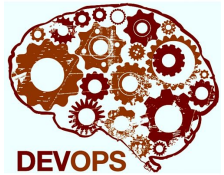
Introduction to Devops and continuous delivery

About the instructor

- Omri Siri
  - Age 32
  - Tel-Aviv
- Work experience:
- Currently
  - Devops freelancer
- Previously:
  - Director of operations @ Avantis - a startup in the AdTech industry
  - Lead DevOPS @ Liveperson
    14+ years of Linux experience



Course Agenda

- Introduction to DevOPS and continuous delivery
- Puppet in a nutshell
- CI/CD with jenkins
- Docker
- Kubernetes overview

Lesson Agenda

- Introduction and goals
- Organization characteristics
  - Organizational pain
  - Identifying waste
- Introducing devops
  - Making a devops transition
  - Change culture
  - Change organization
  - Team structure
- Automation tools categories
  - Collaboration
  - Planning
  - Issue tracking
  - Monitoring
  - Configuration management
  - Source control
  - Environments
  - Continuous Integration and deployment

Introduction and goals

- Everybody is talking about devops
- Devops is not a Job title
- Devops is a movement
- We'll see what devops is comprised of but first lets understand what it solves

Organization characteristics

- Organizational pain
- Identifying waste

Organization Characteristics

- What is our organization structure
- What does the delivery pipeline look like
  - Manual deployments
  - Deployment manuals?
- What functions does the organization have?
  - IT, HR Dev, QA, Bizdev, Project managers etc...
- Policies

Organizational Pain

- Complex delivery pipelines
- Outages and slow recovery
- Legacy systems
- Lack of expertise
- Long deployment cycles
- Slow delivery is related to lost revenue due to lost opportunities
- Frustration and job satisfaction
  - Developers who can’t deploy as quick become frustrated
  - Causes attrition
- Friction
  - Information security saying no to everything
  - IT refusing resources due to bureaucracy
  - No transperancy

Organizational Pain

- QA missing bugs
- Frustrated customers
- Conflicts
  - Between Developers wanting to release code and ops ensuring uptime
  - Business wanting to add features and R&D wanting to close technical debt
- Wasting time

Identifying waste

- All the pains mentioned cause waste
- Waste is directly correlated to lost revenue
- Knowledge waste
  - Constant Re-Orgs shuffle essential knowledge and knowledge is lost in the process
  - Teams not collaborating (Dev and ops, qa and pms etc..)
- Time waste
  - Constant meetings
  - Waiting waste
    - Approvals, tests, deployments, qa etc...
  - Operational waste (asking for more than neccessary just to not need to ask more later..)

Identifying waste

- Complexity
- Repetitive tasks that can be automated
- Long nights?
- Constant human error
- Configuration drift

All these waste resources and aren't producing value

Introducing devops

- Making a devops transition
- Change culture
- Change organization
- Team structure

Devops!

- The whole point of Devops is increasing productivity while:
  - Reducing friction
  - Increase revenue by reducing waste
  - Increasing
- Devops is LEAN
  - Focusing on customer and business value
  - Reducing time to market
  - Finding bottlenecks and relieving them
  - Theory of constraints - I can only go as fast as the slowest constraint
- CAMS - Culture, Automation, monitoring, Sharing
- Continuous Improvement

Making a devops transition

- Devops is not a team its a mindset, a culture, a way of thinking
- This needs to be a company wide change

Change culture

- Make sure everyone knows WHY
  - What are the shared objectives
  - Can everyone feel the pain?
  - Find pain points
  - Every company is different
- Empowerment
  - Devs and engineers should be able to contribute without fear
  - Allow people to use their judgement
- Accountability
  - Everyone is empowered but they are accountable for the results
  - “You broke it you fix it”
  - No finger pointing or blaming
- People are rewarded for doing even if they make mistakes sometimes

Change culture

- Take ownership
  - Ownership should be encouraged and rewarded
- Teamwork
  - Across all teams and employees
  - SHARE KNOWLEDGE!
    - Inside the team - daily standup meetings
    - Globally - This can be through weekly “showoffs” or meetups
  - Mingle - Know your peers - BEER is a great way
  - When people know each other it's easier to work together
  - RESPECT

Change culture

- Learning
  - Continuous improvement of yourself is key
  - Management should promote spending time learning (not only after hours) (4 hours a week?)
  - Give resources for learning (like this course?)
  - Value is exponential!
  - Share knowledge
  - Public meetups are also a great tool to learn
- Trust
  - Trust between groups should be a critical goal to successfully create a thriving culture
- Reinforce the company values and goals
  - Share golas

Change organization

How does the organization adapt to the cultural changes?

- Gain understanding of what we're working on
  - Change team structure appropriately
  - You can't automate what you don't understand
  - Fixing the pain-points
  - Understand the PEOPLE doing the work
  - If you don't know what's not working you can't fix it
  - Gain system wide understanding of the system

Change organization

- Recognize bottlenecks and constraints
  - Even if you can't change everything you can incrementally improve the process to make their lives easier
  - E.g:
    - The "Hey it worked on my machine" issue:
    - Can be solved by creating a production like environment for devs (Did anyone say docker?)
  - Deployment bottlenecks
  - QA bottlenecks
    - Manual testing procedures are slow
    - Environments aren't up to spec
  - Ops bottlenecks
    - Maintaining multiple "Works of art" is time consuming

Change organization

- Bottleneck cont.
  - Better communications
    - Surprising OPS or QA with a complex change will create a bottleneck
- Alter team structure
  - Remove silo teams - when they leave you loose knowledge
  - Share “Tribal” knowledge
  - Devops is not a team its a cultural re-org
  - Full-stack vs specialists
- Streamline procedures
  - IT (devops) should be part of the processes and procedures
  - Find issues in the flow between Dev, Qa and prod
  - Change management
  - Deployment management

Change organization

- Streamline procedures cont.
  - Approval overload
  - Manual double checks
  - Most of the above can be automated to a satisfactory degree
- Addressing objections
  - Some issues can be
    - Everyone thinks they are special - I have to have this very special deployment process
      - Fact is you're probably not
    - Am I scared of change?
    - Devs touching production??
    - Security
  - Streamlining the process is actually a benefit everyone
    - Security patches can be deployed quicker

Change organization

- Addressing objections cont.
  - QA can deploy new testing environments quicker
  - New features can be tested in a near production environment
  - Compliance is much easier when everything is automated

Devops Automation tools categories

- Collaboration
- Planning
- Issue tracking
- Monitoring
- Configuration management
- Source control
- Environments
- Continuous Integration and deployment

Collaboration

- Much more than having meetings or daily standups
- Easily connect teams
- Make sure everyone is up to date - even from remote locations
- Real time chat -> Slack, IRC, skype etc
  - During outages everyone is available and fast
  - Decision making can be transparent if done on a company accessible forum
  - Create discussions
- Have a company wiki with the knowledge publicly available

Planning

- Task/Kanban boards
  - Action items
  - Planning
  - Bugs
- Makes priorities visible to the team
  - Promotes transparency
- Tools like:
  - Trello
  - asana

Issue tracking

- Similar to planning but allows a single location for all the organization to track issues
- Tools include
  - Jira
  - Zendesk

Monitoring

- Monitoring is an essential part of DevOps
  - Measure everything
  - Allows to detect issues early on, and correlate deployments with performance issues
  - Early detection allows for quicker resolution
  - Can and should be integrated in the CI pipeline
  - Should be visible to everyone
  - Should be system wide
  - Tools include:
    - Graphite
    - Pagerduty
    - Logstash & Kibana
    - Statsd
    - Much much more.

Configuration management

- Allows us to enforce a state
  - Prevents configuration drift
  - Should be source controlled - Infrastructure is code
  - Deployment is automated not manually
  - No one touches production directly only through the CM
  - Tools include :
    - Chef
    - Puppet
    - Salt
    - ansible

Source control

- All software assets are tightly controlled and every change saved
  - History is visible to everyone
- This includes changes to configurations and infrastructure
- Promotes compliance and accountability
- Tools include:
  - Git/github
  - Svn

Environments

- Tools exist to allow developers to consistently built an environment which is as similar to production as possible
  - Docker
  - Vagrant
  - AWS

Continuous Integration

- All about incremental change
  - Merge changes to mainline code multiple times a day
  - Find errors early in the development stage
  - Promotes collaboration between devs
- Build, test and deploy to QA multiple times a day and integrate changes from multiple developers
- Immediate feedback to code pushes
- Tools include:
  - Jenkins
  - Teamcity
  - Travis CI

Deployment

- Deployments should be an extension of sort to CI
- Goal is to make deployments BORING
- Deploy frequently and early
- Shouldn’t be a special occasion
- Errors should be caught early on by our streamlined process
- Should be as automated as possible
  - Doesn’t have to be continuous
- Immutable servers
- Tools include:
  - Jenkins
  - Cloudformation
  - kubernetes

Summary

- Devops is about continuous change and improvement
- CAMS :
  - Culture
  - Automation
  - Monitor
  - Share