Cloud First Application Development

Lunchtime Learning

December 8, 2016
ISC’s Cloud First Program

Core Team
- Application Design & Tools
- Org Transformation
- Architecture, Infrastructure, & Migration
- Integration
- Communications & Engagement

Client Engagement
Funding & Finance
Contracts & Procurement
Security
IAM
Integration: What (review)

- Two primary flavors
  - Authentication and authorization
    - PennKey/SAML
    - PennGroups
    - Penn Community
  - Application-specific data
    - Data into applications (think student enrollment data into Canvas, or employee job data into KnowledgeLink)
    - Data out of applications, to other applications or to reporting and analytics environments like the Data Warehouse
Integration: Current State

- Primarily point-to-point integration between SaaS, hosted, on-prem solutions and/or data warehouse
- Variety of technical approaches
  - PL/SQL / Oracle tool approaches
  - Java / FAST / other development environments
  - Mule / ESB
  - Penn Community APIs
- In most cases, dependent on highly-skilled developers
Integration: Goals

- **Toolset** that will accommodate
  - Use by business analysts / non-programmers
  - Support Warehouse needs
  - Support SaaS implementations with support for data to/from on-prem sources
  - Support Penn-developed applications
  - Support intra-application integrations (B2B / SaaS-to-SaaS)

- Processes and documentation on standard usage for the tools we acquire

- Integration Service supporting needs across ISC and the University’s schools and centers
Integration: work to date

- Lessons learned (nothing surprising)
  - Most vendors of SaaS solutions we use don’t have great API/web service platforms

- RFI/RFP content for Penn-wide consumption

- ETL tool acquisition effort in progress
  - Currently working on live proof-of-concept use cases with two vendors
  - Both vendor solutions meet multiple needs which may provide integration solution(s) for cloud-based services

- Research into other integration directions
  - iPaaS, mPaaS, xXaas...
  - This is a rapidly evolving product space; evaluation pending available expertise/resources
Today

- Why change so much?
- What’s our project?
- Updates and demos
  - Technology Stack: Matt Schleindl
  - Behavior and Test Driven Development: Sam Donnelly
  - Agile Development: Lisa McBriar
What goes into an app

- A user interface
  What people see
But there’s a lot more lurking...

...Technical Debt
What goes into an app

- Business logic
  Actual value
What goes into an app

- AuthN and AuthZ
- Security
What goes into an app

- Persistence and data stores
  Making it all matter
What goes into an app

✓ Integration points
  Playing well with others
What goes into an app

✓ Testing
  Proof it’s doing what it should be
What goes into an app

- A source repository
- Storage for code and config
What goes into an app

✓ Deployment pipeline
  Getting built and available
What goes into an app

- A platform
  Somewhere to run and scale
That’s a lot of stuff

- User interface
- Business logic
- Authentication and security
- Persistence and data stores
- Integration points
- Testing
- Source repository
- Deployment pipeline
- Platform
Until now...

- We’ve built these ourselves
  - And VERY successfully so!
  - Like many, many others
  - With full control
  - Minimal short-term risk
  - One language
  - Unique ISC terminology

- As 3-tier monolithic apps

FAST framework and LCF
And for each one of those pieces...

We have to:

- Provide help, support, and examples
- Train and gain mindshare
- Build components and modules
- Maintain security
- Incorporate new technologies
- Innovate and revamp
- React to industry changes
- Test and roll out
We can’t keep up

... or thousands of projects, contributors, testers, examples...
Modern applications are...

- No longer self-contained
- Think service areas
  - 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

Tighter Coupling, Less Flexible/Portable, Less Complex Outer Architecture
And each one is...

- Broken down even more
- An assembly of others’ work
  - Open source
  - Vendor products
  - PaaS, SaaS, modules
- Small, disposable pieces
The result is a conscious choice

- **Someone else built it**
  - And VERY successfully so!
  - Like many, many others
  - Less control
  - More short-term risk
  - More languages
  - Common terminology
  - More time spent on the parts people see, not frameworks!!!
Working this way also means...

- New processes
  - Agile development
  - Automated testing
  - Service-oriented delivery model
  - Microservices
  - DevOps
  - Architecture lifecycle management
  - Open source engagement
So what is this project?
First of all... who:

- Tim Bouffard, Application Architect
- Sam Donnelly, Sr. Application Developer
- Bryan Hopkins, Sr. IT Project Leader
- Anome Mammes, Sr. Application Developer
- Lisa McBriar, Sr. Business Systems Analyst
- Matt Schleindl, Application Architect
So… what is this project?

- Agile development
- Automated testing
- Service-oriented architecture
- Microservices
- DevOps
- Architecture lifecycle management
- Open source engagement

- User interface
- Business logic
- Authentication and security
- Persistence
- API platform
- Testing
- Source repository
- Deployment pipeline
- Platform and scaling
So... what is this project?
Work iteratively

Pull from Backlog
- Based on ISC priorities

Define Criteria
- Identify methodology
- Identify technologies

Build and Evaluate
- Hands-on experience
- Assign scores

Pilot(s)
- Best-of-breed implementation
- Test assumptions
Yeah but where are we now?
Yeah but where are we now?

https://www.isc.upenn.edu/cloud-first-application-delivery-refresh
Results – UI Framework

- Accessibility
- Community and Modules
- Cost
- Design for Cloud
- Development Process and tools
- Operations
- Resource Pool
- Stability/Viability
- Key Capabilities

AngularJS + Bootstrap + Yeoman generator
Possible Points
ReactJS + Bootstrap + Webpack + Yeoman generator + Redux
Results – Server Framework

- Community and Modules
- Cost
- Stability/Viability
- Data Store Compatibility/Security
- Resource pool
- Design for Cloud
- Operations
- Integration
- Development Process and Tools
- IAM

Python + Django
Possible Points
NodeJS + ExpressJS
Results - Local Data Store

- PostgreSQL
- Possible Points
- MongoDB
Details! Decisions...

- **User interface:** AngularJS + Bootstrap + Webpack
- **Business logic:** Django REST Framework + Zappa
- **AuthN and security:** NodeJS + ExpressJS + Passport-SAML
- **Persistence:** PostgreSQL
Details! Placeholders...

- API platform: **AWS API Gateway**
- Backend testing: **Django TestCase + Mocha + Chai**
- UI testing: **Selenium + Karma + Gherkin**
- Source repository: **Gitlab**
- Deployment pipeline: **Jenkins**
- Testing automation: **Jenkins**
- Platform and scaling: **AWS ECS + AWS Lambda**
- Agile development: **JIRA Agile Plugin + Kanban**
Not even started...

- Service-oriented architecture
- Microservices
- DevOps
- Architecture lifecycle management
- Open source engagement
- More...
No more slides. Demos!

**Technology Stack**: Matt Schleindl

**Behavior and Test Driven Development**: Sam Donnelly

**Agile Development**: Lisa McBriar
Comments/Questions

- Questions?
- Website: https://www.isc.upenn.edu/cloud-first
- Comments and suggestions for future topics can be sent to: cloud-first@isc.upenn.edu