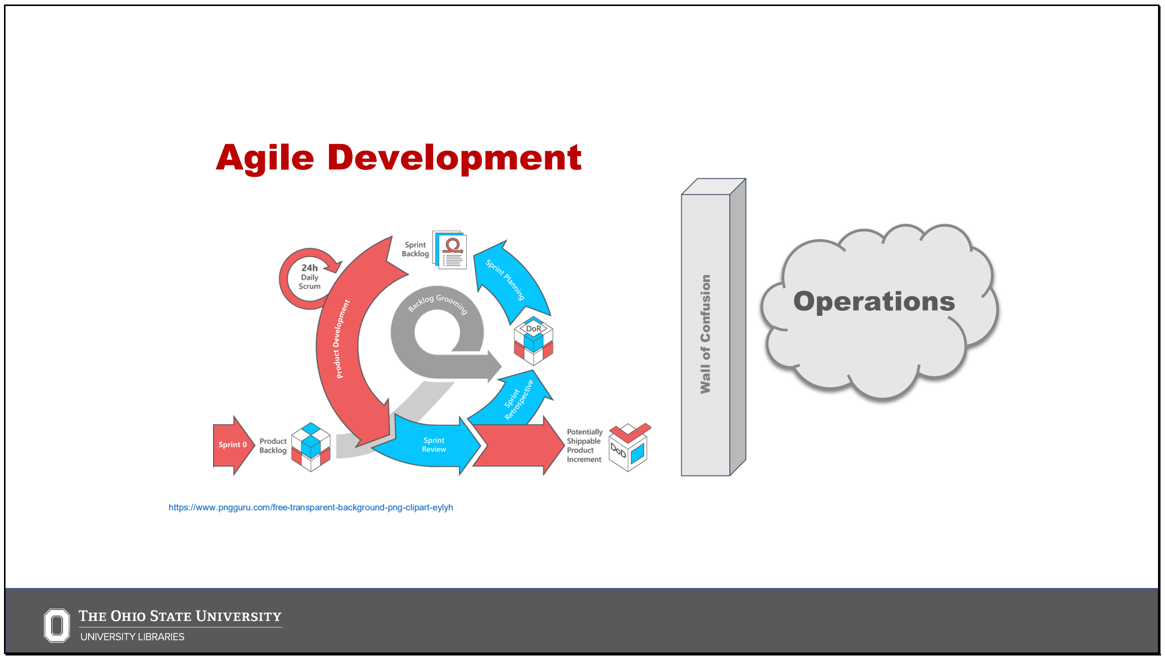
https://www.slideshare.net/jallspaw/10-deploys-per-day-dev-and-ops-cooperation-at-flickr/76

https://levelup.gitconnected.com/the-wall-of-confusion-623057a4dd26

THE OHIO STATE UNIVERSITY
UNIVERSITY LIBRARIES

There is a phrase in DevOps--the "wall of confusion"--between those who write the code and those who support the systems the code lives on. This wall would be expected if you think about how the different functions are incentivized and rewarded. Developers are rewarded for their creativity, innovation, taking risks, and visible output. Infrastructure and operations people are rewarded for keeping the systems up, stable, and secure, and their work is not visible—if it is visible, then there's probably a problem. So, the natural tendency of course would be to avoid risks. What we can end up with is tension and even competing goals between developers and sysadmins.

Slide 6



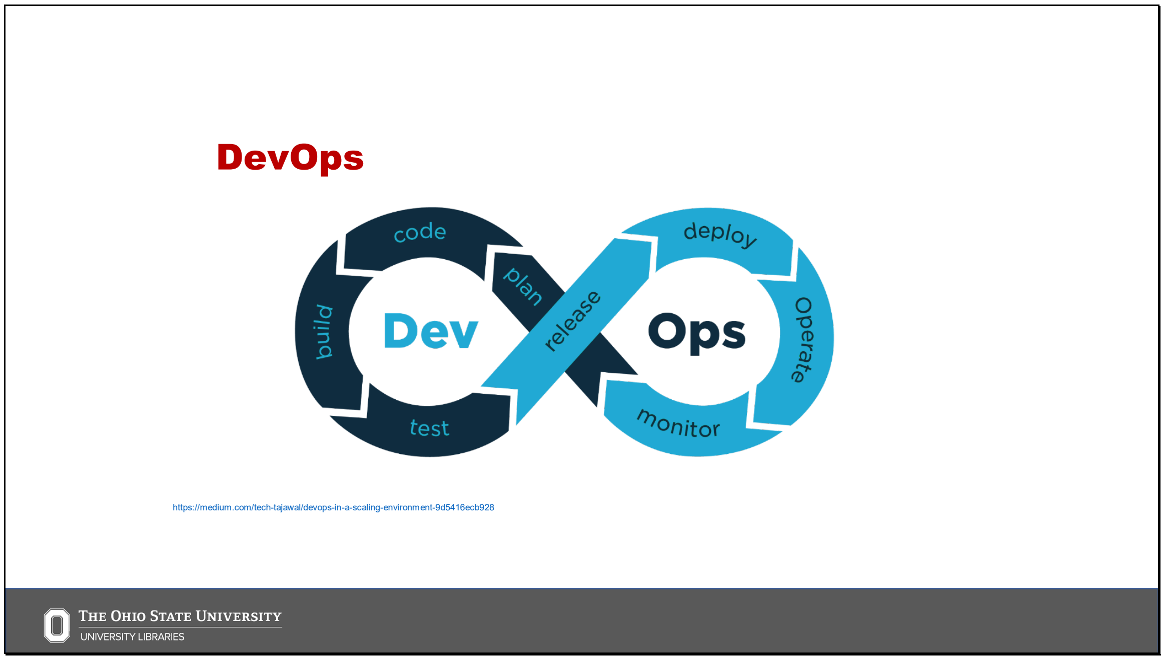
Agile Development

https://www.pngguru.com/free-transparent-background-png-clipart-eylyh

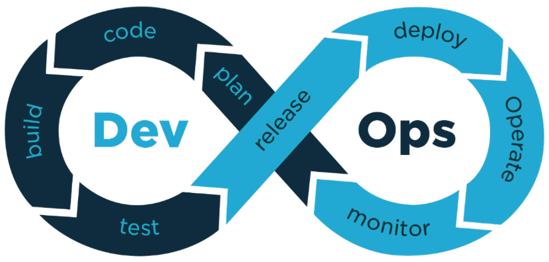
THE OHIO STATE UNIVERSITY
UNIVERSITY LIBRARIES

If we add the wall of confusion to our diagram of agile development, it might look like this--with the handoff of the product to the sysadmins and operations staff. It's a mystery as to what happens on either side of the wall. What if we eliminated that handoff?

Slide 7



DevOps

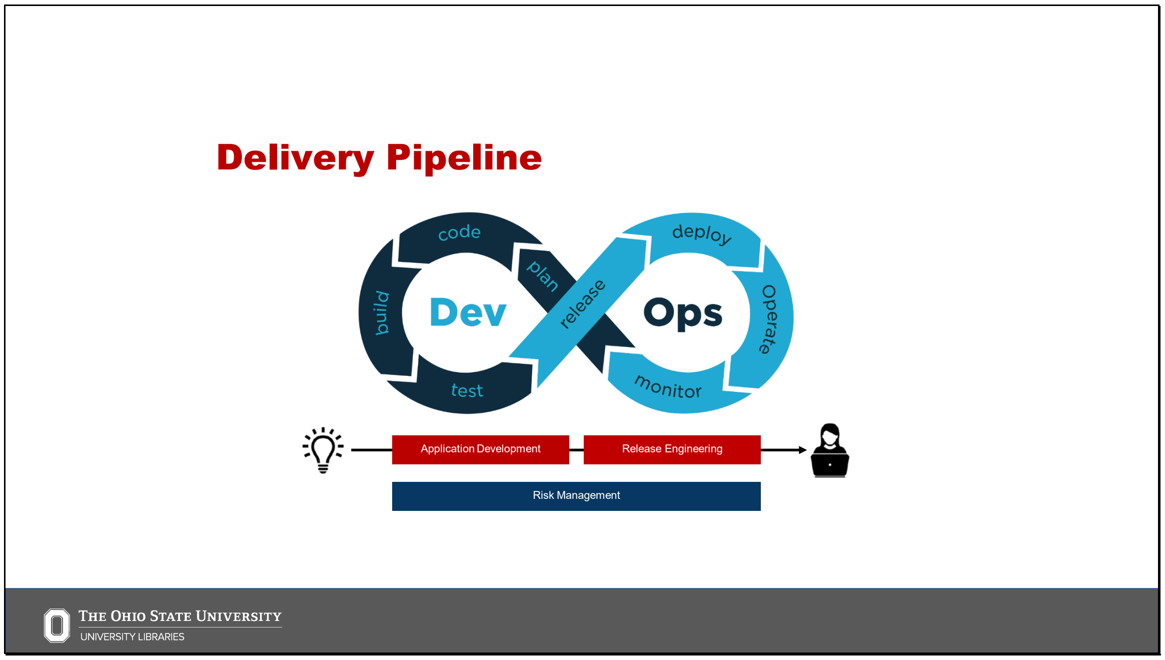


https://medium.com/tech-tajawal/devops-in-a-scaling-environment-9d5416ecb928

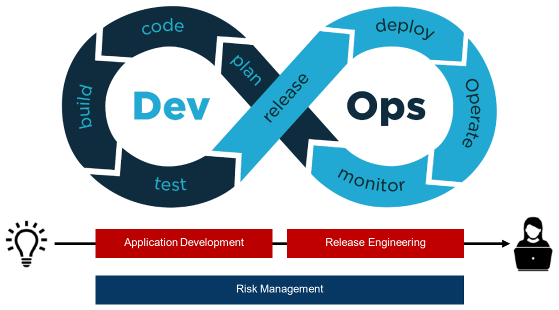
The Ohio State University
University Libraries

That’s exactly what DevOps intends to do--break down the wall of confusion by treating software delivery as a continuous, iterative pipeline through collaboration and feedback loops. If this sounds like Lean Thinking to you, it definitely is. Lean has heavily influenced DevOps which I will talk about in a second.

Slide 8



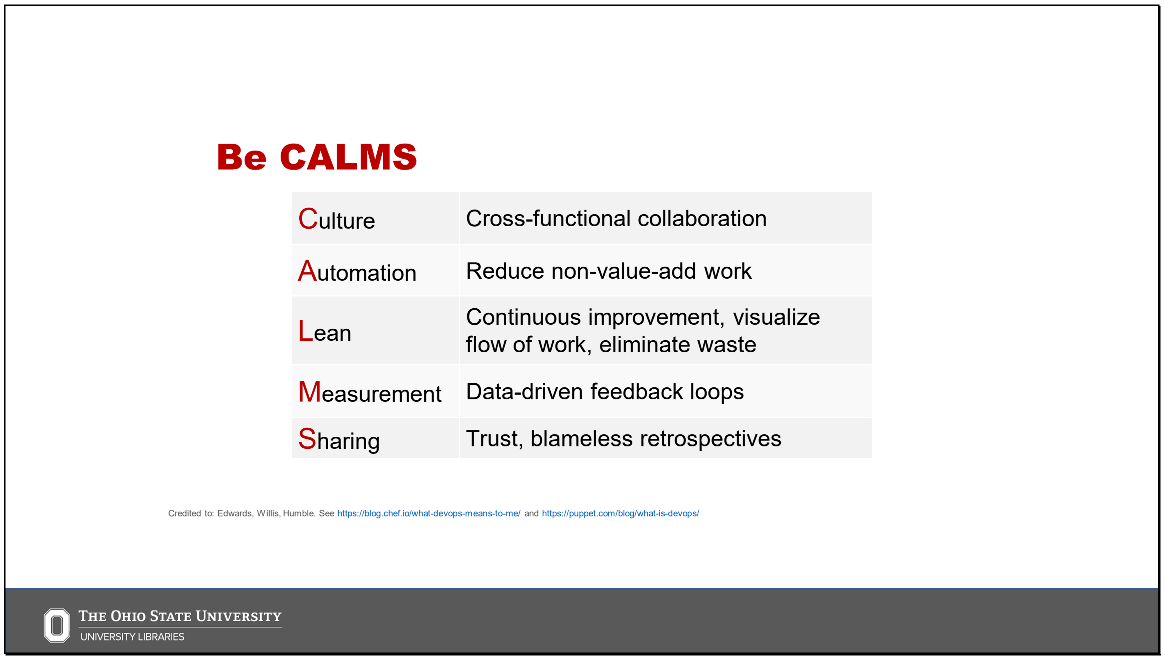
Delivery Pipeline



THE OHIO STATE UNIVERSITY
UNIVERSITY LIBRARIES

In our particular organization, our two systems administrators moved to my team to join the developers. The functions and expertise we need for end-to-end delivery of software are in one team of 9 people. This includes UX expertise and application security. We are extending Agile to include everyone, and we are intentionally putting the user's experience back into the equation. We are two years into our DevOps transformation, and I have learned that it is a journey and doesn't happen overnight by simply re-organizing teams. And that's because changing culture is the hard part.

Slide 9



Be CALMS

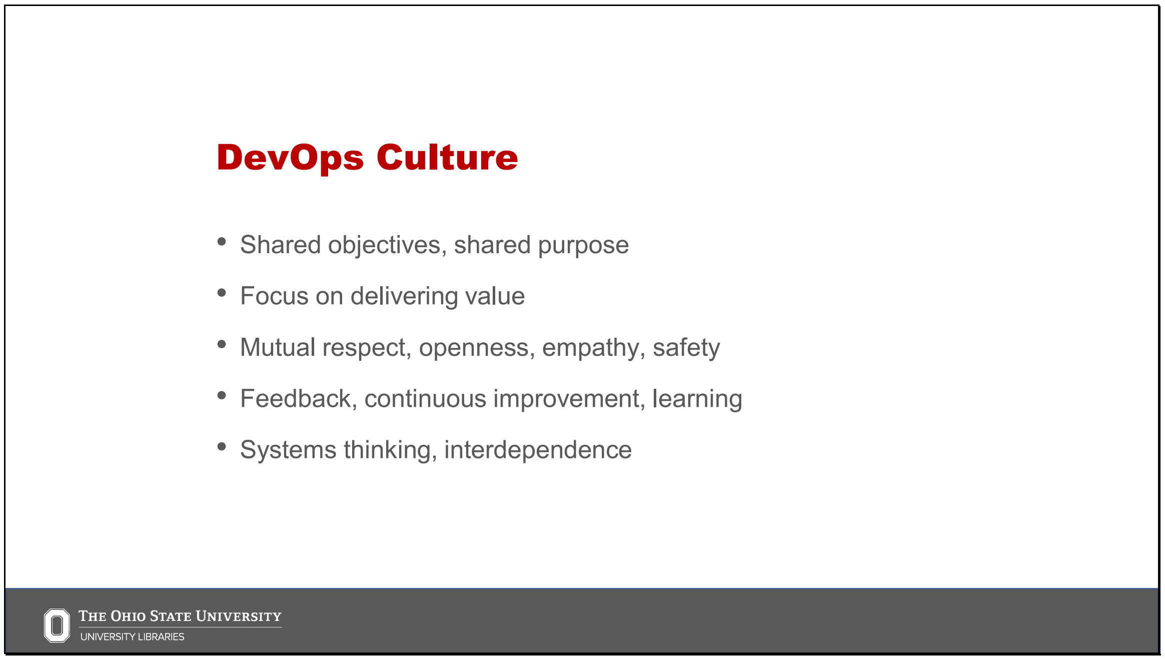
Culture	Cross-functional collaboration
Automation	Reduce non-value-add work
Lean	Continuous improvement, visualize flow of work, eliminate waste
Measurement	Data-driven feedback loops
Sharing	Trust, blameless retrospectives

Credited to: Edwards, Willis, Humble. See https://blog.chef.io/what-devops-means-to-me/ and https://puppet.com/blog/what-is-devops/

The Ohio State University
University Libraries

When it comes to culture, I've found the CALMS framework of DevOps to be useful. CALMS is an acronym for:

- Culture--cross functional collaboration
- Automation--reducing the work that doesn't add value to the user
- Lean--continuous improvement, visualizing flow of work, eliminating waste like handoffs
- Measurement--data-driven feedback loops
- Sharing--to decrease blame and increase trust



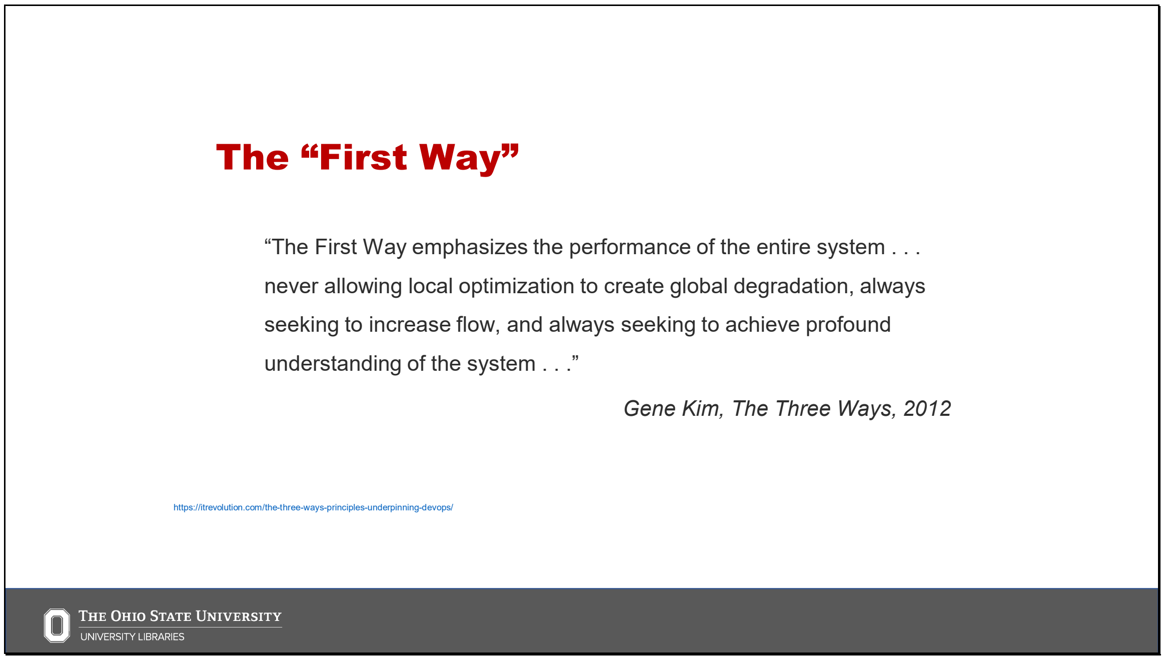
DevOps Culture

- Shared objectives, shared purpose
- Focus on delivering value
- Mutual respect, openness, empathy, safety
- Feedback, continuous improvement, learning
- Systems thinking, interdependence

THE OHIO STATE UNIVERSITY
UNIVERSITY LIBRARIES

To expand a bit on DevOps culture, it looks something like this:

- We have shared objectives—we all know WHY we are doing what we are doing and we are all committed to that purpose.
- We use technology and processes for what they are best at—automating and streamlining so we can spend time on creating value—what is meaningful to our patrons and users.
- We are all in this together—we respect each other's expertise and contributions toward these shared goals.
- We measure to learn and use introspection, retrospection, and data to improve.
- We appreciate that our own actions affect those around us—that we are interdependent in achieving our objectives.



The “First Way”

“The First Way emphasizes the performance of the entire system . . . never allowing local optimization to create global degradation, always seeking to increase flow, and always seeking to achieve profound understanding of the system . . .”

Gene Kim, The Three Ways, 2012

https://itrevolution.com/the-three-ways-principles-underpinning-devops/

THE OHIO STATE UNIVERSITY
UNIVERSITY LIBRARIES

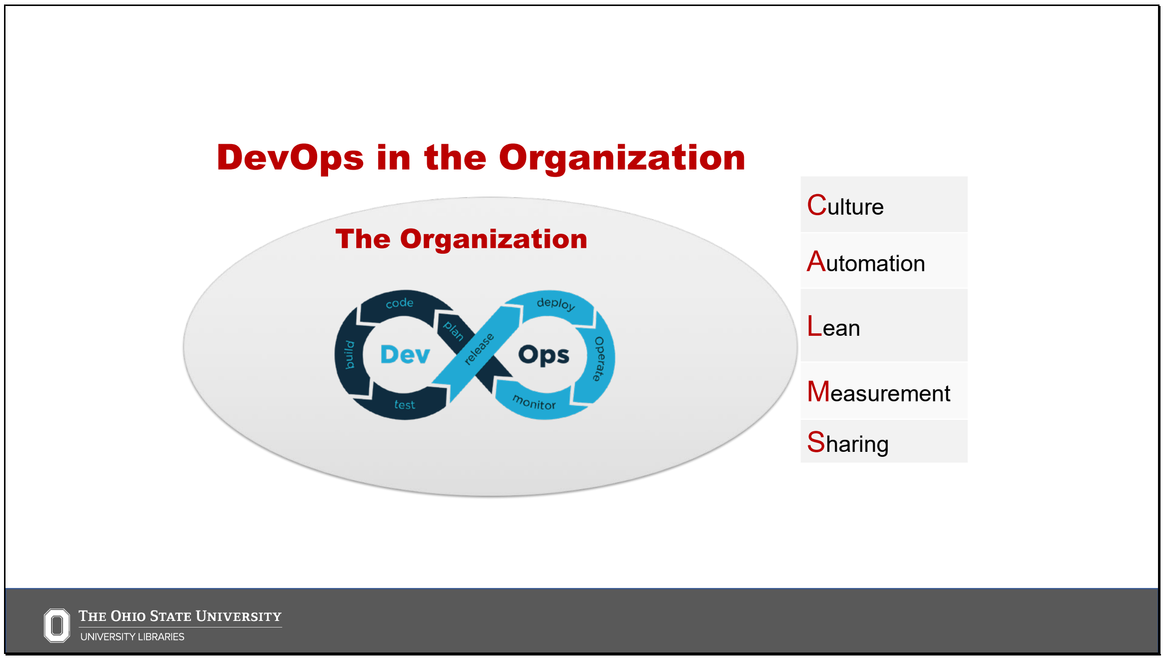
This is a tall order. You may be asking yourself--where do I even start? The “Three Ways” are at the basis of all DevOps principles and are described in *The Phoenix Project*, a book I would highly recommend. They are: 1) Flow, 2) Feedback and 3) Learning. The First Way is the place to start: visualizing the flow of work throughout the entire organization, and as Gene Kim says--a thought leader in the field--“never allowing local optimization to create global degradation, always seeking to increase flow, and always seeking to achieve profound understanding of the system.” Jennifer will now talk more about this.

DevOps in the Organization

The Ohio State University
University Libraries

We have a development team structured around and following DevOps practices, developing a DevOps culture.
As Beth said, there are organizational growth opportunities within her team (as with all teams migrating to DevOps practices) that they're working through.

But Applications Development is also embedded in a larger organization: University Libraries.
What about the rest of the library?



DevOps in the Organization

The Organization
code
plan
deploy
build
Dev
release
Ops
Operate
test
monitor

Culture
Automation
Lean
Measurement
Sharing

THE OHIO STATE UNIVERSITY
UNIVERSITY LIBRARIES

How do the Applications Development team's DevOps culture, the CALMS framework, function within a larger organization that, while appreciating many of these same values, doesn't know about and hasn't adopted the DevOps principles and methodology?

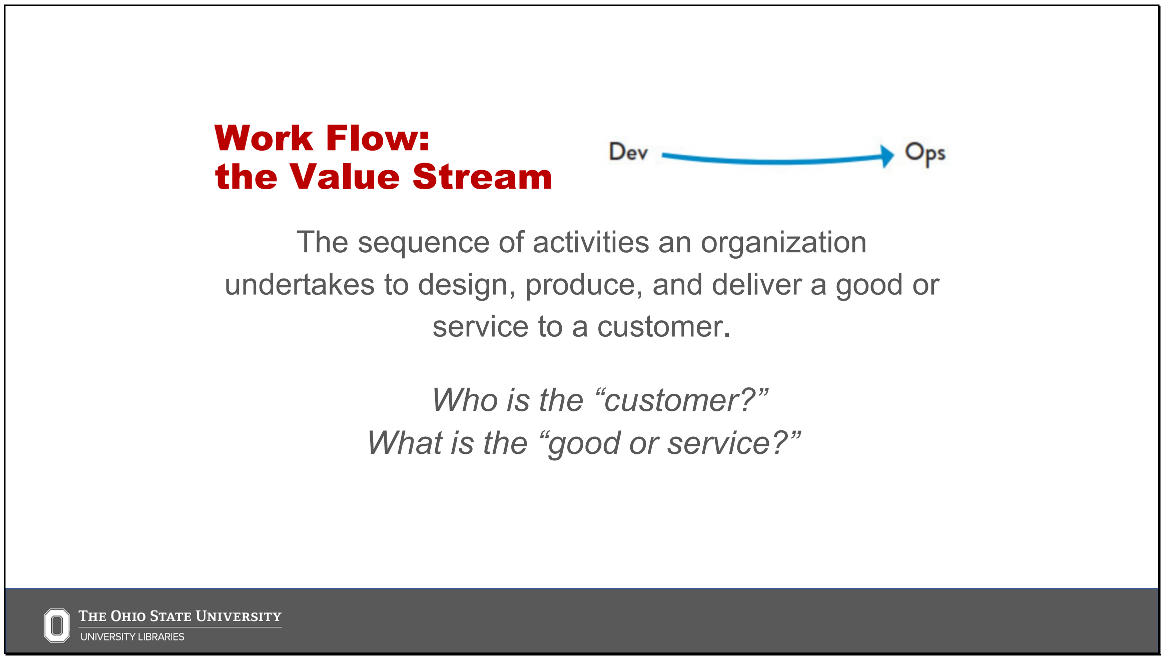
How can these same DevOps principles and practices help us to partner better across the organization, and also advance organizational learning?

I'm going to use a quick case study: Re-thinking our digital preservation workflow.

Without going into all the details about our preservation environment, let me just share that we do not have a common definition across the organization of exactly what digital preservation is, what our expected outcomes are or should be, when we should or shouldn't do it,
and how we all may contribute to doing digital preservation effectively.
Lots of assumptions are built into and structure our work, but if we don't know what those assumptions are we can't know if they're actually the right assumptions on which to be building a digital preservation program.

For the rest of this presentation I'm going to highlight a couple concepts and tools in DevOps methods that we can use to make our organizational work in this initiative effective.
I'm specifically going to focus on three concepts that I believe will start to move us toward higher levels of communication and trust across teams, and reveal assumptions that are frustrating our work: the Value Stream, Feedback and Feedforward Loops, and Organizational learning.

Slide 14



Work Flow: the Value Stream

Dev
Ops

The sequence of activities an organization undertakes to design, produce, and deliver a good or service to a customer.

Who is the "customer?"
What is the "good or service?"

The Ohio State University
University Libraries

Beth's team, along with the Digital Preservation Librarian, is leading the organization in a an exercise to visualize the flow of digital preservation work throughout the entire organization.

A key concept in Lean and DevOps is the Value Stream, which is: The sequence of activities an organization undertakes to design, produce, and deliver a good or service to a customer.

This language ("customer," "goods," "services") comes from the manufacturing sector, but here you can see how it maps well enough to our work in higher ed.

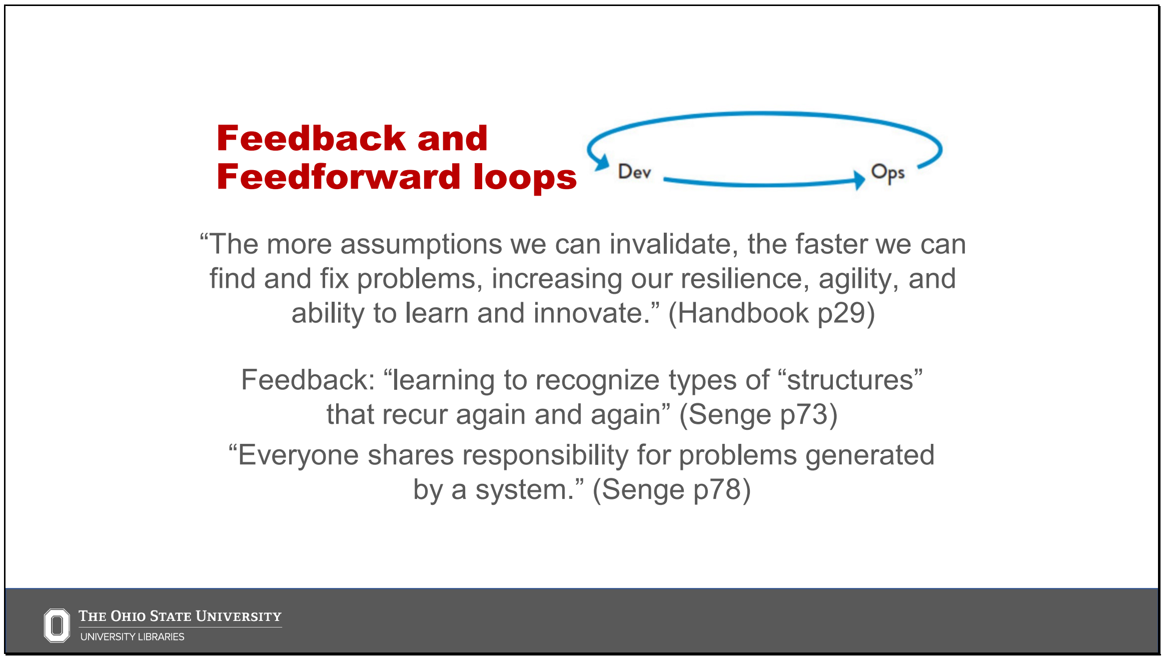
Beth's team and the Digital Preservation Librarian are working with all organizational partners to map our current digital preservation workflow from start to finish. That is called a value stream mapping. It will allow us to know all the inputs and outputs of our current system.

In order to design a value stream that gets you the results you want, you first need to know who is the customer or customers you're serving and what are the goods or services you should be offering.

It turns out those questions are harder to answer than I had originally thought when I came to Ohio State 3.5 years ago. Everyone contributing to the present value stream seems to have slightly different ideas about what the systems we've developed should actually be delivering and for whom. This is a people challenge, not a technology challenge.

In addition to understanding our current value stream, we also need to identify who we're building our systems for and why. That requires a different kind of investigation. If you're curious how they're doing it: they're using a method called SIPOC which identifies Suppliers, Inputs, Process, Outputs and Customers. Because of time constraints, we won't go into details here about SIPOC.

Slide 15



Feedback and Feedforward loops

"The more assumptions we can invalidate, the faster we can find and fix problems, increasing our resilience, agility, and ability to learn and innovate." (Handbook p29)

Feedback: "learning to recognize types of "structures" that recur again and again" (Senge p73)

"Everyone shares responsibility for problems generated by a system." (Senge p78)

THE OHIO STATE UNIVERSITY
UNIVERSITY LIBRARIES

While mapping our current workflow is important, in and of itself this won't get us to ideal state.

Rather, we need to increase the amount of information flowing through and about the system so we can understand what is and isn't working. We need to know what assumptions are underlying our systems so we can see if they are really the right drivers for the development and ongoing improvement of our workflow.

Feedback and feedforward loops allow us to iteratively test design and operating assumptions.

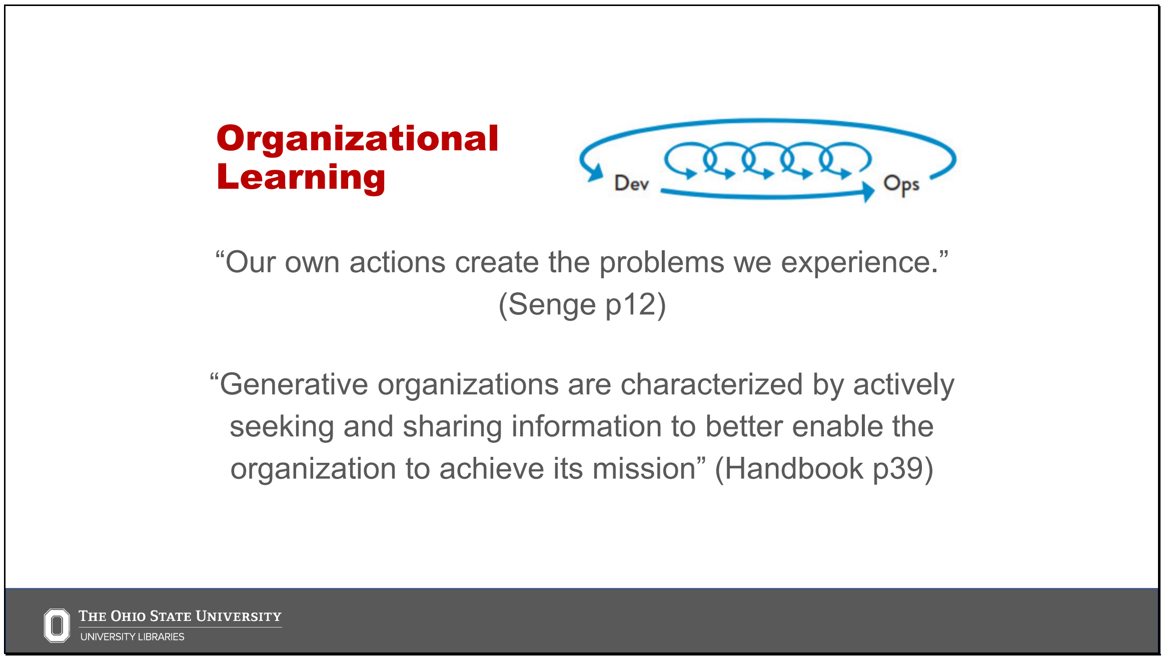
In the DevOps Handbook the authors state: "The more assumptions we can invalidate, the faster we can find and fix problems, increasing our resilience, agility, and ability to learn and innovate."

I'll also add that this will reduce human frustration with the system.

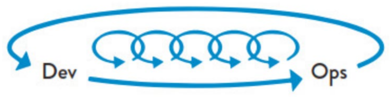
The idea of feedback loops comes from Peter Senge who explains in his book *The Fifth Discipline* that Feedback is "learning to recognize types of "structures" that recur again and again" (Senge p73) And he says that "Everyone shares responsibility for problems generated by a system."

Our ultimate goal is to have shared understandings and shared investment, and to build trust among people and also in the systems we use. Unless the system itself is built on eliciting this feedback, we won't get there.

Slide 16



Organizational Learning



"Our own actions create the problems we experience." (Senge p12)

"Generative organizations are characterized by actively seeking and sharing information to better enable the organization to achieve its mission" (Handbook p39)

The Ohio State University
University Libraries

This kind of systems thinking sees the collection of people and tools and workflows and assumptions and processes and documentation etc. etc. as all interconnected and all continually influencing and reinforcing (and also frustrating) each other.

As Senge says "Our own actions create the problems we experience."

If we are doing DevOps right -- within IT and in bringing these practices out into the organization -- we will be creating a learning organization that is adept at creating new knowledge and integrating it into *improved* work practices.

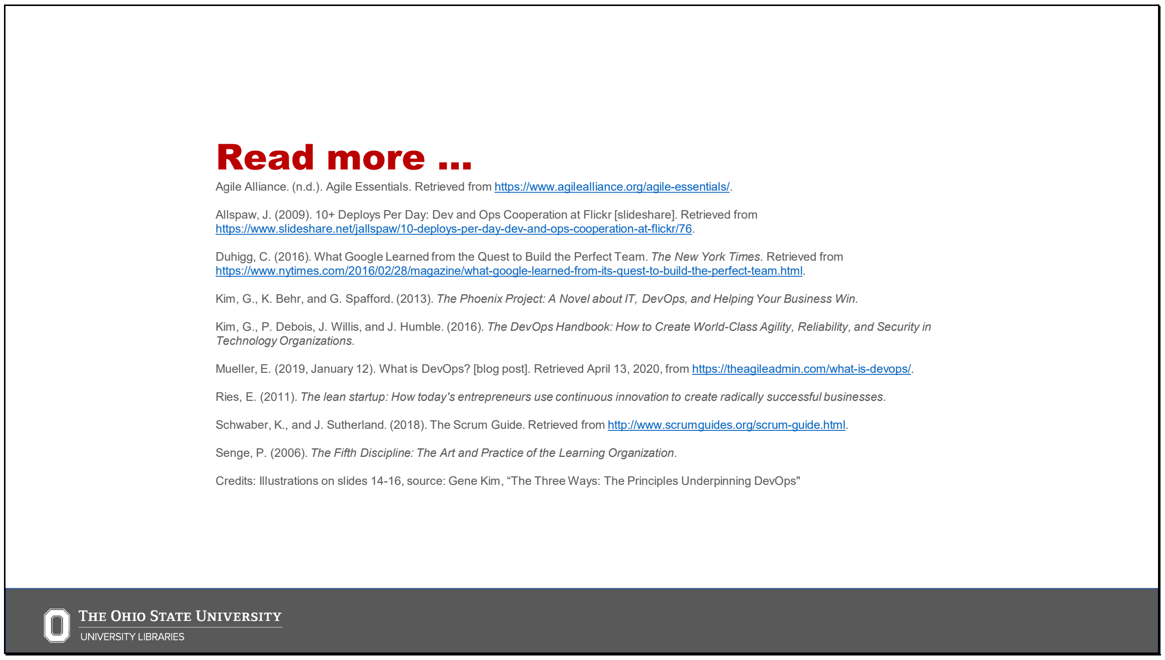
We must learn from our mistakes.
We must not blame but instead investigate problems and learn from them.
We do mid-stream reviews and retrospectives.
We elicit and test assumptions.
We understand that complex systems are complex, and so are the people who are also part of the systems, and that all the parts influence each other.
We need to get to a point where we can interrogate the assumptions underlying our work without people feeling attacked and defensive.

If we do this right -- with DevOps principles, some good tools and practices, and goodwill to learn from our experiences together --
we won't just be able to develop a better digital preservation workflow, we will also have developed more trust in our work, in our systems, and also in our mistakes and our learning.

In many ways the technical work is the easy part. From my perspective, and I believe Beth's too, it's the people work that is the most challenging *and* the most rewarding when we get it right.

Questions?

The Ohio State University
University Libraries



Read more …

Agile Alliance. (n.d.). Agile Essentials. Retrieved from https://www.agilealliance.org/agile-essentials/.

Allspaw, J. (2009). 10+ Deploys Per Day: Dev and Ops Cooperation at Flickr [slideshare]. Retrieved from https://www.slideshare.net/jallspaw/10-deploys-per-day-dev-and-ops-cooperation-at-flickr/76.

Duhigg, C. (2016). What Google Learned from the Quest to Build the Perfect Team. *The New York Times*. Retrieved from https://www.nytimes.com/2016/02/28/magazine/what-google-learned-from-its-quest-to-build-the-perfect-team.html.

Kim, G., K. Behr, and G. Spafford. (2013). *The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win.*

Kim, G., P. Debois, J. Willis, and J. Humble. (2016). *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations.*

Mueller, E. (2019, January 12). What is DevOps? [blog post]. Retrieved April 13, 2020, from https://theagileadmin.com/what-is-devops/.

Ries, E. (2011). *The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses.*

Schwaber, K., and J. Sutherland. (2018). The Scrum Guide. Retrieved from http://www.scrumguides.org/scrum-guide.html.

Senge, P. (2006). *The Fifth Discipline: The Art and Practice of the Learning Organization.*

Credits: Illustrations on slides 14-16, source: Gene Kim, "The Three Ways: The Principles Underpinning DevOps"

The Ohio State University
University Libraries