Cloud First Application Development

Lunchtime Learning

July 20, 2017

Bryan Hopkins, Lisa McBriar, Sam Donnelly
Cloud First AppDev Team

Not everyone is speaking, but this is everyone’s work

- Tim Bouffard, Application Architect
- Dave Curran, Sr. Business Systems Analyst
- Sam Donnelly, Sr. Application Developer
- Bryan Hopkins, Sr. IT Project Leader
- Anome Mammes, Sr. Application Developer
- Lisa McBriar, Sr. Business Systems Analyst
- Frank Nguyen, Sr. Application Developer
- Matt Schleindl, Application Architect
Today

- Stack vs. Framework
- Technology: Closer look at the cloud stack
  - How does each help?
- Processes: Agile and Test Driven Development
- Putting it in your hands
- What’s on your mind?
What goes into an app

✓ A user interface
  What people see
Behind the scenes

- User interface
- Business logic
- Authentication and security
- Persistence and data stores
- Integration points
- Testing
- Source repository
- Deployment pipeline
- Platform
- Process and SDLC
Cloud First Principles

- Choose simplicity, speed, and iterative improvement over up-front completeness and strict controls.
- Continuously iterate and re-evaluate decisions; nothing is beyond question.
- Be mindful of increasing technical debt.
- Choose automated solutions over manual ones, but respect the implications to manual process owners.
- Use tools and technologies “off the shelf” in secure ways and as the provider intended.

Credit: XKCD.com [https://xkcd.com/1742/](https://xkcd.com/1742/)
So how does that translate?

- Assembling a stack
- Prioritize
- Evaluate
- Integrate
- Automate
Our Process

Blank Slate

1. Select a few areas of focus
2. Choose disposable placeholders
3. Evaluate / Pilot
4. Release to Production
5. Retrospective
6. Update architecture and processes
Where do we stand?
# User Interface

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Full framework</td>
<td>• Responsive by default</td>
<td>• Open source</td>
</tr>
<tr>
<td>• Thousands of components</td>
<td>• Accessible components</td>
<td>• From Google and Twitter</td>
</tr>
<tr>
<td>• Readily available examples</td>
<td>• Cleaner, simple UIs</td>
<td>• Easy to find resources</td>
</tr>
<tr>
<td>• Data binding</td>
<td>• UX vision more achievable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Inexpensive</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Contractors already know it</td>
<td></td>
</tr>
</tbody>
</table>

- AngularJS
- Bootstrap
- Jetbrains Webstorm

University of Pennsylvania ISC: Cloud First Program
# Business Logic

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
<tbody>
<tr>
<td>▪ Full framework</td>
<td>▪ <strong>Self documenting APIs</strong></td>
<td>▪ Open source</td>
</tr>
<tr>
<td>▪ Thousands of packages</td>
<td>▪ Automatically generated test interfaces</td>
<td>▪ Tremendous campus presence</td>
</tr>
<tr>
<td>▪ Readily available examples</td>
<td></td>
<td>▪ Very easy to learn</td>
</tr>
<tr>
<td>▪ Simple and powerful: few lines of code</td>
<td>▪</td>
<td>▪ Easy to find resources</td>
</tr>
<tr>
<td>▪ Django-aware code completion</td>
<td>▪</td>
<td></td>
</tr>
<tr>
<td>▪ Great Django Testing integration</td>
<td>▪</td>
<td><strong>Inexpensive with a free tier</strong></td>
</tr>
</tbody>
</table>

---

*Image of Django, Python, and PyCharm logos.*

**University of Pennsylvania ISC: Cloud First**

**ISC's Cloud First Program**
# Authentication and Security

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Thousands of packages</td>
<td>• No effort PennKey login with Shibboleth</td>
<td>• Open source</td>
</tr>
<tr>
<td>• Readily available examples</td>
<td></td>
<td>• Easy to find resources</td>
</tr>
<tr>
<td>• Incredible I/O performance</td>
<td>• Inexpensive with a free tier</td>
<td></td>
</tr>
<tr>
<td>• Full stack</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Framework-aware code complete</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Fullstack debugging with AngularJS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Data Store

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Oracle-like features</td>
<td>• Drop-in replacement for Oracle</td>
<td>• Open source, ethical, and Free</td>
</tr>
<tr>
<td>• That actually work!</td>
<td>• Works great with Talend</td>
<td>• Strong local presence</td>
</tr>
<tr>
<td>• Highly performant</td>
<td>• DB of choice for warehousing</td>
<td></td>
</tr>
<tr>
<td>• Relational AND NoSQL</td>
<td>• Very easy to use</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Don’t need to know SQL</td>
<td>• Inexpensive</td>
</tr>
<tr>
<td></td>
<td>• Works with every RDBMS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Powerful autocompletion</td>
<td></td>
</tr>
</tbody>
</table>
## Integration

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Fully supported SDK in every language</td>
<td>• Powerful and flexible security models</td>
<td>• Incredibly cheap</td>
</tr>
<tr>
<td>• Easy to follow examples</td>
<td>• API Keys and Usage plans</td>
<td>• Fully managed</td>
</tr>
<tr>
<td>• Framework and platform agnostic</td>
<td>• Configurable “stages”</td>
<td>• Serverless</td>
</tr>
<tr>
<td>• Built in DDoS protection</td>
<td></td>
<td>• Taggable for <strong>cost transparency</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Service orientation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Fully supported SDK in every language</td>
<td>• Works with Mule</td>
<td>• Inexpensive</td>
</tr>
<tr>
<td>• Failure tolerant</td>
<td></td>
<td>• Taggable for <strong>cost transparency</strong></td>
</tr>
<tr>
<td>• Limitless scalability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**API Gateway**

**Amazon SQS**
Closer Look: Microservices and SOA

- No longer self-contained
- Think service areas
  - DAR
  - Finance
  - Student systems
  - HR
  - Research
- Composed of small pieces
  - Reusable
  - Built that way
  - Deployed that way

Looser Coupling, More Flexible/Portable, More Complex Outer Architecture

Monolithic App
- WAR/EAR
- App Server
- Data Store

Coarse-Grained Services
- Service Domain
- App Server
- Data Store

Microservices
- Service A
- Service B
- Service C
- Service D
- Service E
- Data Store

Tighter Coupling, Less Flexible/Portable, Less Complex Outer Architecture
Closer Look: Microservices and SOA

- No longer self-contained
- Think service areas
  - DAR
  - Finance
  - Student systems
  - HR
  - Research
- Composed of small pieces
  - Reusable
  - Built that way
  - Deployed that way
## Testing

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Confidence in the spec</td>
<td>• Easy to create and understand</td>
<td>• Bugs caught early take 10x less time</td>
</tr>
<tr>
<td>• Built for AngularJS</td>
<td>• Fully automated regression tests</td>
<td></td>
</tr>
<tr>
<td>• Easy stubbing and mocking with Sinon</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Straightforward</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Like Junit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Easy to test WebServices: no setup</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Source Repository

For Developers...
- Offline work
- Easy branches and merging
- Webhooks for CI/CD integration

For Analysts...
- Create documentation developers actually use
- Intuitive web interface to browse files

For Project Managers...
- Open source
- Universal adoption (git)
# Deployment Pipeline

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Automate with Python</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Developers’ Swiss Army knife</td>
<td>• More frequent deployment</td>
<td></td>
</tr>
<tr>
<td>• Automated tests run on each build</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Slack Integration</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Jenkins*
## Platform

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Serverless</td>
<td>• Incredibly cheap</td>
<td>• Can be sized and scaled to order</td>
</tr>
<tr>
<td>• Only worry about code</td>
<td>• Nothing to patch</td>
<td>• Taggable for cost transparency</td>
</tr>
<tr>
<td>• Provisioned as code</td>
<td>• Massively scalable</td>
<td></td>
</tr>
<tr>
<td>• Isolated microservices</td>
<td>• Taggable for cost transparency</td>
<td></td>
</tr>
<tr>
<td>• Easy DRF integration with Zappa</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Can run any docker image</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Service isolation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Blue-green deployments</td>
<td></td>
</tr>
</tbody>
</table>
# Platform

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
</table>
| • Fully Managed Service  
  • Provisioned as code | • Access like any other database | • Fully managed  
  • Nothing to patch  
  • Scalable  
  • Taggable for cost transparency |

Amazon RDS
Closer Look: Infrastructure as Code

- Everything about your app is just code
- Creating it defines and documents it
- No patching. Delete and re-create.
- No waiting! No manual action! Right the first time!
Closer Look: Infrastructure as Code

```
"MyDB" : {
  "Type" : "AWS::RDS::DBInstance",
  "Properties" : {
    "DBSecurityGroups" : [
      {"Ref" : "MyDbSecurityByEC2SecurityGroup"}, {"Ref" : "MyDbSecurityByCIDRIPGroup"} ],
    "AllocatedStorage" : "5",
    "DBInstanceClass" : "db.m1.small",
    "Engine" : "MySQL",
    "MasterUsername" : "MyName",
    "MasterUserPassword" : "MyPassword"
  },
  "DeletionPolicy" : "Snapshot"
}
```
How does the tech fit together?
What about process?
## Process and SDLC

<table>
<thead>
<tr>
<th>For Developers...</th>
<th>For Analysts...</th>
<th>For Project Managers...</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Familiar tool</td>
<td>• Plugin store</td>
<td>• Transparency</td>
</tr>
<tr>
<td>• Easy to use</td>
<td>• Dependency tracking</td>
<td>• Reports and progress-at-a-glance</td>
</tr>
<tr>
<td>• Highly collaborative</td>
<td>• Drag and drop prioritization</td>
<td>• Collaborate externally</td>
</tr>
<tr>
<td>• Focus on only a few details at a time</td>
<td>• Focus is on “Yes”</td>
<td>• Speed and time-to-value</td>
</tr>
<tr>
<td>• Frequent feedback and adjustment</td>
<td>• Rely on domain knowledge and organization</td>
<td>• Responsive to changes in priorities</td>
</tr>
<tr>
<td></td>
<td>• No document writing</td>
<td></td>
</tr>
</tbody>
</table>
Closer Look: Cloud First Agile

Terminology

- **Backlog**
  Sort of list of everything the clients want. Things at the top are better defined than things at the bottom.

- **Sprint**
  Period of time where the team focuses on implementing the clients’ top priorities.

- **User story**
  The basic unit of work capturing a feature or requirement from the end user’s perspective.

- **Epic**
  A collection of User Stories that together define a significant feature or business process.
Closer Look: Cloud First Agile

Create backlog
- Create VERY high level story list

Prioritize and Define
- Pick top 1-2 stories
- Feature files
- Screens

Build and Iterate
- Technical subtasks
- Playbook
- Unit tests
- Daily Standup

Test and Deploy
- E2E Tests
- Able to launch?
- MVP?

University of Pennsylvania ISC: Cloud First Program
Closer Look: Cloud First Agile

Backlog drag and drop example

https://penn-isc-es.atlassian.net/secure/RapidBoard.jspa?projectKey=APND&rapidView=21&view=planning
Closer Look: Cloud First Agile

User Story Playbook
✓ Feature File
✓ Screen Mockups
✓ Architecture
  q Swagger
  q Diagram
  q Review with partners
✓ Unit Tests
✓ E2E Tests
✓ Code Review
✓ Accessibility Testing

- The checklist will be used by the developer assigned to the story to review standards agreed upon and keep track of progress.
- 🔄 = In progress
- 🟢 = Done/complete
- 🔴 = Skip based on dev and architect agreement
- Once the developer is ready for 5.6 Code Review and 5.7 Dev team UNIT load testing, the jira can be reassigned to whomever will help with these.
- For 5.8. AFTER it has been deployed and checked in AWS TEST, the jira can be assigned to the analyst for functional testing.

<table>
<thead>
<tr>
<th></th>
<th>DEVELOP ONCE A SPRINT BEGINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.</td>
<td>As part of Story Prep (4.0) in a previous Sprint, the devs and architects will review each story for questions and make clarifications. This step is as the Sprint begins, to make sure all questions have been answered. STORY JIRA and dev subtasks are moved to ‘Selected for Development’ column at this time.</td>
</tr>
<tr>
<td>5.</td>
<td>Test Driven Development (TDD) &amp; Unit Testing Guidelines</td>
</tr>
</tbody>
</table>

Developer is responsible for defining UNIT tests to be created for each module, these will be verified through code review (see...
Closer Look: Cloud First Agile

Feature Files

- Easy to read
- Used to create Selenium tests

```text
#UX_1_J#
Scenario Outline:
Once a user has registered 5 devices, both input fields are disabled, the register button is disabled and a message displays about not being able to have more than 5 devices registered.

Given user is on the Penn Device Portal Personal Page
  When User has registered 5 devices
  Then input fields are disabled
  And Register button is disabled
  And a message indicating 5 devices are registered
```
Boards!
## Boards: Example

### GIVING Sprint 5

<table>
<thead>
<tr>
<th>To Do</th>
<th>Selected for Development</th>
<th>In Progress</th>
<th>Testing / Feedback</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORS error loading assets on fund search page</td>
<td>APIs: Cybersource, APIs GIVING-60, GIVING-61</td>
<td>Capture items for deployment playback: None ADR-228, GIVING-46, VDORDER-339</td>
<td>Create referential integrity (foreign key) constraints in reference tables: None None</td>
<td>Giving Form SETUP: None None</td>
</tr>
<tr>
<td>None None GIVING-124</td>
<td></td>
<td></td>
<td></td>
<td>Giving FORM: None None GIVING-63, GIVING-66</td>
</tr>
<tr>
<td>Need S3 creds for giving test and prod image folders</td>
<td>Need PostgreSQL service account and creds for aws test and prod</td>
<td></td>
<td></td>
<td>PREP in form fund search page SETUP: None None</td>
</tr>
<tr>
<td>None None</td>
<td></td>
<td></td>
<td></td>
<td>Giving FORM: None None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Giving FORM: None None GIVING-85</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>In form fund search page SETUP: None None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Giving FORM: None None GIVING-94</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PREP UX-2 Giving In-Form Fund Selector</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Giving FORM</td>
</tr>
</tbody>
</table>

---

University of Pennsylvania ISC: Cloud First
Boards

- Two Modes
  - Scrum Board: Active development in sprints
  - Kanban Board: Expediting support work
  - The only real difference is timeframe

- Transparency
  - Team members see what each other are up to
  - Stakeholders can see what our priorities are – and give input
  - Support the standup meeting and client meetings
UX Design: My Journey
UX Design

What is UX Design?

“The process of enhancing user satisfaction by improving the usability, accessibility, and pleasure provided in the interaction between the user and the product.” --Wikipedia

https://uxmag.com/articles/explaining-ux-design-to-your-team
UX Design

Why is UX Design important?

- Drives client and team conversations about USABILITY; most importantly, accessibility and responsiveness

- Enables early design of UI components meeting Web Compliance Accessibility Guidelines (WCAG) 2.0 Level AA
Accessibility design principle example

Don’t use color alone to convey information

While color can be useful to convey information, color should not be the only way information is conveyed. When using color to differentiate elements, also provide additional identification that does not rely on color perception. For example, use an asterisk in addition to color to indicate required form fields, and use labels to distinguish areas on graphs.

Example: Using color to convey meaning

- Color only

<table>
<thead>
<tr>
<th>Required fields are in red</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Email</td>
</tr>
</tbody>
</table>

Example: Refer to something using color alone

- Color only

<table>
<thead>
<tr>
<th>Which is the right-angled triangle?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Green</td>
</tr>
<tr>
<td>Blue</td>
</tr>
<tr>
<td>Red</td>
</tr>
<tr>
<td>Yellow</td>
</tr>
<tr>
<td>Don’t know</td>
</tr>
</tbody>
</table>

- Color and symbol

<table>
<thead>
<tr>
<th>Required fields are in red and marked with an *</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Email *</td>
</tr>
</tbody>
</table>

- Color and number

<table>
<thead>
<tr>
<th>Which is the right-angled triangle?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Green (1)</td>
</tr>
<tr>
<td>Blue (2)</td>
</tr>
<tr>
<td>Red (3)</td>
</tr>
<tr>
<td>Yellow (4)</td>
</tr>
<tr>
<td>Don’t know</td>
</tr>
</tbody>
</table>

https://www.w3.org/WAI/gettingstarted/tips/designing.html
Responsiveness design principle example

Create designs for different viewport sizes

Consider how page information is presented in different sized viewports, such as mobile phones or zoomed browser windows. Position and presentation of main elements, such as header and navigation can be changed to make best use of the space. Ensure that text size and line width are set to maximize readability and legibility.

Example: Content and navigation adapt to smaller mobile screen

Display in a wide window with small text uses multiple columns for primary content, visible navigation options, and visible secondary information.

Display in a narrow window, such as a mobile phone, or with large text uses single column for primary content, navigation options are revealed using an icon, and secondary information is also revealed via icon.

https://www.w3.org/WAI/gettingstarted/tips/designing.html
UX Design

- **Why is UX design helpful?**
  - Something to show clients QUICKLY
  - Something for clients to show their users
  - Something to easily change; frequently

- **How is our UX design process evolving?**
  - Torchbox wireframes & components (VDI)
  - Torchbox wireframes & prototype app (GIVING)
  - Penn Mockups (VDI, GIVING, PennDevice)
UX Design:

Torchbox Wireframe

Penn Mockup (Lucidchart)

Virtual Desktop (VDI)

Screenshot from Torchbox component demo
UX Design:

Torchbox Wireframe

Screenshot from Torchbox prototype app

GIVING

Penn Mockup (Lucidchart)

BACKGROUND Image:
Display the Default Spotlight Image, if there are not configured at the “program” or “program/fund” levels. A custom spotlight image configured at the “program/fund” level would display that image in place of the Spotlight Images: 1600x960 (Desktop background) / 680x440 (Mobile).

Desktop Filename: Hibernate_id-school-desktop.php
Mobile filename: Hibernate_id-school-mobile.php

Gift Cart:
See Lucid Chart - Giving Cart/Basket

PERSONAL INFORMATION:
Name Fields:
Prefix - Drop down will be populated from the CONTROL_TABLE. The data elements for the have value of “PREFIX_NAME” in the PARAMs.

First Name (required)
Middle Name
Torchbox Annotation
Bekah Evans
Middle name field required
UX Design:

- **Penn Device**
  - **Users:** Students, Faculty, Staff
  - **Call to actions**
  - **Lucidchart shape templates**
  - **Collaborative editing with clients**
  - **Client-User reviews (screenshots or pdfs)**
  - **Torchbox feedback**
  - **Styleguide**
  - **Easy integration w JIRA**
UX Design: Responsiveness

- Develop
- Check page
- Adjust design
- Repeat...

Firefox >
Web Developer >
Responsive Design Mode
UX Design: Accessibility

- Develop
- Check page
- Adjust design
- Repeat...

Chrome WAVE Plugin
Iterating with confidence:
Test Driven Development Demo
Putting it all Together
We are being judged...

- Increasingly for our...
  - Speed!
  - Accessibility (usability for those with disabilities)
  - Responsiveness (usability on a wide range of devices)
  - Cost transparency

- Less for our...
  - Risk profile
  - Meticulous custom services

Less is not none!
A shift in risk, not a magic bullet

AWS S3 Outage – 2/28/2017

• Bad day for the cloud
• Developers in the dark
• Apps couldn’t restart
• Images didn’t load

Data Center Outages – 6/21, 28/2017

• Good day for the cloud
• Developers unaffected (but it was dark)
• Apps ran and could be built (thanks to Shibboleth distribution)
• Campus integrations degraded gracefully

Credit: The Register
https://www.theregister.co.uk/2017/03/01/aws_s3_outage/

Credit: Matt’s iPhone
Summarizing the Upside
We want to let you build this

Consistent
Accessible
Responsive
And never again see this:

Problem with looping trying to login, are cookies enabled in the browser, and is the URL protected by the web server (ask Chris or Bryan to add your app to the web server config)?
Because you should never feel like this:
... you should feel like this!
How to start?

- Many already have, that’s great!
- Think of it as learning to learn, not learning a framework or frameworks
- Almost everything you’ve seen here is open source and free. Try it out!
  - Python / Django REST
  - PyCharm Community Edition
  - AngularJS
  - Bootstrap
  - NodeJS
  - PostgreSQL
  - Gitlab

- [https://www.isc.upenn.edu/cloud-developer-curriculum](https://www.isc.upenn.edu/cloud-developer-curriculum)
- Share and ask questions in #cloud-first on Slack
  - [http://upenn-isc.slack.com](http://upenn-isc.slack.com)
- Join Dev-SIG on Slack (#dev-sig) and Gitlab
  - [http://upenn-it.slack.com](http://upenn-it.slack.com)
  - [https://gitlab.com/upenn-dev-sig](https://gitlab.com/upenn-dev-sig)
What’s next…

- Lowering the barrier of entry for API Services
- Improving test automation
- Improving build automation
- Improving deployment automation
- Improving agile processes

… and whatever’s on your mind. Thank you!
Special Thanks and Credits

James Choate
Justin Ettore
Katie Faucett
Dane Fetterman
Tiffany Hanulec
Teresa Leo
Janet Lind
John Mulhern III
Janice Panesar
Isobel Thompson
Namitha Venkatesh
Mathias Wegner