Please walk us through a significant project or initiative on a major product where you were in a lead role or a key contributor. Choose an example that you're proud of and that best illustrates your tactical, strategic and leadership abilities.

Begin by describing the project and the team that worked on it, your role on the team, and then tell the story all the way through to a retrospective, including an impact assessment. We would like to understand your investigation and decision making process and how you interacted with cross functional stakeholders. What traps did you fall into or wrong-turns did you take? What lessons did you learn and what changes were made resulting from those lessons (if any)?

In the week leading up to July 4th, 2019, the application development group at KBRA was getting ready to launch two new products: KBRA Analytics, a competitor to S&P's SNL Financial data platform, and KBRA.com 2.0, a revamp of our public facing website that was mobile-friendly. In total, there were approximately twenty-four developers working on these initiatives across multiple teams, and they were all reliant on two systems engineers who set up hosting and other services for their projects.

At this point, KBRA was following a traditional model where development and operations were completely separate with a wall (Jira tickets) in between. Adding to the turmoil was a lack of personnel on the operations side of the house, which led to them cutting corners, accidentally breaking things, making changes in prod and forgetting to commit them, and more. This led to very unhappy folks on both side of the wall, wherein both developers and operations wanted to get more done but couldn't.

Working with our CTO and the head of Infrastructure, I proposed a shift in our approach. We'd move to a true DevOps approach with application development responsible for the infrastructure needed to run the applications and our systems group would provide basic services like networking and VMWare. Due to other circumstances, this shift was handled as part of a restructuring in the company that resulted in the two systems engineers leaving the company without notice.

This meant that I was suddenly responsible for approximately 170 servers, each a "pet"[1] managed by a potentially out of date Puppet script. There had been numerous outages in the weeks leading up to this event and postmortems hadn't been documented if they'd been conducted.

To kick off this massive, sudden transition, I spoke to the developers who had been the biggest, vocal critics of the old way. Not the folks who had just been raising the alarm, but those who were advocating for a different path. The folks who had worked on true DevOps teams in previous positions and those with Linux experience. We quickly identified two developers who'd worked on a Developer Empowerment team before and two others who'd done Linux administration. This was the team that'd herald the transition. We communicated this to the development team as a whole, set up a Slack channel where they could quickly ask for help, and went to work assessing the landscape.

During the first two weeks, we had a severe outage in one of Docker swarms that our infrastructure team shared had happened before, only in the Docker EE swarm, and went away by rebooting

[1] The History of Pets vs Cattle and How to Use the Analogy Properly

"something". Without the two engineers who'd exited, we didn't know what to reboot. Chris, one of the developers on the new team, stepped up and replaced the UCP routing in the swarm with Traefik, the open source tool that was being used in the Docker CE swarms. We worked through the night to do this, Chris doing the bulk of the technical work and myself doing the outward communication to the business, stakeholders, and users.

After that night, I knew Chris was the person to permanently lead the new team and asked him to step into that role. He accepted and, together, we began not only patching and repairing the servers we'd inherited but developing a vision for the future. That vision coalesced as "Development teams will have the responsibility to build and run their applications from inception to product, they will be empowered to do so, and they will be given resources to help them."

One of our development teams, the folks responsible for the web app portion of KBRA Analytics, grabbed the bull by the horns and took control of their own Docker and CI configuration. They moved the files into their repository, split it up as they saw fit, and redeployed their application in a way that made sense to them. I was so busy helping Chris firefight during this period that my role with this trailblazing team was only encouragement. I'd talk to them about their progress, talk them up to other teams and our CTO, and bought them dinner the night they finally released. I also invited them to do an all-hands presentation, sharing what they'd accomplished and demonstrating how other teams could do the same.

At the same time, many of our other development teams weren't keen on the changes. For teams that didn't often need frequent infrastructure changes or hadn't been blocked by the lack of infrastructure resources, they simply saw The Vision™ as new responsibilities. Being on call in case their app went down was new and scary. Troubleshooting something they didn't have experience was new and scary. Having to learn Docker was new and scary.

I took this opportunity to work closely with the biggest cynics, listening to their concerns and letting them vent. I took copious notes while remaining mostly quiet, repeating back what I heard so they knew I was listening. And then I help a fireside chat, wherein I answered all the questions and concerns I'd heard. I reminded everyone of my open-door policy, inviting further feedback and additional questions, concerns and complaints.

I also had questions from our various business group heads. They'd witnessed and justifiably complained about the outages that'd occurred before the transition and they wanted to know what we were doing to prevent the same from happening again. To this end, I prepared a handout to share with them. One side was facts about the transition, about Chris' new team and their qualifications, and the pros and cons of the DevOps approach. I also included a written testimonial from the KBRA Analytics team that had been the trendsetters as well as excerpts from an article discussing ING's agile transformation[2], including DevOps. I hoped the thoughts of other leaders in the financial field would help our stakeholders relate. It worked. We began discussing other points from the ING article and things we could do to improve our processes outside of DevOps.

[2] ING's agile transformation

Near the end of Q3 2019, Chris and I presented a diagram to our CTO detailing all the services we were currently running. We grouped the 170 servers, as well as many hand-configured Azure services, into buckets and talked about what we hoped to eliminate, what we wanted to better orchestrate and keep, and what we wanted to move to SaaS providers. We also requested new hires for the developer empowerment team as two developers we'd leveraged early on were eager to return to application development full-time.

The first initiative we set forth for the developer empowerment team was called "Stabilize Docker Swarms" which was intended to build new swarms that were orchestrated and treated as "cattle" instead of pets. Chris was adamant that this required a new CI/CD tool and I worked with him to articulate **why** because I was initially afraid it was just his preference to use Gitlab over Jenkins. He worked with a number of interested developers to identify a dozen criteria that they would use to evaluate CI/CD tools and I helped him prepare that for a presentation to our CTO. We then evaluated a number of tools, including one that our Head of Product was championing because someone had told him about it at a party.

With the criteria results in hand, our CTO, Head of Product, and developers easily understood why we selected Gitlab and I gave Chris reinforcing feedback; I wanted him to create similar criteria matrixes for future decisions and wanted him to know I'd appreciated the work he'd put in to getting everyone's buy-in.

Chris' team also adopted Terraform for their orchestration and I worked with him to have someone from the team present about that at our CTO's periodic Architectural Review Board. They prepared slides and I provided feedback. We refined the deck and designed a live-coding demonstration. The presentation went so well, our CTO asked them to do it again for all of application development and infrastructure. Because of those efforts, Terraform has been adopted to varying degrees by a number of teams throughout the company. Additionally, I'm working with other teams presently to find training for Terraform so they can get started too.

This story is very fresh in my mind because Chris, his team, and I sat down this past week to reflect on the past six months. We'd set a six-month goal for where we wanted to be by the end of Q1 2020 and I wanted to share their progress with the firm. The team had been having weekly retrospectives to talk about improving their work and processes, but I thought a broader, strategic lookback was appropriate as well.

As with most plans, things hadn't gone exactly as we'd originally thought. We hadn't anticipated completing our transition from Atlassian server products to the Atlassian cloud by this point, so that was a plus, but we also identified some duplicate services we were paying for because not all of the development teams had finished their transition to the new services. Most importantly, Chris conducted a poll on Slack with the developers and learned that, overwhelmingly, they felt that the new empowerment team had made it easier to do their own work.

Chris and I prepared a presentation after the retrospective which we gave this past Friday. We openly shared the original plan and where we are. We shared a new plan for the next six months and

communicated the help we'd need from the developers to get there. We reiterated the best ways for the developers to ask for support and highlighted examples of awesome collaboration between the teams. After the presentation, our Director of Data Management reached out to Chris and I to thank us for sharing.

On a personal level, this experience has been hugely educational and I want to highlight some of those items specifically here.

Change Management: Previously, when I heard the phrase change management, I thought of the heavy processes I'd seen at other firms. During this experience, working with an executive coach and hearing an amazing presentation by Dr. Kim Perkins from The NOBL Collective[3], I learned that change management is about humans. It's about making clear what's changing, acknowledging the loss, giving choices, and helping people feel empowered. It's about engaging with cynics to turn them into champions. Like so many things in the Agile world, it's about communication.

Managing Multiple Teams: I've managed multiple teams for several years at KBRA, but this experience gave me new insight into the challenges that come with it. As I had to focus on the developer empowerment team very much early on, I learned to rely on my reporting managers while still being involved. I started a weekly staff meeting, following the Tactical Meeting format from "Death by Meeting" by Patrick Lenocioni. This helped me get status updates and note items I wanted to discuss more later.

Additionally, I truly saw the impact of an institutional change has on multiple teams and how team dynamics plays into that. We had one team change their DevOps workflow and understanding within a month, and other teams have only just touched the surface today. This can be attributed to a number of things, but really taught me the need to treat each team like an individual, living organism.

Selling a Vision: I had a vision for how DevOps would work at KBRA, but I needed to convince many people to move that vision forward. Initially this was our CTO and head of infrastructure, but over the last few months, it expanded to include business stakeholders, executives, developers and more. Not only did I need to develop the elevator pitch for my vision, but I needed to provide different forms of communication and supporting evidence for different types of people. I've leveraged all types of tools, including demonstrations, book recommendations, testimonials, articles and fireside chats.

[3] https://nobl.io