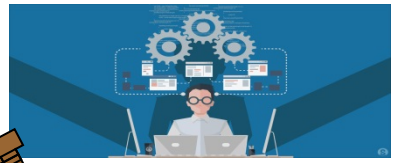
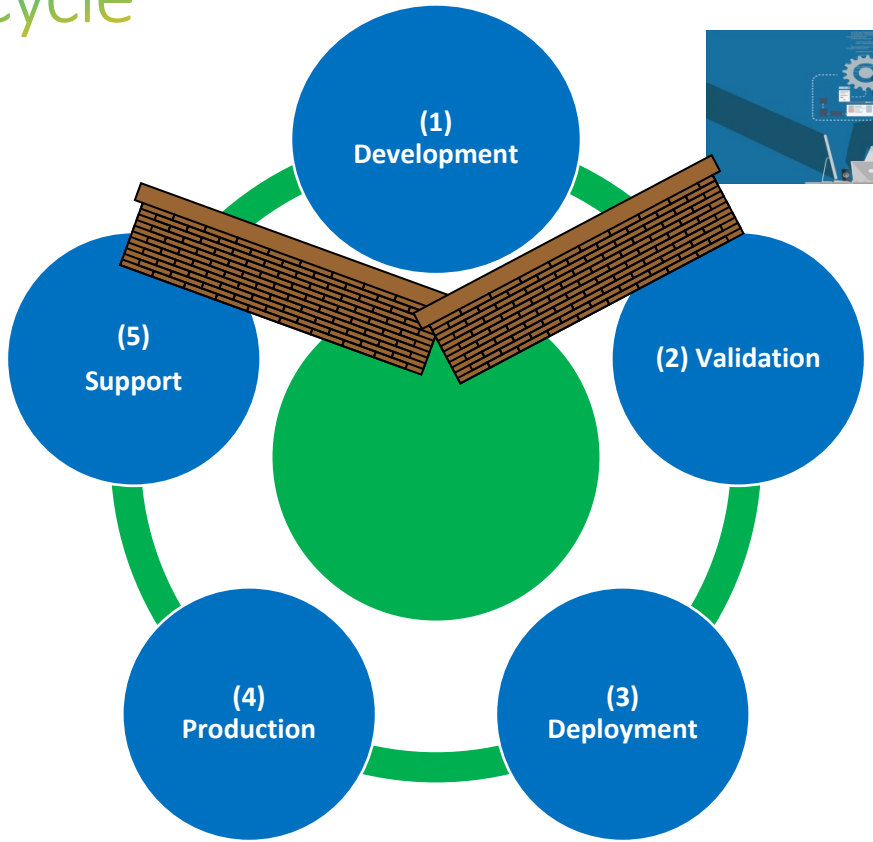
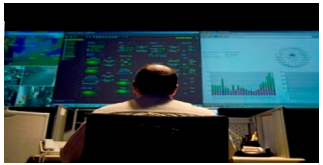
Agile:
An Overview

Narayanan Subramaniam
LinkedIn: http://www.linkedin.com/in/cnsubramaniam

Software Lifecycle

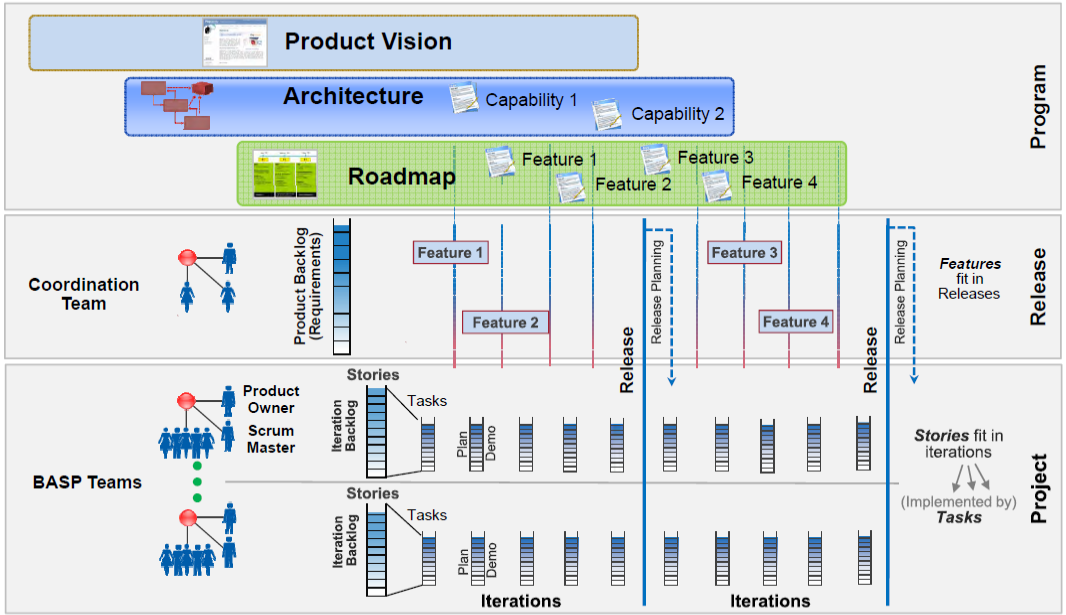






Boeing Approach for Agile including SW Architecture

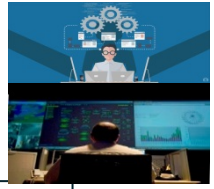
Engineering, Operations & Technology | **Boeing Research & Technology** FaST | **Networked Systems Technology**



Based on an original diagram by Dean Leffingwell

@credit: Boeing SW … https://apps.dtic.mil/dtic/tr/fulltext/u2/a558044.pdf

Agile In A Nutshell



- ✓ Agile attempts to break the barriers between Dev and Test and focuses on small increments while failing fast with emphasis on openly communicating feature behaviours, expectations and results
- ✓ Agile is NOT development engineers doing QA full time, or Architecting / Designing on the fly WITHOUT initial thought/groundwork or reviews - such misinterpretations usually lead to disasters. Cross-pollination helps a lot but is with the purpose of creating heightened awareness.
- ✓ The core principles of taking more ownership, automation of test, as key to survival, are often missed out or partially adopted.
- ✓ It is a different question whether every project is amenable to Agile or not ... certain projects may not be suited for Agile at the outset.
- ✓ Focus on the principles, not just the structure of teams, meetings etc.

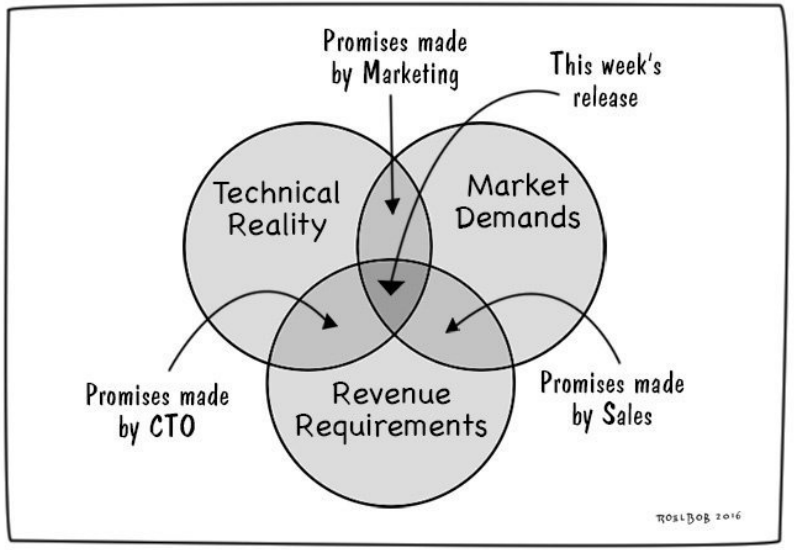
DevOps:
An Overview

Narayanan Subramaniam
LinkedIn: http://www.linkedin.com/in/cnsubramaniam
(+91-9341969647)

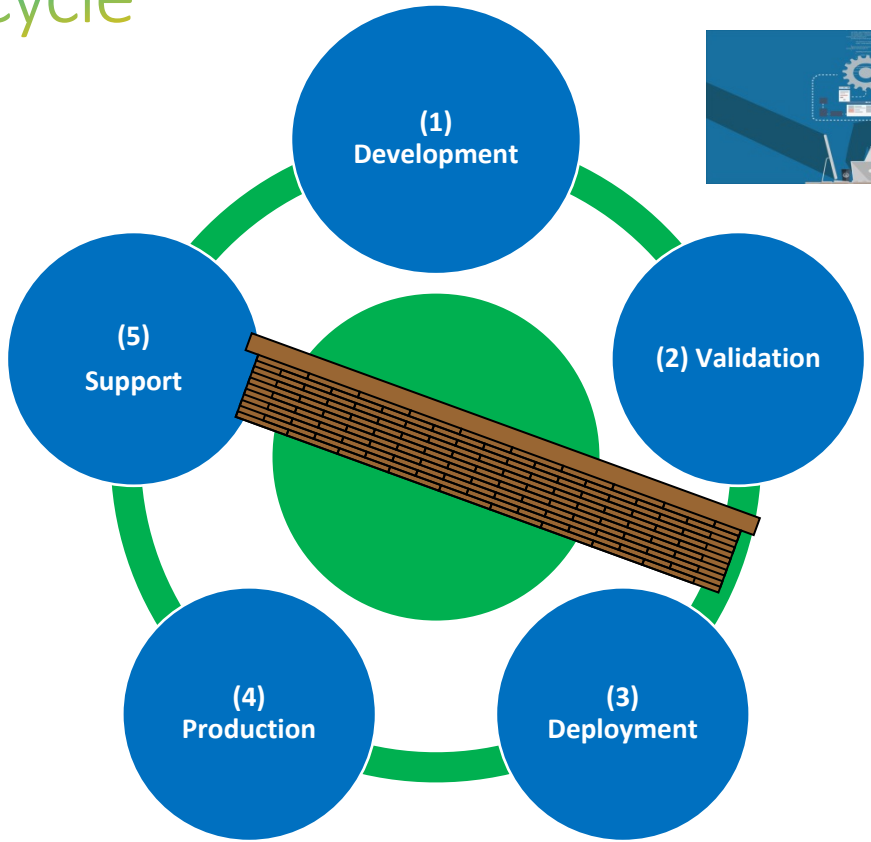
Introduction

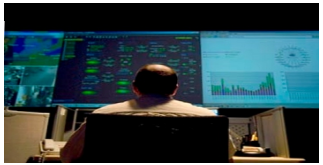
Agenda:

- ✓ **DevOps …. is it all about Tools, PagerDuty and the Cloud ?**



Software Lifecycle

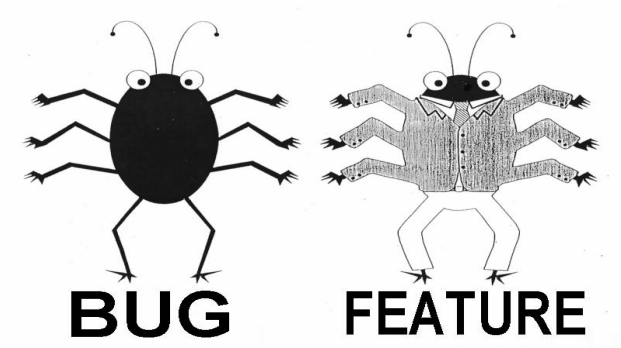




Software Lifecycle

Challenges:

- ✓ **How do I fix my bug, and quickly roll it out ?**
- ✓ **How can I pilot my new feature “carefully” ?**

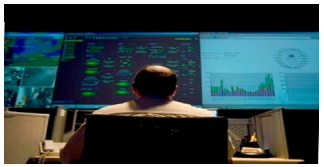


Software Lifecycle

Challenges:

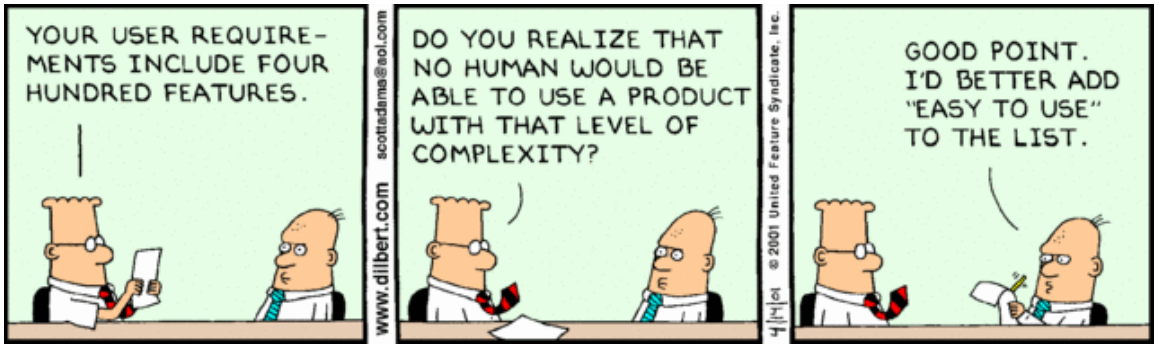
- ✓ **What are the eco-systems my software is deployed in ?**
  - **Configuration combinations**
  - **Interop combinations**
  - **…….**

Software Lifecycle

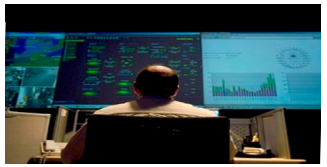


Challenges:

- ✓ **How is my software used ?**
  - **What features are used ?**
  - **Who uses it and when/why ?**
  - **What is the UE (User Experience) ?**



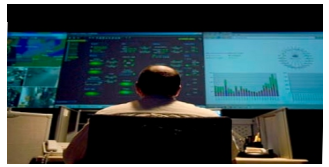
Software Lifecycle



Challenges:

- ✓ **Is the software “vulnerable” ?**
  - **How well does it perform ?**
  - **Does it scale as expected ?**
  - **Is the software being “attacked” ?**

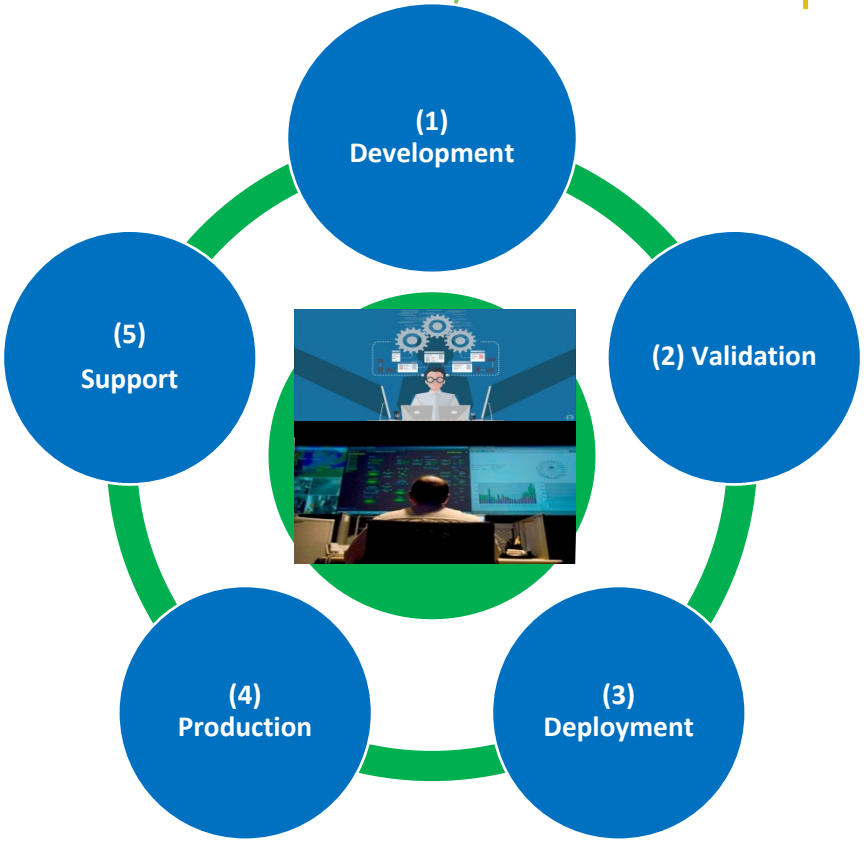
Software Lifecycle



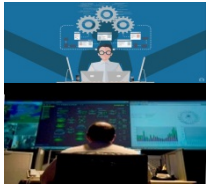
Challenges:

- ✓ **How do I know if the design is “future-proof” ?**
  - **Will it run out of capacity soon ?**
  - **How many more users can it support - SLA ?**
  - **How do I plan for “breaking” changes ?**

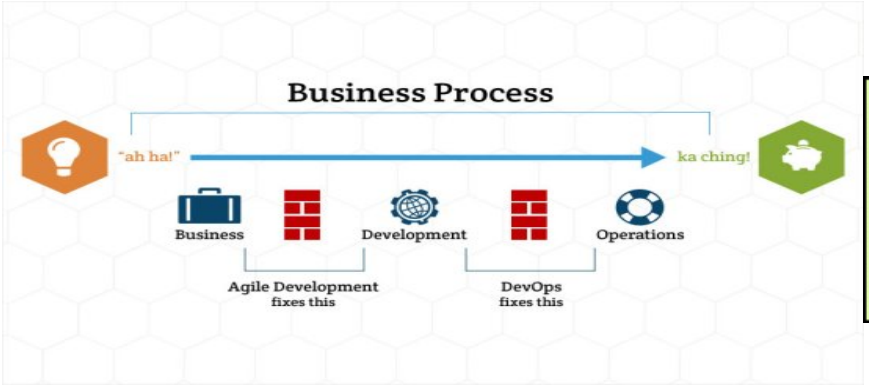
Collaborative Software Lifecycle - DevOps

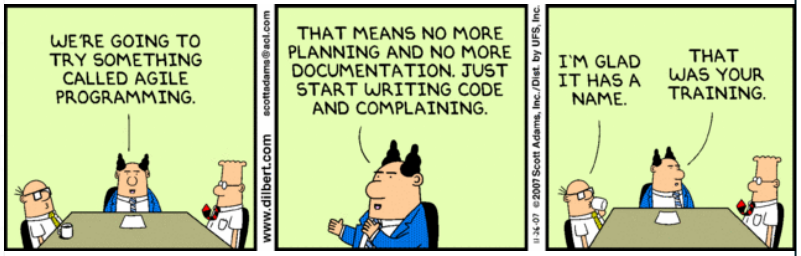


Applying DevOps - Code

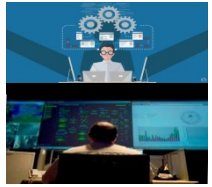


- ✓ **How do I fix my bug, and quickly roll it out ?**
- ✓ **How can I pilot my new feature "carefully" ?**
  - **Continuous Integration & Deployment (CI/CD)**
    - **Plan Big, Execute Small - Architecture/Design is key**
    - **Incremental Code Development & Test – AGILE**

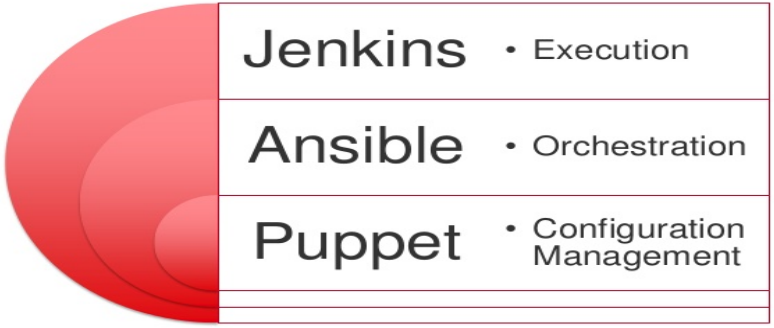




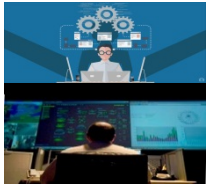
Applying DevOps - Code



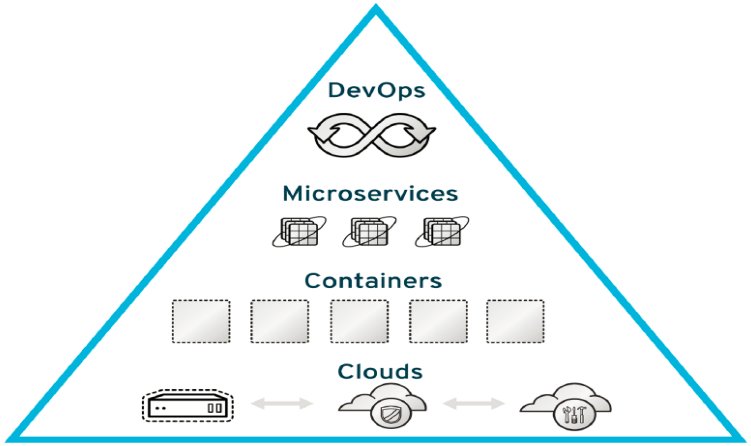
- ✓ **How do I fix my bug, and quickly roll it out ?**
- ✓ **How can I pilot my new feature “carefully” ?**
  - **Continuous Integration & Deployment (CI/CD)**
    - ➢ **Preferably one source tree**
    - ➢ **Automate – builds, installs, tests, configuration**

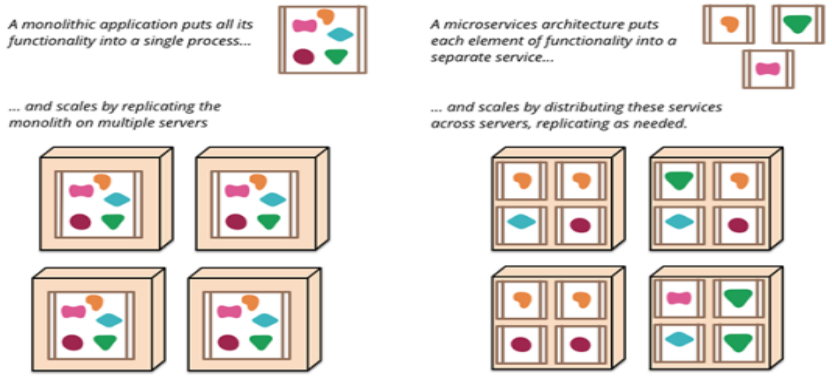


Applying DevOps - Design

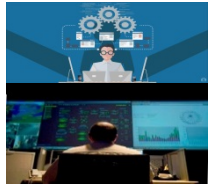


- ✓ **How can I pilot my new feature "carefully" ?**
  - **Continuous Integration & Deployment (CI/CD)**
    - **Highly Available – Microservices architecture/design**
    - **Fail-fast and Fail-forward**

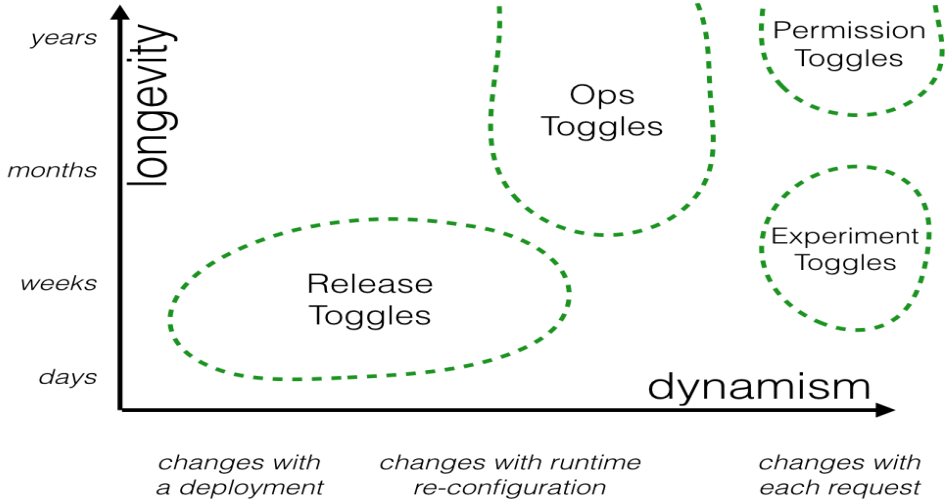




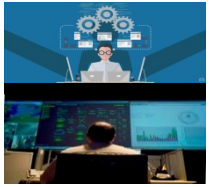
Applying DevOps - Design



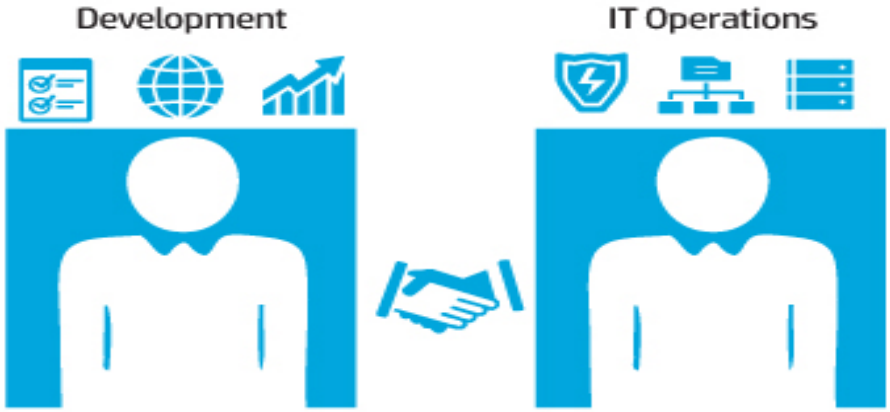
- ✓ **How can I pilot my new feature "carefully" ?**
  - **Continuous Integration & Deployment (CI/CD)**
    - **Feature Toggles – Canary Releases**



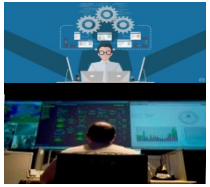
Applying DevOps - Deployment



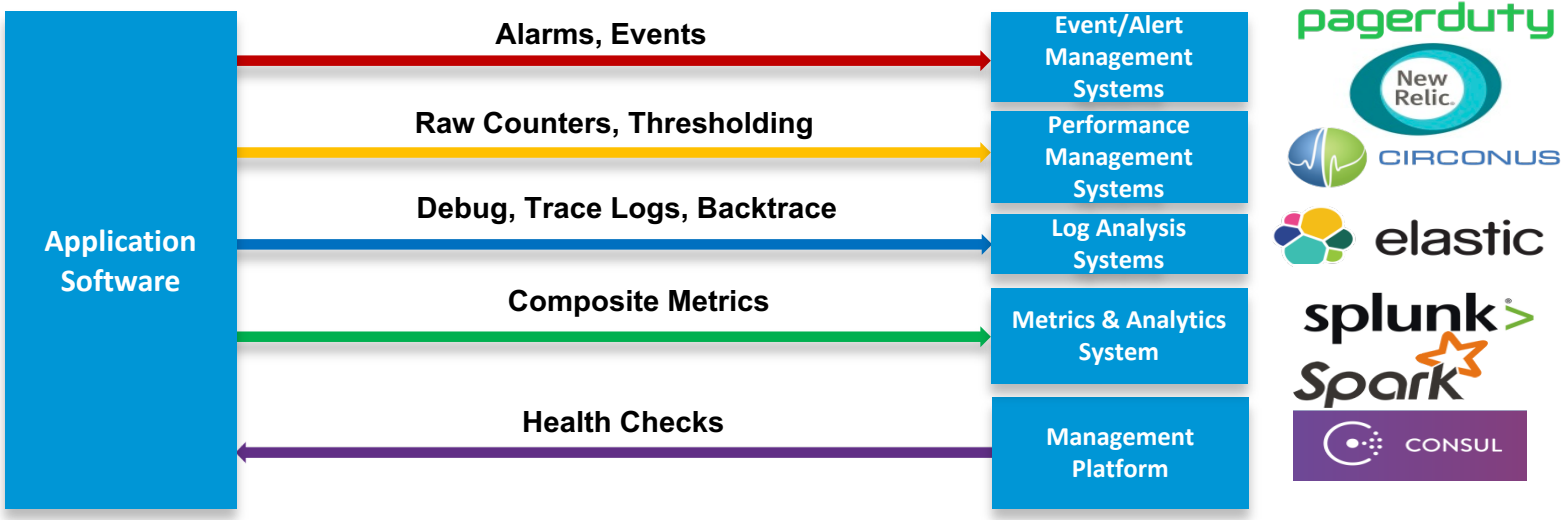
- ✓ **What are the eco-systems my software is deployed in ?**
  - **Multiple Data-Center, Regions/Geographies ?**
  - **Self versus Partner Hosted ?**



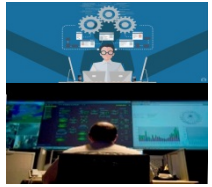
Applying DevOps – Metrics



- ✓ **How is my software used ?**
- ✓ **Is the software "vulnerable" ?**
- ✓ **How do I know if the design is "future-proof" ?**

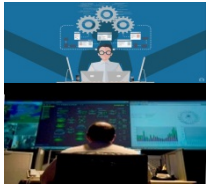


Applying DevOps – Metrics

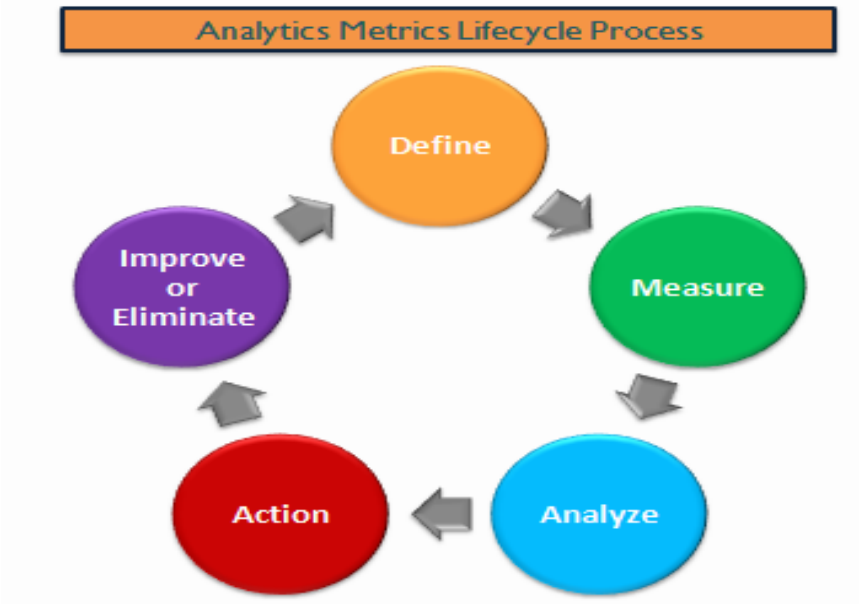


Technology Metrics	Business Metrics	Value Metrics
Deployment frequency	Revenue & Profit	Productivity
Lead time for changes	Avoided costs	Quality
Change error rates	Customer feedback	Opex
Failure rates	Cash flow	Capex
Lines of code	Time to market	
Availability	ROI & NPV	
Recovery time	Customer satisfaction	
Job satisfaction	Renewal rates	
	Cost per service/unit	

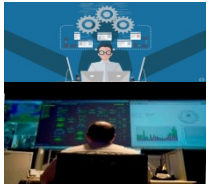
Applying DevOps – Metrics



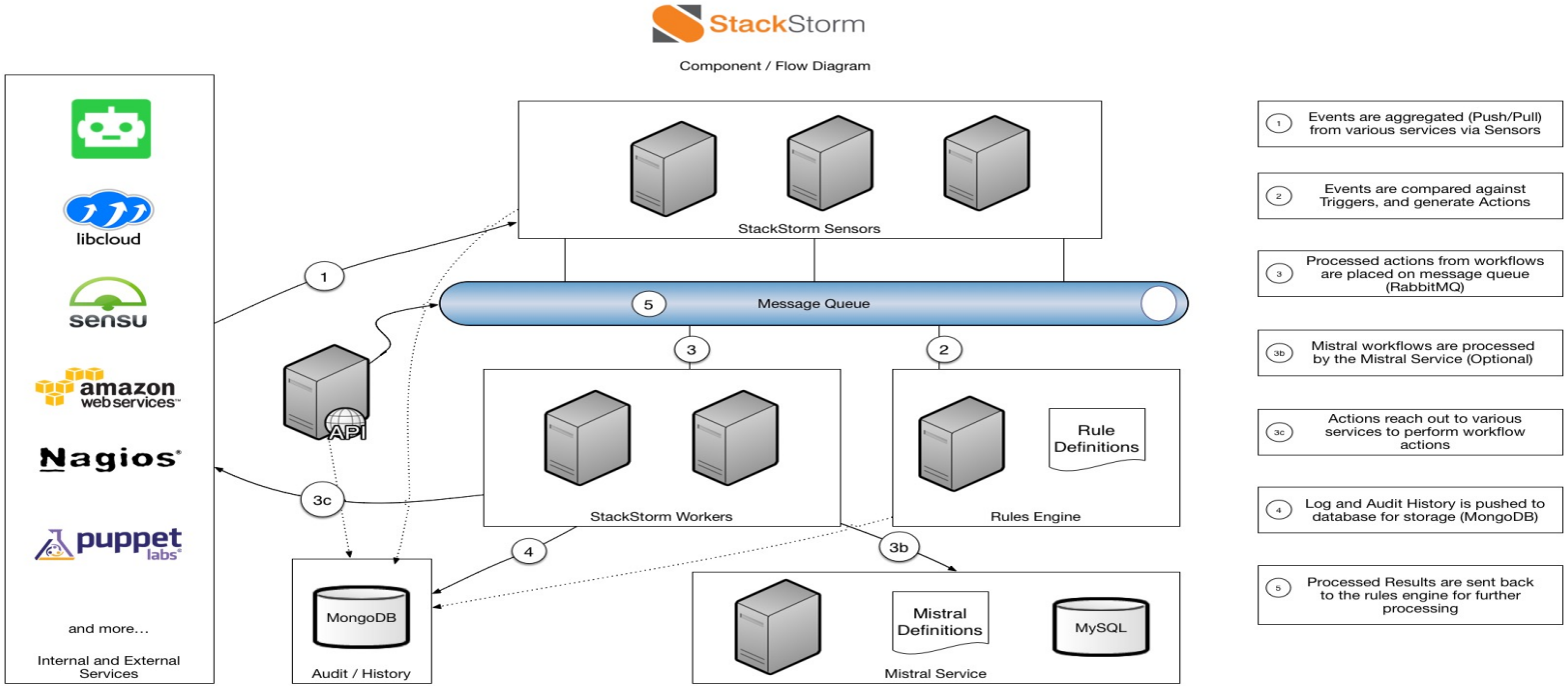
- ✓ **Metrics are Ever Changing !!**
  - **Leverage CI/CD not for features alone, but for metrics**



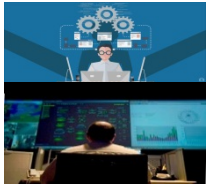
Applying DevOps – Metrics



✓ Automated Orchestration based on Metrics and Events

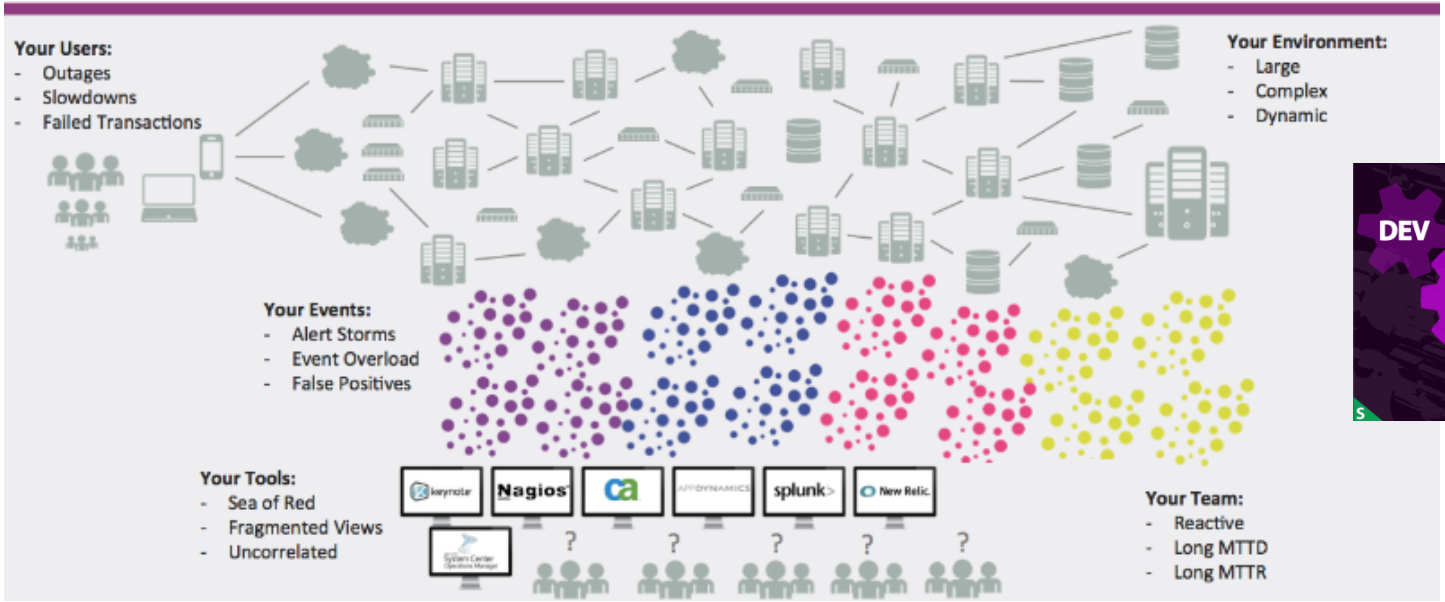


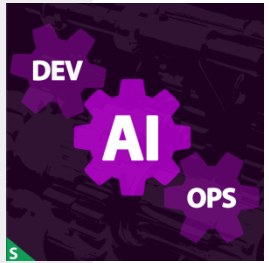
Applying DevOps – Metrics



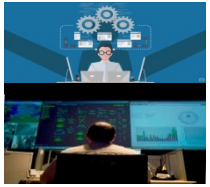
✓ **DataScience and Machine Learning for Scale**

Scale and Change Creates Trouble



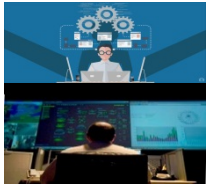


Applying DevOps – Security

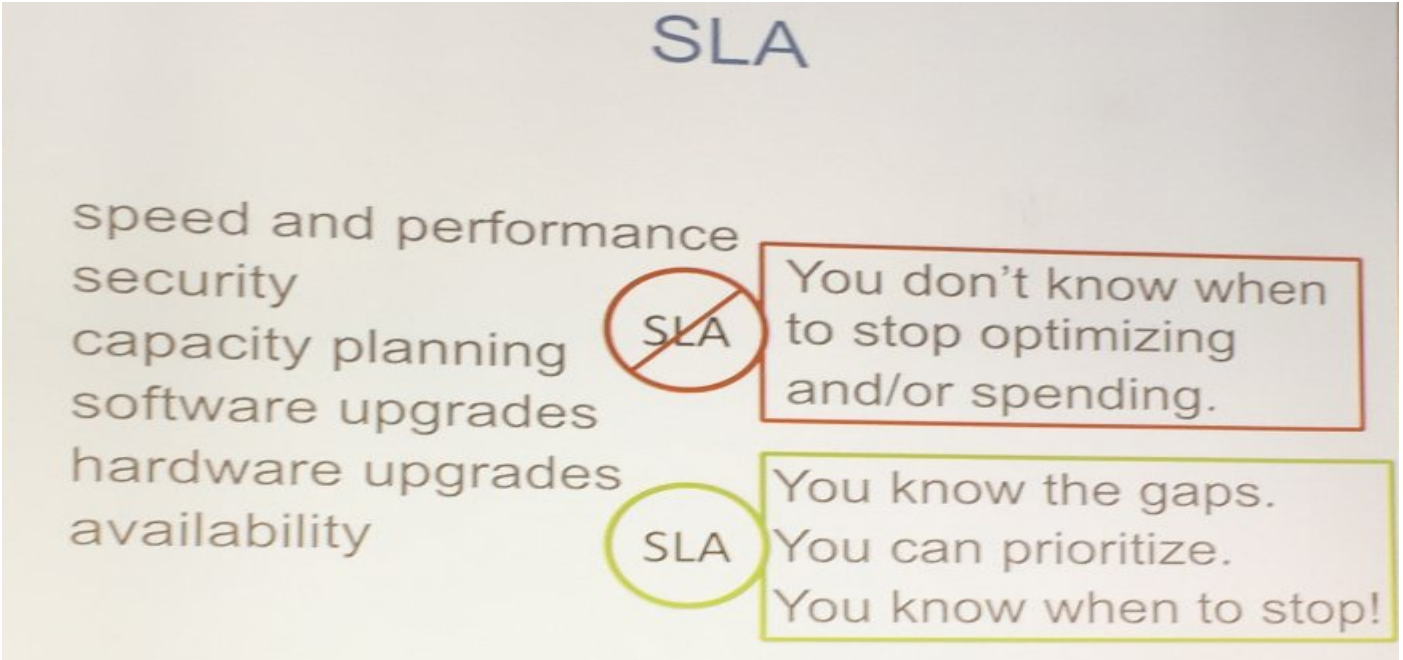


- ✓ **Regulation**
  - **Data Export: Data crossing regions, countries**
  - **Data At Rest: Data retention policy, access control**
  - **Legal Intercept**
- ✓ **PII (Personally Identifiable Information):**
  - **Obfuscation, RBAC to Operations/Admins to reverse-map PII data**

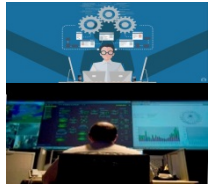
Applying DevOps – SLA Management



- ✓ **SRE as a Gatekeeper to SLA's**

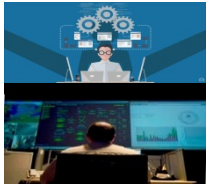


DevOps – Backlog Management



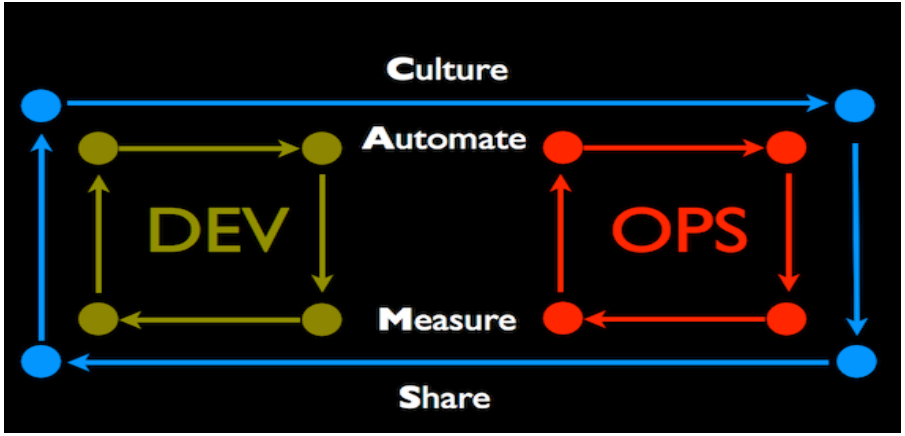
- ✓ **Two Backlogs !!**
  - **Feature Backlog – Product Management driven**
  - **DevOps Backlog – Metrics and Automation driven**
  - **Team Rotation helps**

Summary



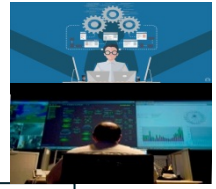
Takeaway:

- ✓ **DevOps:**
  - **Mindset – E2E**
  - **Agile execution**
  - **Automation**
  - **Metrics, ML**
  - **Regulation, PII**
  - **Multiple Backlogs**



- ✓ **Focus on the Principles, Tools are incidental**
- ✓ **SME, Architects, Product Management remain key stakeholders**
- ✓ **DevOps Principles applicable beyond Cloud based products**

DevOps In A Nutshell



- ✓ DevOps attempts break the barrier between Dev, Test and Operations with ever increasing emphasis on automation not just of test but of operations as a whole.
- ✓ Metrics based design is the key paradigm for DevOps. It usually builds on the goodness of Agile TDD.
- ✓ Agile is NOT about development engineers doing Operations or Operations folk writing feature code. Cross-pollination helps a lot but is with the purpose of creating heightened awareness. Such misinterpretations usually lead to disasters.
- ✓ The core principles of taking more ownership, metrics based design to quantify SW quality/usability, automation of operations and support, as key to survival, are often missed out or partially adopted
- ✓ Focus on the principles, “culture, measure and share”, and not just on “automate”

THANK YOU !