Caderno: wiki
Criada em: 01/10/2014 16:26 **Atualizada...** 01/10/2014 16:27
URL: http://www.mobiltec.com.br/blog/index.php/tag/deploy-automatizado/

http://www.mobiltec.com.br/blog/index.php/tag/deploy-automatizado/

Automatização de processos: Como fazer?

Postado **29 de julho de 2013** por **Marketing Mobiltec**

Na maioria dos projetos de TI há processos na fase entre o desenvolvimento do código até a liberação de um sistema ou uma funcionalidade em produção. Geralmente esses processos seguem um padrão, um ciclo que é repetido a cada novo *release*. Então por que não fazer isso de forma automatizada? Com base nessa questão, neste post irei apresentar as experiências práticas que passei e continuo passando para automatizar os processos nos projetos que atuo. Para essa automatização utilizei duas ferramentas:Jenkins, Open Source, e o Octopus Deploy, grátis com a condição de possuir apenas um projeto ativo por vez.

Leia mais

Para iniciar o post, vou contextualizar dois cenários baseados em dois clientes (com desenvolvimento de sistemas web e sistemas móveis) para os quais estamos trabalhando para automatizar processos deste a integração dos sistemas até seu *deploy* em produção. Em um deles, cliente C, também mantemos seus servidores onde publicamos seus sistemas *web* e serviços de comunicação com os sistemas móveis na internet. Sendo que para realizar a manutenção desses servidores possuímos uma VPN entre nosso ambiente Mobiltec e os servidores de produção do cliente, o que facilita o processo de *deploy*.

Já o outro cliente, cliente R, possui uma equipe de infraestrutura que mantém seus servidores e sistemas seguindo padrões de segurança e acessos as máquinas de forma rigorosa. Por esta razão, torna-se necessário seguir processos burocráticos que acabam onerando a liberação de novas funcionalidades em produção. Ao atingir o ponto onde a automatização de seus projetos até produção se tornou um grande desafio de arquitetura, deveríamos definir como fazer e como argumentar a razão desta mudança, enfim,

apresentar ao cliente as vantagens que teríamos com a automatização dos projetos!

Mas como tudo começou?

Para o nosso cliente R, temos um processo de geração de pacotes onde organizamos os instaladores, fontes, arquivos de atualização entre outros arquivos/documentos que geramos a cada *release*. Esse processo é algo custoso pois além de compilar e testar, devemos montar a estrutura de pastas e organizar os arquivos nela, onde perdíamos uma média de 20 a 30 minutos na geração desse pacote.

Na epoca, com o objetivo de facilitar esse processo, foi criado um Batch extremamente simples que realizava o processo de compilação dos projetos e organização das pastas e arquivos. O Batch em si ainda levava alguns minutos para executar todo o processo, mas com isso liberávamos um membro da equipe desse trabalho permitindo que ele se focasse em outra atividade.

Só que queríamos mais, agilizar mais! Como utilizamos o TFS para versionamento de código aqui na Mobiltec, começamos a utilizar o TFS Build, mas não conseguimos obter muitos resultados satisfatórios. Com a participação em eventos descobrimos o Jenkins, conhecemos melhor o Octopus Deploy e então começamos uma nova abordagem.

Mas o que são essas ferramentas e para que servem?

O **Jenkins**, é uma ferramenta de integração contínua desenvolvido em Java. É uma derivação do famoso Hudson que era mantido pela Sun. Após a aquisição da Sun pela Oracle, por diversas razões, vários de seus desenvolvedores e o fundador do Hudson optaram manter uma versão totalmente livre.

Sua instalação e configuração é muito simples, então nesse post apenas irei apresentá-lo e

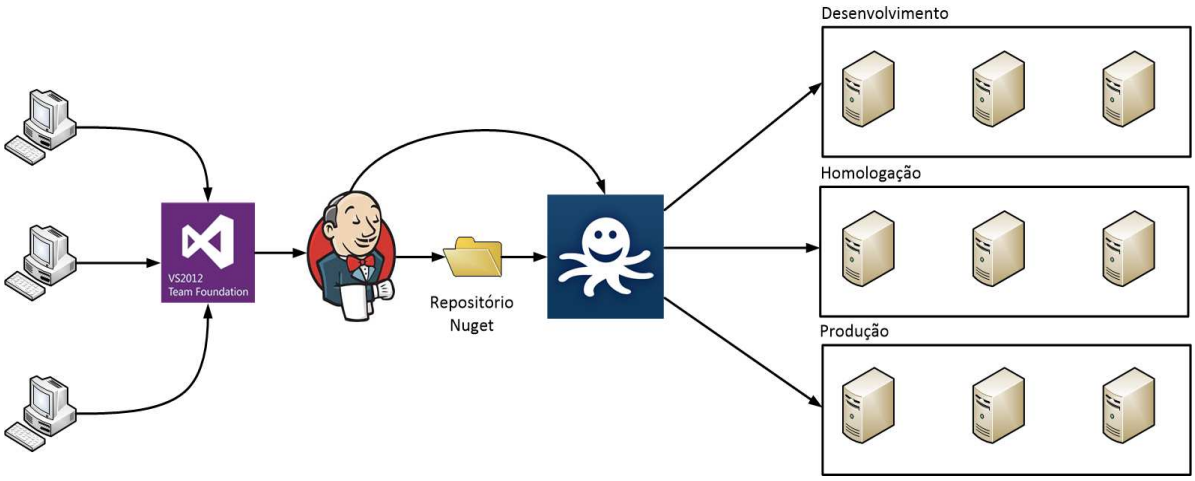
mostrar como estamos o utilizando. No próprio site do Jenkins há documentação bem detalhada de como instalar e configurar a ferramenta.

O **Octopus Deploy** é uma ferramenta de *deploy*automatizado que visa realizar o *deploy* de sistemas ASP.NET e Windows Services, de forma simultânea em diversas máquinas de um determinado ambiente. Uma vantagem que auxilia no processo de *deploy* é a possibilidade de definir variáveis para que sejam aplicados em arquivos de configuração conforme o ambiente em que está sendo aplicado.Outra vantagem é que seus *deploys* são baseados em pacotes Nuget, o que permite uma fácil comunicação entre os sistemas de integração com ele.

Sua instalação requer um pouco mais de conhecimento e é um processo mais longo, pois é necessário instalar um servidor e clientes do Octopus onde serão realizados os*deploys*. Mas em seu site há uma documentação que explica o processo de forma bem detalhada. Com isso, iremos apenas apresentar sua utilização.

Mas e como elas se integram? Como funciona o processo automatizado?

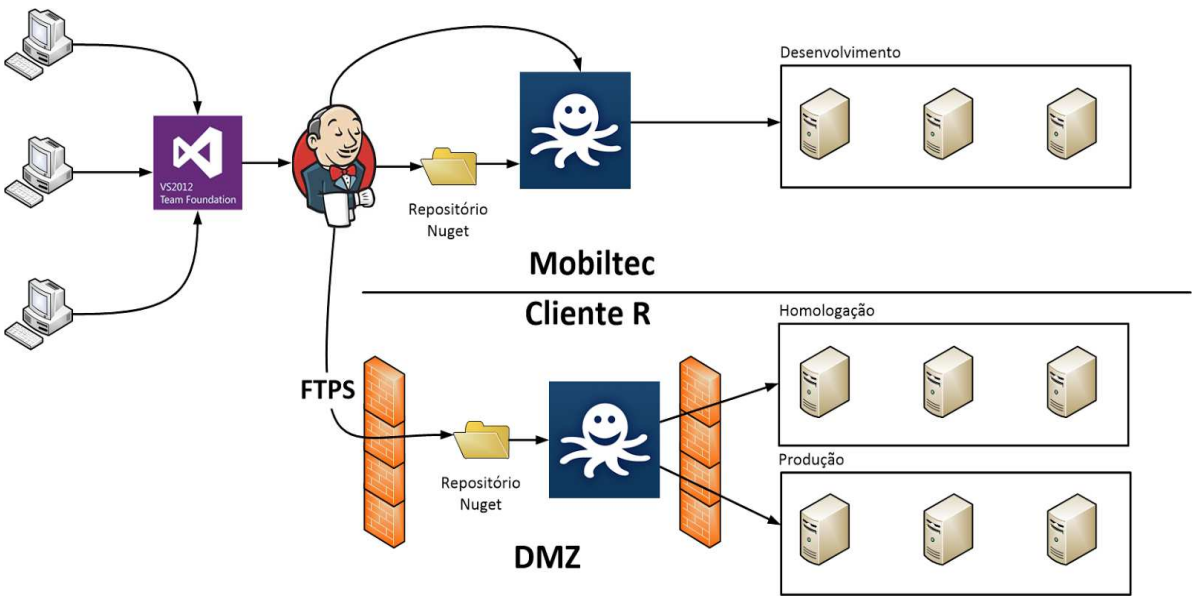
Como já mencionado, aqui utilizamos o TFS como ferramenta de versionamento de código. O Jenkins já vem pronto para trabalhar com CVS mas ele possui um *plugin* para trabalhar com o TFS, o que permite que ele identifique *check-ins* e baixe ou atualize as fontes em seu *workspace* – e a partir daí seja realizada a compilação do código. Para realizar a compilação ele utiliza de outras ferramentas como o Ant, para o Java, o NAnt para o .NET e com base nelas você pode compilar ou executar processos para outras linguagens ou *scripts*. No nosso caso, após a compilação geramos pacotes Nuget que posteriormente são redirecionados pelo Jenkins em um Repositório Nuget. A partir desse Repositório Nuget, o Octopus cria um *release* que será utilizado para realizar o *deploy*nos ambientes, conforme apresentado na imagem abaixo.



O Octopus Deploy, fornece uma aplicação, *tools*, de linha de comando, que permite criar*releases* e até solicitar o *deploys* de alguma *release* em um ambiente, de forma totalmente remota. Com base nessa *tools* e por meio do Ant e NAnt, é possível fazer com que após o processo de integração e disponibilização do novo pacote Nuget no repositório o Jenkins já pudessemos realizar o *deploy* da nova versão nos ambientes desejados.

Como ainda não possuímos testes automatizados em alguns projetos, nesses por padrão, realizamos de forma automática somente em Desenvolvimento. Após testes manuais via Octopus realizamos o *deploy* para o próximo ambiente com apenas um clique.

O cenário apresentado é o mais próximo que temos com nosso cliente C, a única diferença é que temos ambiente de Desenvolvimento, QA, Homologação e Produção. Já para o cliente R, por questões de segurança trabalhamos com dois servidores do Octopus, um internamente na Mobiltec e outro na infraestrutura do cliente R. No nosso servidor do Octopus, realizamos o *deploy* em desenvolvimento de forma automática, porém no cliente R eles são responsáveis pela criação do *release* e *deploy* em Homologação. Após testes por parte deles, *deploy* em Produção, conforme apresentado na imagem abaixo.

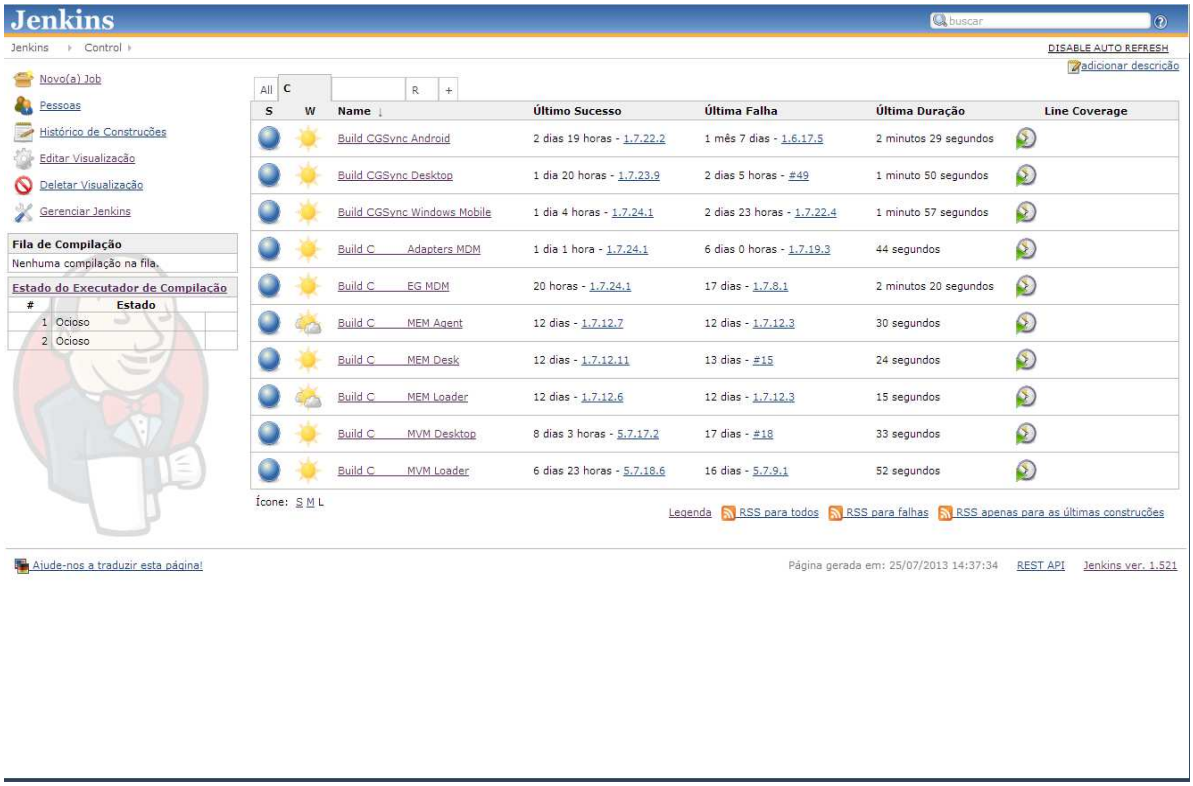


O que mais me chama atenção no Jenkins é que ele basicamente executa “comandos” que são passados para ele. Isso permite que o Jenkins seja usado muito além de uma ferramente para compilar/integrar/testar.

Hummm… e como está hoje?

Não está “uma Brastemp” ainda… mas já ganhamos muito tempo para nos focarmos em outras atividades. Infelizmente ainda não temos testes automatizados em todos os projetos, então precisamos realizar testes manuais antes de passar. Mas já temos a capacidade de corrigir algum *bug* em questão de minutos e após testes, disponibilizarmos corretamente em produção.

Atualmente temos todos os projetos sendo integrados no Jenkins e empacotados para *deploy* via Octopus.



E todos que são possíveis de serem implantados via Octopus estão sendo!

C Projects

	C Desenvolvimento	C QA	C Homolog-Prod	C Prod-Prod
MDM Adapters C	**1.7.24.1** 24 July Success	**1.7.24.1** 24 July Success		**1.7.24.1** 24 July Success
MDM C	**1.7.24.1** 24 July Success	**1.7.24.1** 24 July Success	**1.7.23.7** 24 July Success	
MEM Agent	**1.7.12.7** 12 July Success	**1.7.12.7** 12 July Success		**1.7.12.7** 12 July Success
MEM Desktop	**1.7.12.11** 12 July Success	**1.7.12.11** 12 July Success	**1.7.12.11** 12 July Success	**1.7.12.11** 12 July Success
MEM Loader	**1.7.12.6** 12 July Success	**1.7.12.6** 12 July Success		**1.7.12.6** 18 July Success
MVM Desktop	**5.7.17.2** 17 July Success	**5.7.17.2** 17 July Success	**5.7.17.2** 17 July Success	**5.7.17.2** 17 July Success
MVM Loader	**5.7.18.6** 18 July Success	**5.7.18.6** 18 July Success	**5.7.9.16** 9 July Success	**5.7.9.16** 9 July Success

Conclusão

Tenho certeza que a maioria das pessoas que leram a introdução se perguntaram: “Mas por onde eu começo?”, “Isso é complicado por isso, por aquilo, por…”, “Não vejo nenhuma vantagem nisso!”, “Mas eu não possuo testes automatizados!”, “Mas…” Sim! A ação de automatizar os processo é complicado, exigiu muito estudo e pesquisa e em alguns casos demorei mais de ano para poder implantar. Mas posso afirmar com certeza que vale cada minuto gasto para implantar, pois hoje não gasto nenhum minuto nesses processos e nunca mais irei gastar!

Nos próximos post irei apresentar como configurar as ferramentas e integrá-las, caso

alguém tenha alguma dúvida, deixe seu comentário!