UNICESUMAR – UNIVERSIDADE CESUMAR
CENTRO DE CIÊNCIAS EXATAS TECNOLÓGICAS E AGRÁRIAS
CURSO DE GRADUAÇÃO EM ENGENHARIA DE SOFTWARE

Gerenciamento de Configuração

Marcos Vinicius Oliveira Carneiro

MARINGÁ – PR
2020

PYTHON

Em Python a criação de uma branch é realizada de forma padrão, são utilizados os comandos git fetch upstream para atualizar os dados do repositório, git checkout -b <branch-name> upstream/master cria uma nova branch clone local, git add <filenames> adiciona os arquivos e git commit -m '<message>' faz o commit adicionando uma mensagem sobre a alteração e o commit que foram feitos, git push origin <branch-name> faz o push no git. Há três tipos de branch em Python, a branch master que é a branch que lança as novas atualizações. Branch de manutenção que é uma branch para correção de bugs e além disso, evita que a compatibilidade entre as outras releases sejam quebradas. E uma branch com menos de 5 anos não é uma branch de manutenção, é de segurança. As únicas alterações que são realizadas na branch de segurança são as que corrigem problemas como travamentos, invasões, alterações de privilégio.

Para realizar o merge das branches é feito o push dos commits, eles são verificados e então é realizado o merge, mas, se ocorrerem conflitos no merge, eles devem ser resolvidos, geralmente os conflitos surgem por causa das alterações realizadas em diferentes branches, contendo arquivos diferentes. A resolução é feita manualmente, checando os arquivos que estão em conflito, editando-os e removendo os marcadores de conflito que são adicionados pelo git. Após isso, é só realizar o commit desses arquivos e fazer o push normalmente. O pull request deve seguir o estilo do Python, ou seja, caso o código seja em Python, ele deve seguir o PEP 8, caso seja em C, ele deve seguir o PEP 7. Se houver algumas discrepâncias, o desenvolvedor responsável por fazer a pull request conseguirá corrigí-las, porém, se não seguir nenhum dos guias (PEP 7, PEP 8), o pull request será colocado em espera até que resolvam o problema de formatação. Após as correções, deve-se tomar cuidado com as versões anteriores, para que o pull request não seja rejeitado e deve ser feito testes para verificar se o pull request está correto e será aceito. Além disso, para que ele (pull request) seja aceito é necessário esperar, pois, são inúmeras solicitações de pull request, portanto, até que sejam revisados, o seu request fica na espera. Quando o seu pull request for revisado, eles colocarão comentários para informar como o seu código pode ser melhorado, então e feita as mudanças informadas no comentário e reenviado o pull request. Esse processo será repetido até que a sua solução seja útil, importante para eles.

O code review é um dos principais gargalos do Python por causa da falta de revisões. Por mais que seja feita uma busca para identificar bugs e vários problemas serem corrigidos, não pode ser realizado o merge, pois, ninguém

revisou a solução proposta. Para realizar a revisão é necessário seguir alguns passos. Sendo eles, obter uma cópia do repositório, criar testes e executá-los. Verificar os problemas que devem ser corrigidos e executá-lo. Fazer check-out e realizar o pull request. Caso as alterações cause problemas no arquivo C, execute o buid novamente. Inicie o prompt do Python e tente reproduzir o problema. Com o pull request aplicado, o problema deve ser resolvido. Deve-se também tentar encontrar outros problemas que o autor possa ter esquecido de mencionar. Após executar todos os testes, poderá enviar um pull request, porém, deve-se estar ciente e garantir que os testes sejam aprovados.

As releases são dividas em vários estágios (etapas) sendo elas, pre-alpha, onde não há nenhum lançamento oficial. Alpha, são como lembretes para os desenvolvedores mudarem a semântica ou adicionar algo em Python. Beta, é a primeira publicação para o público, permitindo que os usuários de sugestões de novos recursos e identifiquem falhas, problemas. Release Candidate(RC), só os desenvolvedores principais podem corrigir bugs e que geralmente são graves (travamentos, por exemplo). Final, é quando a versão final é lançada e apenas o RM (Release Manager) consegue realizar alterações. Após a publicação final, inicia-se um novo ciclo de desenvolvimento. O controle de versão é feita principalmente pelo git e mercurial e a rastreabilidade de requisitos pode ser feita pelo plugin Sphinx que adiciona diretivas e funções que identificam e relacionam documentos. Além de funcionar como ferramenta de gerenciamento de requisitos orientada a documentos.