# DevOps Architectural Proposal

## Problem Statement

1. The limited number of product environments (dev, qa, beta, stage, production) creates bottlenecks in the release schedule as one or more Dev, QA or marketing tasks are delayed due to the unavailability.
   1. The number of trained and authorized IT personnel that can create environments limited.
   2. The people that can do this also have to support the organization on problems that are often higher priority.
2. At times, the run up to release cutover has taken weeks increasing schedule time and costs.
   1. Multiple dry runs are often required to advance to the next product environment.
   2. Cutover to production can require multiple attempts within the same maintenance window and has required additional sessions.
      1. Employees who participate in the late night cutovers are not compensated. After one such late night, one production DBA said it was time to find a healthier place to work.
      2. The shared pain of production cutovers is a cultural joke.
3. Integration of new technology into the continuous integration and deployment processes is successful but often requires multiple attempts.
   1. Differences in OS versions/patches, library levels and incomplete installation/administration steps all contribute to this problem.
   2. This problem begins with rollout to other developers resurfaces when it is incorporated in the build process, deployed to qa, stage and finally production environments.
4. Internal fixed capacity environments (raised floor, servers, disks, software licenses, etc.) require a larger ongoing capital expenditure than elastic alternatives that are priced based on actual demand.
5. Maintaining fixed capacity environments requires high levels of personnel time.
   1. Finding and allocating sufficient raised floor space
   2. Procurement and upgrade of resources
   3. Hardware failures require driving to Inverness and Cheyenne
   4. Currently, there is only one person in Colorado to deal with these issues.
   5. Hiring more people is not a practical solution
      1. Long lead times to replace departing individuals
      2. Adding people is a quantum fixed cost that doesn’t scale.

## Solution - DevOps

* [What is DevOps](http://theagileadmin.com/what-is-devops/)?
  + **“DevOps is the practice of operations and development engineers participating together in the entire service lifecycle, from design through the development process to production support.”**
* ***What is DevOps not?*** *–Reducing operations staff or Devs doing Operations job*
* Technologies of DevOps: git, Vagrant, Coreos and Docker.

## Benefits

1. Reduced costs: Fixed costs are replaced by costs that scale to actual demand.
2. Less disruption (schedule impact) as technology adoption is done through containers that bundle dependencies and allow configuration management of apps, libraries, OS, clusters and cluster management.
3. Increased synergy as Development simplifies the Build team and Operations concerns throughout the release lifecycle.
4. Reduction in last minute surprises and potentially eliminating late night cutover sessions. Although overkill for Troppus, flickr deploys to production 10 times a day.
5. Higher motivation as key personnel are turned loose on more rewarding tasks.

# Demo Agenda

* Problem Statement
* Solution
* Benefits
* Demonstrate
* Elasticity of services using 913 netlocation REST endpoint
* Configuration management of environments (Vagrantfile, Dockerfile, etc.)
* Ease of deployment
* Review proposed next steps
* Continue testing netlocation elasticity
  + Upgrade to JMeter v2 913 netlocation.
  + Document error loss as netlocation servers resources are added and dropped reflecting the 913 reboot across time zone use case
  + Add additional health checks to determine bottlenecks revealed by JMeter.
* Prototype missing areas: firewall, cluster logging & monitoring, upgrades w/o service disruption.
* Investigate continuous integration and automation tools to allow the following:
  + Dev adoption of new technology using containers (Mongo)
  + GIT controlled build and deployment of OS, patches, apps, etc (containers)

## References

[The Agile Admin – What is DevOps?](http://theagileadmin.com/what-is-devops/)

[Wikipedia - DevOps](http://en.wikipedia.org/wiki/DevOps)

[Gartner report: Ok, We Get it, Docker’s Great. But what for?](http://blogs.gartner.com/richard-watson/ok-get-dockers-great/)

[Just Enough Developed Infrastructure](http://www.jedi.be/blog/2012/05/12/codifying-devops-area-practices/)

[Theory of constraints](http://nl.wikipedia.org/wiki/Theory_of_constraints)