

**Faculdade Presidente Antônio Carlos**  
**Engenharia de Computação**

Projeto de Automação de Operações e Entrega Contínua para  
Empresa de Desenvolvimento de Software

Ivan de Moura Miranda

Faculdade Presidente Antônio Carlos  
Engenharia de Computação

21 de novembro de 2016, v1

# Sumário

- 1 Introdução
- 2 Situação da empresa no início do projeto
- 3 Trabalho realizado
- 4 Resultados obtidos

# Introdução

Maiores desafios encontrados por desenvolvedores em seu ambiente de trabalho (STACK OVERFLOW, 2016):

- Expectativas irrealistas
- Documentação ruim
- Requisitos mal especificados
- Processo de desenvolvimento ineficiente

# Introdução

Soluções existentes:

- Modelos e *frameworks* de boas práticas:
  - *ITIL*
  - *COBIT*
  - *CMMI*
- Metodologias ágeis:
  - *SCRUM*
  - *XP*
  - *DevOps*

# Introdução

## Objetivos

- Modificar o ambiente de desenvolvimento de uma empresa baseando-se no estudo de boas práticas e ferramentas gratuitas buscando a automação das tarefas de operações e o aumento da qualidade dos softwares desenvolvidos
- Validar os benefícios da automação e entrega contínua em um cenário real
- Avaliar os métodos propostos

- Associação de direito privado, constituída na forma de sociedade civil de fins não lucrativos, com autonomia administrativa e financeira, fundada em 2016
- Tem por finalidade central realizar ações sociais de utilidade pública na área de desenvolvimento de *software*, produtos de *software* e pessoas (DEVELOP, 2016)
- Situada no centro de Ouro Preto

- Divisão de equipes por projeto
- Comunicação informal
- Setores pequenos formados por especialistas

# Situação da empresa no início do projeto

O processo de desenvolvimento

- Aplicadas algumas práticas do *SCRUM* como *product backlog*, divisão do projeto em *Sprints* e reuniões diárias
- Ciclo baseado em definição de requisitos, implementação e testes manuais
- Compilação de código fonte no computador do desenvolvedor
- Processo de implantação de novas versões totalmente manual
- Espera acumular muitas alterações para realizar a entrega



# Situação da empresa no início do projeto

Dados coletados

Tabela 1: Dados do repositório de código fonte no início do projeto

Métrica	Quantidade
Número de <i>commits</i> realizados	125
Número de artefatos gerados	65
Número de mesclagem de código realizados	6
Número de <i>bugs</i> reportados pelo cliente	20
Número de atividades desenvolvidas pelo time que foram registradas no sistema de controle de atividades	70

# Situação da empresa no início do projeto

Dados coletados

Tabela 2: Características das entregas no início do projeto

<b>Data da entrega</b>	<b>Tempo necessário</b>	<b>Número de tentativas má sucedidas até a conclusão da entrega</b>
16/08/2016	2 horas	0
23/08/2016	6 horas	3
30/08/2016	3 horas	1
07/09/2016	6 horas	5
14/09/2016	8 horas	7
19/09/2016	4 horas	2

# Situação da empresa no início do projeto

## Problemas do cotidiano

- Esquecer de apagar diretórios de locais indevidos
- Executar a versão de produção em diretórios incorretos
- Permissões indevidas
- Dificuldade em gerenciar configurações
- Execução de *scripts* em produção sem os devidos testes anteriormente

# Trabalho realizado

## Automação das tarefas de entrega

- O *upload* de novas versões de forma manual é comprometido pela velocidade e estabilidade da conexão de internet no escritório
- Decidido compilar o código em um servidor remoto, sem problemas relacionados a conexão de internet
- Necessário garantir a estabilidade da base de código antes de automatizar este processo

# Trabalho realizado

Garatindo estabilidade a base de código

- Testes automatizados
- Refatoração constante
- *Code review*
- Integração contínua

# Trabalho realizado

Entrega contínua

- Entrega de *software* antes:

- Acumulava várias atualizações para realizar entregas com baixa frequência.
- Picos de esforço relacionados a entrega
- Muitas alterações para investigar a causa de *bugs*
- Alto risco

- Entrega de software depois:

- Consegue entregar pequenas atualizações com alta frequência
- Pouco esforço relacionado a cada entrega
- Poucas alterações para investigar a causa de *bugs*
- Baixo risco

# Trabalho realizado

Ferramentas utilizadas

- Docker
- Gitlab
- Jenkins
- Zabbix

# Resultados

Dados coletados

Tabela 3: Dados do repositório de código fonte no fim do projeto

Métrica	Quantidade
Número de <i>commits</i> realizados	434
Número de artefatos gerados	115
Número de mesclagem de código realizados	19
Número de mesclagem de código rejeitadas	3
Número de <i>bugs</i> reportados pelo cliente	6
Número de atividades desenvolvidas pelo time que foram registradas no sistema de controle de atividades	34
Tempo médio de execução da <i>build pipeline</i>	5 minutos



# Resultados

Dados coletados

Tabela 4: Características das entregas no fim do projeto

Data da entrega	Tempo necessário	Número de tentativas má sucedidas até a conclusão da entrega
14/10/2016	13 segundos	0
18/10/2016	17 segundos	0
23/10/2016	1 minuto e 21 segundos	0
25/10/2016	1 minuto e 17 segundos	0
26/10/2016	16 segundos	0
28/10/2016	12 segundos	0
31/10/2016	14 segundos	0

# Resultados

## Comparativo

**Tabela 5:** Comparativo baseado em uma média mensal dos dados coletados antes e depois da conclusão do projeto.

	Antes	Depois
<b>Tempo médio necessário para realizar a entrega de uma nova versão</b>	5 horas	33 segundos
<b>Média de tentativas má sucedidas até a conclusão de uma entrega</b>	3	0
<b>Número de commits realizados</b>	62	434
<b>Número de artefatos gerados</b>	32	115
<b>Número de mesclagem de código realizadas</b>	3	19
<b>Número de mesclagem de código rejeitadas</b>	1	3
<b>Número de bugs reportados pelo cliente</b>	10	6

# Resultados

## Feedback da equipe de desenvolvimento

- Realizando alterações pequenas e constantes é mais fácil identificar e corrigir *bugs*
- Não precisa dedicar um dia para fazer a entrega pois é totalmente automatizada
- Maior satisfação do cliente com a redução do tempo de resposta para correção de *bugs*
- O *code review* se mostra burocrático em situações de urgência, mas colabora com o compartilhamento do conhecimento sobre a base de códigos.


## Trabalhos futuros


- Centralização de *logs*
- Uso de orquestradores para contêineres
- Implantação contínua
- Provisionamento de *hardware*

## Conclusão

- Metodologias ágeis são extremamente vantajosas quando aplicadas corretamente
- A automação de tarefas manuais economiza muito tempo investido no projeto
- Existem muitas ferramentas gratuitas eficientes para alcançar os objetivos desejados
- O principal fator para o estabelecimento de um ambiente de desenvolvimento eficaz é a colaboração de todos os membros do time e a busca constante por automação e melhoria do processo

## Referências I

 DEVELOP. *Estatuto da Associação DevelOP*. Ouro Preto: [s.n.], 2016.

 STACK OVERFLOW. Stack overflow developer survey 2016 results. 2016.  
Disponível em: <<https://stackoverflow.com/research/developer-survey-2016>>.  
Acesso em: 10 out. 2016.