Operational Environment Resiliency

DevOps

Adaptability

© 2020 GMV Property – All rights reserved
CONTENT

INTRODUCTION

DEVOPS AT GMV: ADAPTABILITY

BENEFITS OF DEVOPS: COVID-19
DevOps Adaptability

Introduction

Experience at the GMV Control Centers Business Unit

In particular in the Eutelsat section

- Also some relevant experiences from other areas
- Agile approach for implantation

Relying on DevOps practices to handle new situations and challenges
DevOps Adaptability

Understanding DevOps

Len Bass, Ingo Weber, and Liming Zhu—three computer science researchers from the CSIRO* and the Software Engineering Institute—suggested defining DevOps as:

"a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality"

* Commonwealth Scientific and Industrial Research Organisation (CSIRO) is an independent Australian federal government agency responsible for scientific research
DevOps Adaptability

Main PILARS

- (No) implication of Team Members
  - Motivation/Compromise
  - Classical roles/Egos

- (Undesired) Software Quality
  - Manual intervention
  - Low level of testing

- (Static) Processes/Procedures
  - Lack of improvements

- (No) Feedback from final users
  - Unknown real expectations

CULTURE

AUTOMATE

MEASURE

SHARE
Initially we saw a lack of tools and processes
But then we identified a problem: a culture

Training, coaching, workshops, ...
Need to believe in the change
Motivation and Compromise

New AGILE approach:
- New multidisciplinary and self-organized teams
- New roles: Product Owners (at GMV), Scrum Masters but classical Project Managers

Usage of SCRUM and/or KANBAN
GMV Experience: AUTOMATE

- Code Production
- Build/Release Generation
- Validation and Testing
- Deployment
DevOps Adaptability

GMV Experience: AUTOMATE

CODE PRODUCTION

- From SVN repository to a **GIT**
  - Repository located at **customer premises**
  - Huge improvement in merges and release generation
  - Usage as **Git Flow** as higher layer

- Development in separate **features (US)**
  - Improved testing and validation
  - Better integration
GMV Experience: AUTOMATE
CODE PRODUCTION

- Introduction of **Atlassian Tools**: JIRA/Confluence/Bitbucket/Bamboo
  - Excellent integration
  - Focused on Agile

- Other tools to help developers
  - Development IDEs
  - Integration of dependencies, style, metrics
User Stories have a common Definition of Done (DoD)
- Code controlled and reviewed
- Test cases passed
- ... and other activities project dependent

But a particular Acceptance Criteria
- Defined and validated by the Product Owner
DevOps Adaptability

GMV Experience: AUTOMATE

BUILD GENERATION

- Builds managed by **Bamboo**
  - Incremental, Nightly
  - Automatic after commit
  - Execution of tests
  - Deployment of artifacts
  - Configuration (Ansible)

- Significant effort for **reducing the build generation time**
  - Improvement of our own build systems
  - New compilers (i.e. clang for C++)
  - Parallelization and distribution
DevOps Adaptability

GMV Experience: AUTOMATE

VALIDATION AND TESTING

- **Automatic testing** considered a pillar of the process
- Automation in **unit**, **integration** and **system** tests
- In progress
  - **TDD** – Test Driven Development
  - **BDD** – Business Driven Development
  - Huge effort for introducing automatic tests in legacy code (refactoring)

- **Validation across the sw. development life cycle:**
  - Before control changes → GIT pre-hook for checking **metrics**
  - After control changes → **Code review** process (pull requests) + **Metrics** (coding & security)
  - Build generation → Execution of test cases (**unit test**)
  - Deployed system → Execution of **integration test** cases + **Cross validation** process
  - Staging environment → Automatic end-to-end tests
    → Customer **demo**
GMV Experience: AUTOMATE

VALIDATION AND TESTING

- Involved tools
  - Metrics
    - SonarQube, CppCheck, Cpplint, Checkstyle, Formatter, Gcov, Covertura,
  - Bitbucket
    - Code Review
    - Integration of Metrics
  - Testing framework
    - Xray: manual and automatic tests
  - Testing tools
    - Google Test, Junit, Jest, Mockito, Cucumber, ...
    - Robot Framework, procedure automation tool
GMV Experience: SHARE

- Feedback from Product Owners → Demo
- Feedback from Customer → Docker containers

- **Continuous deployment** at customer facilities
  - Microservices
  - Refactoring of legacy systems
  - Acceptance/Validation environment
  - Ready for operation deployment

- From staging to **operational** platform managed by Eutelsat
  - Automatic build deployment
  - Manned activation (with seconds of downtime)
Deployment effort:
- from 1000 hours per year to 200 hours

Defects reported to customer:
- 25% with respect to before introducing Agile/DevOps

Others:
- Code coverage (depends on project nature)
- Team efficiency (velocity, sprint review, motivation)
- Improvements (sprint retro and retro of retros)
- **Customer satisfaction**
DevOps Adaptability

DevOps Benefits: COVID-19

- DevOps critical for COVID-19
- Nominal activity at GMV continued with minimal impact.
  - **ENVIRONMENTS** adapted:
    - Remote access. Powerful corporative infrastructure
    - **Dockers containers** for project resources
    - **NoMachine** for accessing remotely to particular environments
  - **TEAM:**
    - Scrum Ceremonies already integrated in the workflow
    - Team **communication** and coordination
DevOps Adaptability

DevOps Benefits: COVID-19

- Nominal activity at GMV continued with minimal impact.

  - **SW Efficiency with QUALITY**: automation in place
    - Ms Teams, IDE Clion, JIRA tools and plugins, ...
    - Collaboration, Remote Pair programming
    - Mature environments with a high level of automation
    - Development model based on testing

  - Uninterrupted **DELIVERY** and customer **feedback**
    - Remote demos
    - Scheduled meetings and milestones
GMV Experience: CHALLENGES

- How to deal with a flexible development with a **fixed price** and **fixed schedule** project:
  - Agile Contract

- Deal with other non-yet-agile entities
- Keep customer collaboration
- Keep motivated teams
- Keep improving the efficiency from development to delivery
Thank you