A CUSTOMER’S JOURNEY TO CONTAINERS AND OPENSSHIFT

May 9, 2019

Tony Siranni
INTRODUCTION
INTRODUCTION

WHO AM I?

Tony Sirianni: Teacher, Mentor, Engineer

- Currently a SR. Manager at Abercrombie and Fitch Co.
  - Joined the organization over 6 years ago

- A seasoned IT professional with over 25 years of diverse IT experience
  - Overall career has been focused on UNIX systems
INTRODUCTION

WHY DID WE DECIDE TO MOVE TO CONTAINERS?

WHY DO WE NEED TO MAKE THIS CHANGE?

• Our ecommerce platform was due for a full refresh.

• The software vendor is moving towards a container only solution.

• We needed to have tighter control and consistency of our ecommerce platform deployments.

• We wanted to optimize current CI/CD pipeline to streamline delivery.
ASSEMBLING THE TEAM
ASSEMBLING THE TEAM

IT’S A LINUX PLATFORM....

SO WE BELIEVED WE NEEDED A TEAM OF LINUX ENGINEERS.

• In the context of how we approached projects in the past, this made sense.

• This would have enabled us to solve infrastructure problems quickly due to leveraging these existing skillsets.

• A team composition like this would cause issues with the available resources to operate our current infrastructure and to deliver on competing projects.

• We thought there must be another way....
ASSEMBLING THE TEAM

LET’S BUILD A CROSS FUNCTIONAL TEAM

WHAT SKILLS DID WE CHOOSE TO BUILD THE RIGHT TEAM?

• We needed someone to keep the team headed in the right direction from a technical perspective.

• It’s Linux based so there was a good chance we would need someone with a Linux skillset.

• K8S is a resource scheduler and that sounded a lot like workload automation to us.

• We are an infrastructure team and needed a developer’s perspective.

• Other organizations that we talked to were utilizing middleware engineers.

• We needed help managing cross team collaboration and planning.
ASSEMBLING THE TEAM

WOW, THAT WAS A REALLY GOOD CHOICE

HOW DID OUR TEAM SELECTION WORK OUT?

• We are a very collaborative culture, so this cross functional model worked far better than imagined.

• The core members were excited to share what they learned during this journey and this accelerated the whole team’s growth.

• This team composition provided a diverse skillset that allowed the team members to quickly eliminate blockers.

• They constantly held mini-training breakouts to help the others understand how certain technologies worked.

• The team learned to not get stuck on “too deep in details” conversations. They started to test and demonstrate their ideas to each other.

• ALL members of the team became stronger engineers. This was both in their traditional disciplines and in the new world.
SELECTING OUR CONTAINER PLATFORM
SELECTING OUR CONTAINER PLATFORM

PICKING A PLATFORM SHOULD BE EASY

WE'VE DONE SOFTWARE SELECTION PLENTY OF TIMES, WE GOT THIS!

• Picking a platform is difficult when you don’t understand the technology yet. We weren’t sure what questions we should ask them.

• We picked a trusted partner that has experience to help us ask the right questions.

• There are a lot of vendors selling container platforms. We met and talked to multiple vendors to see who had the features we want.

• Choosing the right container platform turned out to be more difficult than we expected.

• We really just needed it to work out of the box.
SELECTING OUR CONTAINER PLATFORM

WE PICKED A PLATFORM

IT WAS TIME TO GET ROLLING.

• We believed that we asked all the right questions and got answers that made sense.

• We had a highly customized design document.

• It looked like the right thing. It appeared to do all the things we wanted and let us bend it to our will.

• We began the implementation and started to customize the platform.

• Almost instantly we ran into issues. Oh no, this wouldn’t do. We needed to start again...
SELECTING OUR CONTAINER PLATFORM

LET’S LOOK AT THE VENDORS WE MISSED

WE NEEDED TO BE QUICK(er) AND RIGHT THIS TIME! THIS TIME...

• We knew more because of the previous attempt.

• We installed the products as part of this selection phase to prove out some of the features.

• We made sure to not run into the same issues by better testing for our requirements.

• We stepped up the pace and got through the process quicker.

• OpenShift better aligned with our needs and was easier to install with Ansible.
WHY OPENSHEIF
WHY OPENSШФT

WHAT WE LIKE ABOUT OPENSШФT

THE PLATFORM ALIGNED BETTER WITH OUR TECHNOLOGY NEEDS.

• It’s a mature platform that has been around over 7 years. Obviously RedHat is dedicated to it.

• It has an existing customer base, enabling us to draw from other people’s experiences to make ours easier.

• RedHat knows how to implement and operate it so we were able to partner with them to help us through the journey.

• It’s off-the-shelf ready so there weren’t too many architectural problems to solve.

• It comes with opinions on how to implement it. This was perfect and allowed us to focus on things that bring value.
LESSONS LEARNED

LOOKING BACK WHAT DID WE LEARN?

THIS JOURNEY WAS DIFFERENT FROM TRADITIONAL ROLLOUTS.

• Being able to reset the environment and start again is really important. No joke, it’s REALLY important.

• Having multiple POC instances helps increase the amount of work the team can accomplish.

• Get the platform online ASAP and start using it to flush out things you had no idea were important.

• Don’t over engineer the implementation and try not to over plan. Keep it agile.

• For a lift and shift solution, you need the same amount of compute resources you required on traditional infrastructure.
LESSONS LEARNED

DID WE LEARN ANYTHING ELSE?

LARGE PROJECTS ALWAYS BENEFIT FROM DOING CERTAIN THINGS.

• Having a dedicated workspace for the team really helped collaboration. This solved all kinds of issues.

• Have a weekly lunch date across the core teams. Share information about the project or just socialize.

• Be open to new ideas and to how others approach problem solving.

• Being supportive of your team members, peer teams, and customers is really important!
THINGS WE WISHED WE KNEW AT THE BEGINNING
THINGS WE WISH WE KNEW AT THE BEGINNING

NOW YOU TELL US

LOOKING BACK NOW IT SEEMS SO SIMPLE.

• A container is not a VM, it’s a trap to start thinking about them in those terms.

• Using an Agile process (or something close to it) is important, even for infrastructure teams.

• The word “container” can mean many things. Make sure you understand the context.

• This journey is actually easier than it looks. It’s all Linux features/ideas that have been around for a long time, just assembled in a new way.

• Even the “experts” can be mistaken. When it doesn’t seem correct, challenge, test, and READ THE DOCUMENTATION!
CLOSING COMMENTS
CLOSING COMMENTS

HOW DID IT TURN OUT?

WELL, THAT WAS UNEVENTFUL!

• We had to adjust our resource allocations in the “11th hour”.
• We made sure we planned for the go-live.
• Our go-live was amazingly quiet!
• This journey proved that we have a great team who can perform under pressure and deliver new technology in a short timeframe.
QUESTIONS?
tony_sirianni@anfcorp.com