TEC | Tecnológico de Costa Rica

Área Académica de Ingeniería Mecatrónica

MT-7003 Microprocesadores y Microcontroladores

Tarea 1

GitHub, Pytest y Flake 8

Prof. Ing. Rodolfo Piedra Camacho

Realizado por:

César Alonso Argüello Salas

José Pablo Vásquez Rojas

II Semestre 2021

Preguntas Teóricas (16 pts, 2pts c/u)

1) ¿Diferencie la herramienta Git de Github?

La herramienta Git es un sistema de control de versiones, de código abierto, que permite a los desarrolladores de software administrar y llevar un registro de los cambios en el código fuente a lo largo del tiempo. Por otro lado, Github es una plataforma de alojamiento de repositorios, basada en la nube, que incorpora el sistema de control de versiones de Git. Además de contar con todas las funcionalidades de Git, Github posee una interfaz de usuario que lo hace mucho más sencillo de aprender y manejar, al igual que otras funciones que facilitan el desarrollo colaborativo entre usuarios [1], [2].

2) ¿Qué es un branch?

En Git, un branch se refiere a un puntero que apunta a una versión o commit específico de un repositorio. Los branches se pueden utilizar para generar distintas versiones paralelas de un repositorio, donde se pueden realizar cambios individuales en cada branch sin afectar a los demás [3].

3) ¿Qué es un commit?

Un commit se refiere a cada uno de los hitos que han sido guardados durante el desarrollo de un proyecto. Al realizar un commit, Git captura el estado de los archivos que han sido previamente "preparados" (*staged*) y guarda los cambios en el repositorio, llevando un historial de todos los commits realizados. Además, el usuario deberá ingresar un mensaje con una breve descripción de cada commit que realice.

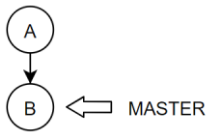
Los commits permiten llevar un cronograma de los cambios realizados y guardados a lo largo del tiempo, los cuales se pueden revertir o simplemente consultar más adelante [3], [4].

4) ¿Qué es la operación cherry-pick?

La operación cherry-pick consiste en seleccionar uno o varios commits específicos de una rama y aplicar sus cambios dentro de otra rama. Esta operación es útil cuando se desean copiar algunos commits o cambios específicos de un branch a otro, evitando realizar un *merge* o un *rebase* completo entre los branches [3], [5].

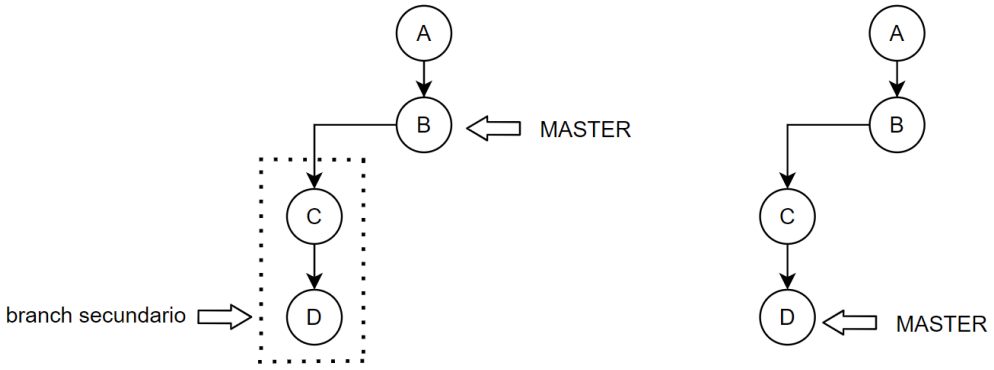
5) Explique de forma gráfica como cambia el “master” de un repositorio cuando se hace merge de un Branch.

Se tiene un branch principal, en el cual se han realizado diversos commits, como los son A y B en este caso.



Caso 1:

Se desea agregar una funcionalidad nueva al proyecto, por lo que sea crea un branch secundario en donde se trabaja dicha característica a partir de la última versión del master, es decir, se trabajará a partir del commit B. De esta manera se crea el commit C y subsecuentemente el commit D.

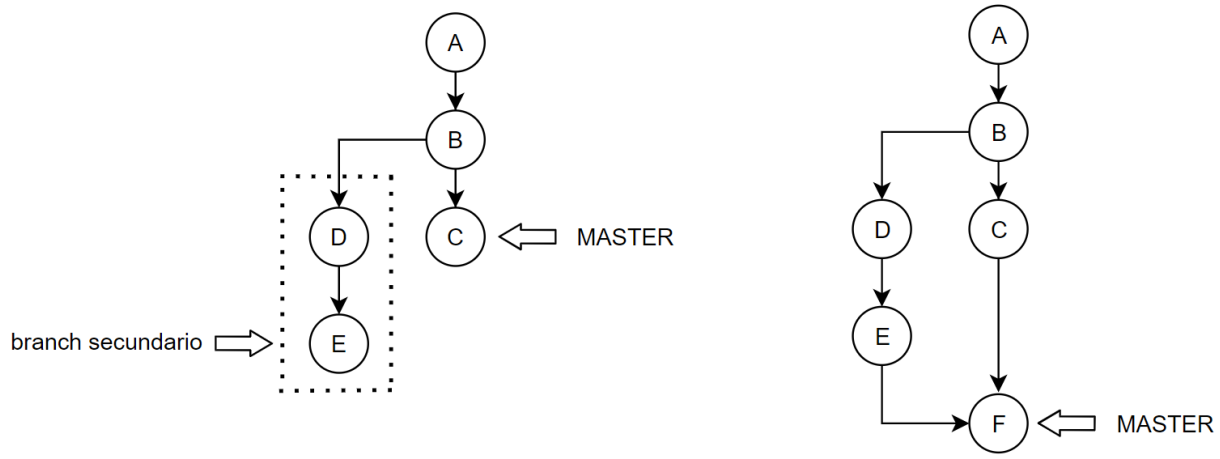


Una vez se desarrolló la característica completamente, se implementará al master. Por lo tanto, la operación *merge* permitirá unir el branch secundario, el cual aloja la última versión del commit D, con el master, el cual apunta hacia el commit B. En este caso, se observa que los cambios siguen una trayectoria línea, por lo que en Git se podrá hacer fast forward merge, el cual simplemente hará que el master apunte al commit D. En cambio, en Github se crea un pull request, el cual creará un nuevo commit del merge cuando sea autorizado, sin embargo, el master también apuntará hacia este nuevo commit, el cual representa la versión más reciente del proyecto

Caso 2:

En caso de que se esté trabajando una mejora en un nuevo branch y al commit base (B) que se usó para crear dicha característica, se le modifique, de manera que el master siga apuntando hacia sus actualizaciones más recientes, se realizará una operación diferente. Para unir al branch

secundario y el master se empleará un 3-way merge, para el cual se deberán resolver los conflictos entre ambos, en caso de que los haya, para poder formar un nuevo commit del merge F, al cual estará apuntando el master, tal como se muestra en la imagen [6].



6) ¿Qué es una Prueba Unitaria o Unittest en el contexto de desarrollo de software?

Una prueba unitaria corresponde a verificar que los componentes unitarios de un software funcionen de la manera en que fueron diseñados. Estos componentes o unidades son la parte más pequeña y comprobable de un software, como lo son las funciones u otros procedimientos. Unittest es una prueba de software de primer nivel, la cual se enfoca en los casos en donde se impacte en mayor parte el comportamiento del código. Estas pruebas son desempeñadas con el fin aumentar la confiabilidad al cambiar o darle mantenimiento a un código, además de ayudar en gran parte a reducir costos y tiempo [7].

7) Bajo el contexto de pytest. ¿Qué es un “assert”?

Los asserts son empleados en las pruebas para verificar excepciones o valores, de manera que, se comprueba que la unidad o componente del software retorne el valor esperado. En el caso que no se cumpla la condición esperada, se detendrá el método de la prueba y no se ejecutará el código restante, por lo que indicará el error y continuará con la siguiente prueba. Es decir, consiste en una funcionalidad de Pytest que evalúa los componentes del código, comprobando que se retorne un valor correcto, por lo que se deben desarrollar de forma tal, que cumplan con el propósito de las pruebas unitarias [8].

8) ¿Qué es Flake 8?

Flake 8 es una serie de comandos creados para examinar la calidad de sintaxis de un código de programación, el cual muestra una serie de recomendaciones para poder limpiar el código, de manera que se logre una consistencia en el formato de los códigos de Python. Gracias a esta herramienta, se pueden prevenir errores de sintaxis, por lo que ayudará a ahorrar tiempo, a aumentar el nivel de confianza en lo que se está haciendo y, además, se podrán cumplir los estándares de codificación [9].

Referencias

[1] Bitbucket, "¿Qué es el control de versiones?", s.f. [En línea]. Disponible en: https://www.atlassian.com/es/git/tutorials/what-is-version-control [Consultado el 11 de agt., 2021].

[2] Gustavo B., "¿Qué es GitHub y cómo usarlo?", 2021. [En línea]. Disponible en: https://www.hostinger.es/tutoriales/que-es-github [Consultado el 11 de agt., 2021].

[3] S. Chacon y B. Straub, "Pro Git", 2021. [En línea]. Disponible en: https://git-scm.com/book/en/v2 [Consultado el 11 de agt., 2021].

[4] Bitbucket, "git commit", s.f. [En línea]. Disponible en: https://www.atlassian.com/es/git/tutorials/saving-changes/git-commit [Consultado el 11 de agt., 2021].

[5] I. Ribas y A. Errasti, "Seleccionando commits: Cherry-pick", 2017. [En línea]. Disponible en: https://www.runroom.com/realworld/seleccionando-commits-cherry-pick [Consultado el 11 de agt., 2021].

[6] W3docs, "Git merge", 2019. [En línea]. Disponible en: https://www.w3docs.com/learn-git/git-merge.html [Consultado el 11 de agt., 2021].

[7] Software Testing Fundamentals, "Unit Testing", 2020. [En línea]. Disponible en: https://softwaretestingfundamentals.com/unit-testing/ [Consultado el 11 de agt., 2021].

[8] Pytest, **"**How to write and report assertions in tests**",** 2021. [En línea]. Disponible en: https://docs.pytest.org/en/latest/how-to/assert.html [Consultado el 11 de agt., 2021].

[9] Flake8, **"**Flake8 man page**",** 2021. [En línea]. Disponible en: https://flake8.pycqa.org/en/latest/manpage.html#synopsis [Consultado el 11 de agt., 2021].