

PEC 3: Control de Versiones y Documentación

Paula Andrea Figueroa

14 de diciembre de 2015

1. Introducción

Para la primera sección de control de versiones se trabajó con git y su complemento github, con el fin de proveer un sistema controlador de versiones, en este caso git, y un repositorio remoto como github. A través de la primera sección se describe todo el procedimiento relacionado a la instalación, uso y conexión de git con github y su posterior uso en equipos de trabajo.

Para la segunda sección se trabajó la documentación del código filtros.c y filtros.h a través del software doxygen, que provee etiquetas que permiten documentar el software describiendo las funciones que realiza, los parámetros de entrada y salida.

Para la realización de latex se realizó las instalaciones del paquete completo de latex para Ubuntu 14.10 y se empezó a realizar el documento a través del manual de latex en línea.

Por último en las conclusiones se muestra los resultados obtenidos, aprendizaje y dificultades encontradas.

2. Control de Versiones

Para esta práctica se trabajará con git y con github. Git es el sistema de control de versiones mientras github es una red social que permite almacenar los proyectos opensource en línea. Es decir git permite manipular los proyectos que se encuentran almacenados en github.

Para poder tener acceso a los proyectos debe tener una cuenta en github.

También se pueden mencionar secciones de la misma forma: ver sección

2.1. Instalar GIT y GITHUB

1. Se instala git con el comando: `apt-get install git`.
2. Con `git --version` para ver la versión instalada.
3. Obtener una cuenta en github ingresando a <https://www.github.com>

```
root@Bernie:/home/paula# git --version
git version 1.9.1
```

Figura 1: Version de git

4. Completar el siguiente formulario:

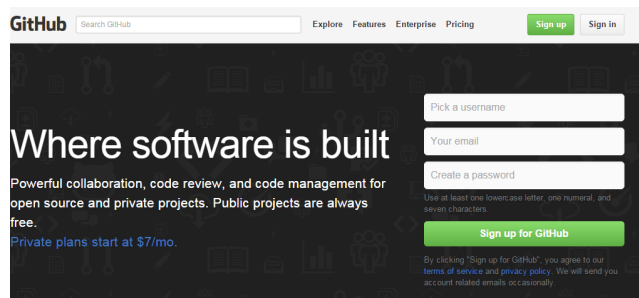


Figura 2: Formulario de Registro de Github

5. Verificar la cuenta ingresando a su correo electrónico

2.2. Configurar el repositorio

1. Se configura un usuario para identificarse al hacer los committs en git con:
git config --global user.name "pfigueroap"
git config --global user.email "pfigueroap@uoc.edu"

```
paula@Bernies:~/pec3desarrollo$ git config --global user.mail "pfigueroap@uoc.edu"
" I
paula@Bernies:~/pec3desarrollo$ git config --global user.name "pfigueroapuoc"
```

Figura 3: Configuración de usuario en git

2. Se inicializan el repositorio con: git init

```

paula@Bernies:~/pec3desarrollo$ git init
Initialized empty Git repository in /home/paula/pec3desarrollo/.git/
paula@Bernies:~/pec3desarrollo$ mkdir libsrc
paula@Bernies:~/pec3desarrollo$ cd libsrc
paula@Bernies:~/pec3desarrollo/libsrc$ cp /home/paula/libsrc/filtros.c /home/pau
la/pec3desarrollo/libsrc/
paula@Bernies:~/pec3desarrollo/libsrc$ cp /home/paula/libsrc/filtros.h /home/pau
la/pec3desarrollo/libsrc/
paula@Bernies:~/pec3desarrollo/libsrc$ ls -al
total 16
drwxrwxr-x 2 paula paula 4096 dic  5 23:53 .
drwxrwxr-x 4 paula paula 4096 dic  5 23:52 ..
-rw-rw-r-- 1 paula paula 1325 dic  5 23:53 filtros.c
-rw-rw-r-- 1 paula paula  102 dic  5 23:53 filtros.h
paula@Bernies:~/pec3desarrollo/libsrc$

```

Figura 4: Copiar filtro.c y filtro.h

3. Se copian los dos archivos filtros.c y filtros.h en la carpeta /libsrc y se verifica que el .git en el repositorio se encuentre creado

```

paula@Bernies:~/pec3desarrollo$ ls -al
total 20
drwxrwxr-x 4 paula paula 4096 dic  5 23:56 .
drwxr-xr-x 20 paula paula 4096 dic  5 23:52 ..
drwxrwxr-x 7 paula paula 4096 dic  5 23:52 .git
drwxrwxr-x 2 paula paula 4096 dic  5 23:56 libsrc
-rw-rw-r-- 1 paula paula  103 dic  5 23:56 README

```

Figura 5: Verificación de archivos en el repositorio

4. Se crea un fichero README en el directorio principal

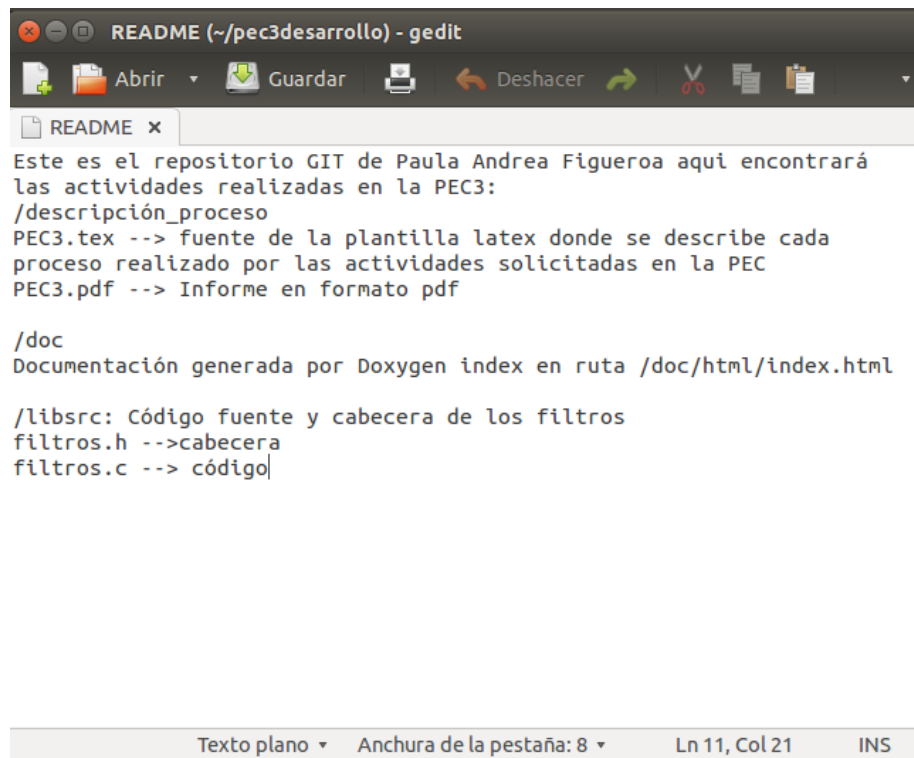


Figura 6: Creación de fichero README

5. Se van añadir al repositorio con los comandos:

Para añadir uno a uno:

```
git add README
```

```
git add /libsrc/filtros.c
```

```
git add /libsrc/filtros.h
```

Para añadir todo de una vez:

```
con git add -all
```

```

paula@Bernies:~/pec3desarrollo$ git add --all
paula@Bernies:~/pec3desarrollo$ git status
En la rama master

Commit inicial

Cambios para hacer commit:
  (use «git rm --cached <archivo>...» para eliminar stage)

    new file:   README
    new file:   libsrc/README
    new file:   libsrc/filtros.c
    new file:   libsrc/filtros.h

```

Figura 7: Adicionar ficheros al git

6. Se verifica el estado de git con el comando: git status

```

paula@Bernies:~/libsrc$ git status
En la rama master

Commit inicial

Cambios para hacer commit:
  (use «git rm --cached <archivo>...» para eliminar stage)

    new file:   README
    new file:   filtros.c
    new file:   filtros.h

```

Figura 8: Estados de ficheros en git

7. Hacer commit

```

paula@Bernies:~/pec3desarrollo$ git commit -m "Inicializa el proyecto y se añade REAM
DE y filtro.c y filtro.h dentro de /libsrc"
[master (root-commit) de6980b] Inicializa el proyecto y se añade README y filtro.c y
filtro.h dentro de /libsrc
4 files changed, 91 insertions(+)
create mode 100644 README
create mode 100644 libsrc/README
create mode 100644 libsrc/filtros.c
create mode 100644 libsrc/filtros.h

```

Figura 9: Estados de ficheros en git

8. Con el comando git log se verifica el estado del commit

```
paula@Bernies:~/pec3desarrollo$ git log
commit de6980bb895f5156cf817f683045c987dff9d358
Author: pfigueroap <pfigueroap@uoc.edu>
Date: Sun Dec 6 00:02:07 2015 -0500

Inicializa el proyecto y se añade REAMDE y filtro.c y filtro.h dentro de /libsrc
```

Figura 10: Log del commit

2.3. Subir al repositorio remoto

1. Agregar el repositorio a github en www.github.com



Figura 11: Crear un repositorio en github

2. Se crea el repositorio llenando el formulario:

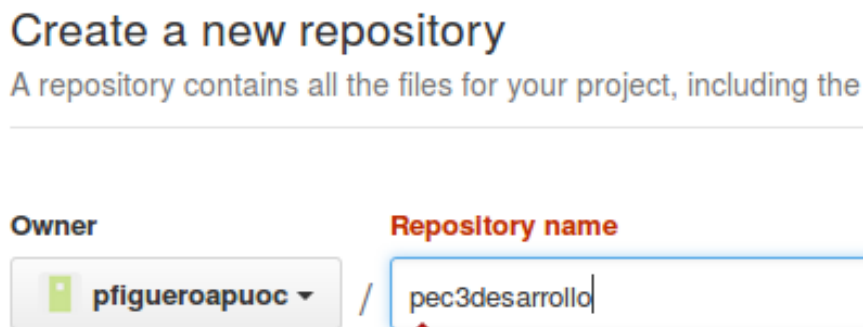
The image shows the 'Create a new repository' form on GitHub. The title is 'Create a new repository' with a subtitle 'A repository contains all the files for your project, including the'. Below this, there are two input fields. The first is labeled 'Owner' and contains a dropdown menu with 'pfigueroapuoc' selected. The second is labeled 'Repository name' and contains the text 'pec3desarrollo'.

Figura 12: Crear un repositorio en github

3. Con el siguiente comando agregamos el repositorio que ya existe, con esta linea se le indica a git cual el repositorio remoto donde se alojaran los archivos:

```
git remote add origin https://github.com/pfigueroapuoc/libsrc.git
```

4. Se agrega los archivos al repositorio remoto con el comando:

`git push -u origin master`

El cual solicita el login y contraseña de Github

```
paula@Bernies:~/pec3desarrollo$ git remote add origin https://github.com/pfigueroapuoc/pec3desarrollo.git
paula@Bernies:~/pec3desarrollo$ git push -u origin master
Username for 'https://github.com': pfigueroapuoc
Password for 'https://pfigueroapuoc@github.com':
Counting objects: 142, done.
Compressing objects: 100% (142/142), done.
Writing objects: 100% (142/142), 3.47 MiB | 137.00 KiB/s, done.
Total 142 (delta 23), reused 0 (delta 0)
To https://github.com/pfigueroapuoc/pec3desarrollo.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.
paula@Bernies:~/pec3desarrollo$
```

Figura 13: Indicando a git el repositorio remoto y trasladando los archivos al mismo

5. De este modo el proyecto ya se encuentra alojado remotamente en github

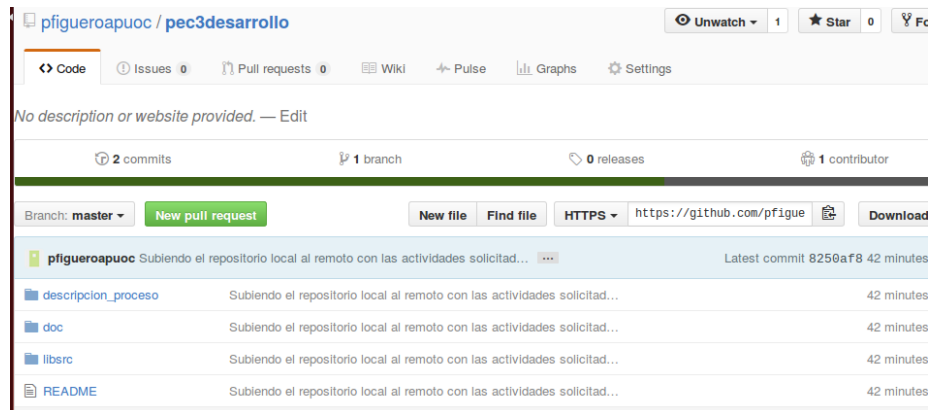


Figura 14: Repositorio en github

2.4. Acceder a un repositorio y obtener copias locales

1. En caso de que otra persona o el mismo creador desee obtener una copia local del repositorio deberá ingresar el comando

`git clone https://github.com/pfigueroapuoc/pec3desarrollo.git`

```

paula@Bernies:~$ mkdir prueba
paula@Bernies:~$ cd prueba/
paula@Bernies:~/prueba$ git clone https://github.com/pfigueroapuoc/pec3desarrollo.git
Clonar en «pec3desarrollo»...
remote: Counting objects: 142, done.
remote: Compressing objects: 100% (119/119), done.
remote: Total 142 (delta 23), reused 142 (delta 23), pack-reused 0
Receiving objects: 100% (142/142), 3.47 MiB | 628.00 KiB/s, done.
Resolving deltas: 100% (23/23), done.
Checking connectivity... hecho.
paula@Bernies:~/prueba$ ls
pec3desarrollo

```

Figura 15: Obtener una copia local de un repositorio

2. Para participar en el repositorio desde github simplemente es buscar el repositorio

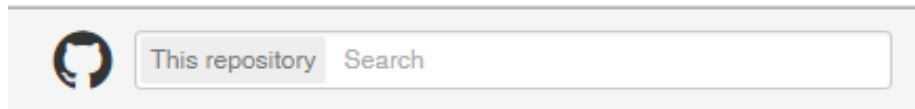


Figura 16: Obtener una copia local de un repositorio

3. Y una vez encontrado darle en fork

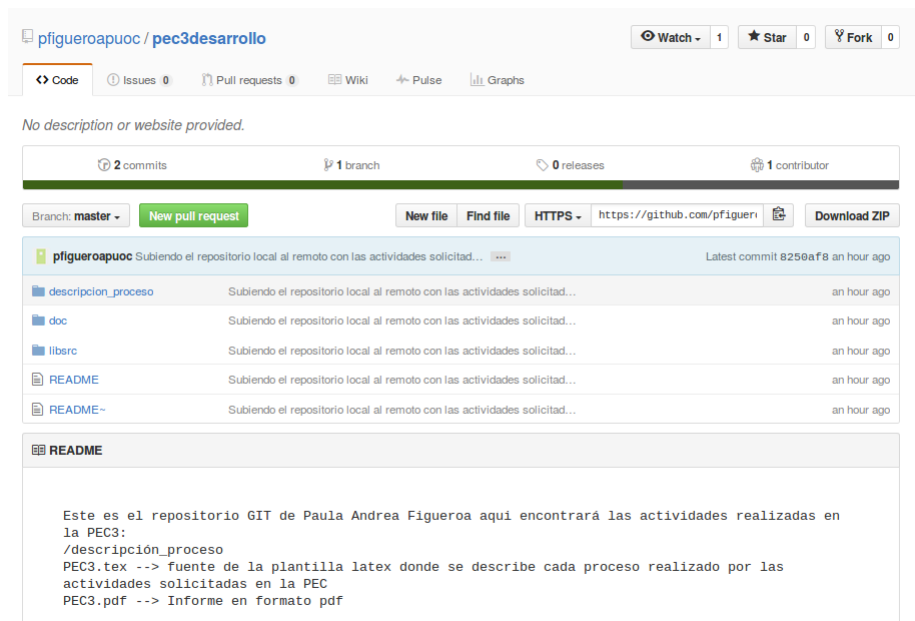


Figura 17: Obtener una copia local de un repositorio

4. Se verifica el email y ya se puede acceder al repositorio.

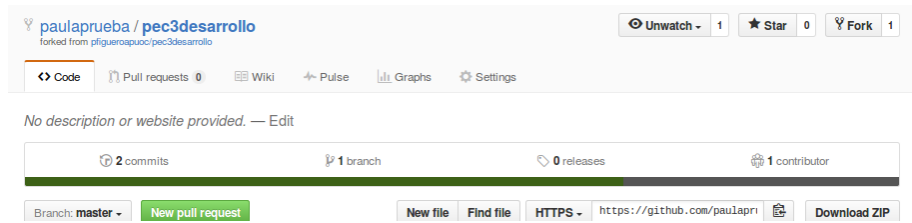


Figura 18: Obtener una copia local de un repositorio

3. Documentación del Código

Para la documentar el código se uso el software doxygen con su herramienta gráfica doxygen-gui.

1. Se instala doxygen y doxygen-gui con el comando: `sudo apt-get install doxygen doxygen-gui`
2. Con un editor de texto se abre cada fichero `filtros.c` y `filtros.h`

```
/**
 * @file filtros.h
 * @brief Cabeceras de las filtros ofrecidos en filtros.c
 * Se listas los filtros principales:
 * fs_head, fs_wc, fs_nl y fs_out
 * Cada filtro tiene como entrada un archivo representado por la variable (fd), y fs_out tiene un parámetro adicional que es un entero el cual indica la columna a imprimir
 * @author Paula Figueras.
 * @date 04/12/2019.
 */

void fs_head( int fd );
void fs_wc( int fd );
void fs_nl( int fd );
void fs_out( int fd, int col );
```

Figura 19: Documentación filtros.h

Cada etiqueta permite:

- @file: muestra el fichero que se esta documentando filtro.h
- @brief muestra una descripción corta
- La etiqueta @ author muestra el autor del código
- Para escribir descripciones más largas separadas en parrafos se deben dejar espacios entre línea y línea.
- @param indica los parametros de entrada o salida
- @see permite crear enlaces a otras funciones
- verbatim y endverbatim permite crear un ejemplo

- En la siguiente imagen se muestra como comentar bloques de código

```
int aux, i; /* aux, i: son variables tipo entero que representan contadores i es para ir recorriendo cada num ingresado mientras que aux controla la memoria asignada al apuntador, ambas inicializan en 1.*/
char aux2; /* aux2: puntero tipo char */
```

Figura 20: Comentar bloques de código

```
int count = 0; /*<!-- count: variable que se declara dentro de la función, tipo entero, es un contador que permite determinar la cantidad de lineas que se tienen en un texto, cada vez que dentro del while se encuentra un salto de linea se incrementa el contador, una vez llega a 3 se sale del ciclo.*/
char c; /*<!-- c: es una variable que se declara dentro de la función, tipo caracter en el cual se irá almacenando el caracter que se lee en el ciclo while */
```

Figura 21: Comentar bloques de código

3. En filtros.h se realiza la codificación necesaria para modificar el mainpage o index que mostrará el html

```
/*<!-- Mainpage Control de Versiones y Documentación
 * \section control Control de versiones
 *
 * \section intro_sec Introducción
 *
 * En la presente documentación encontrará información relacionada con las funciones filtro.c y filtro.h desarrolladas en c, para realizar esta documentación se usó Doxygen.
 *
 * Mientras que en la segunda encontrará las cabeceras de cada una de estas funciones.
 *
 * Por cada archivo (filtros.h y filtros.c) de la copia local del depósito se encuentran los comentarios de cabecera en el que, usando siempre etiquetas de Doxygen, se indica, de forma resumida y
 * también de forma detallada, qué se encuentra en el archivo, el autor y la fecha en la que se genera la documentación.
 *
 * La documentación se generó en formato html en un directorio llamdo doc.
 *
 * La idea de esta sección de la PEC es desarrollar habilidades en la documentación de un producto de desarrollo de software
 *
 * Finalmente toda esta información se encuentra almacenada en el repositorio local libero y adicionalmente en el repositorio remoto en github.
 */
```

Figura 22: Main Page filtros.h

La etiqueta `section` permite añadir secciones en la documentación

4. A través del gui se puede crear la documentación para ello se debe configurar en la pestaña Wizard topic project el nombre del proyecto, el directorio fuente, es decir donde se encuentra filtros.c y filtros.h y hacia que directorio irá lo creado por doxygen en este caso será un directo doc.

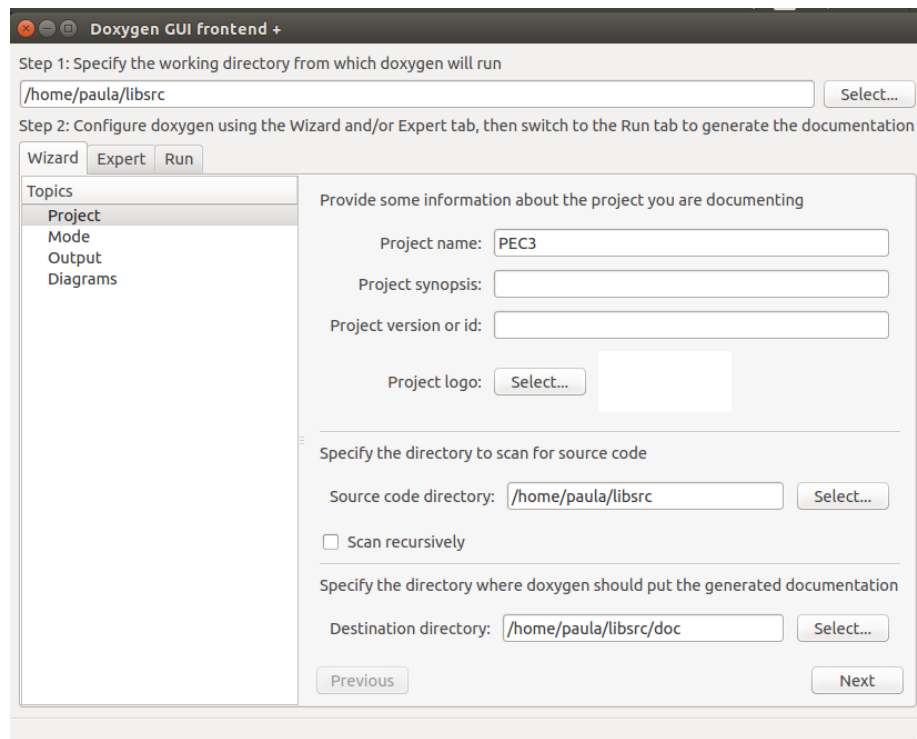


Figura 23: Configuración de Doxygen GUI

5. En el topic mode seleccione el proyecto que desea documentar, en este caso es c

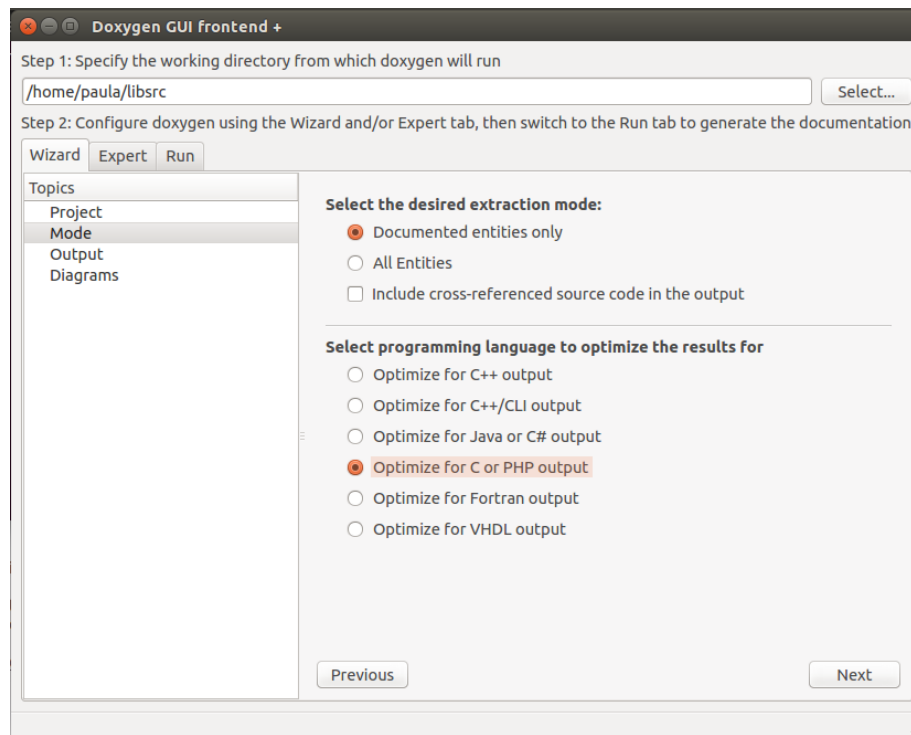


Figura 24: Seleccionando tipo de proyecto

6. En el topic Output seleccione la salida que para este caso es html

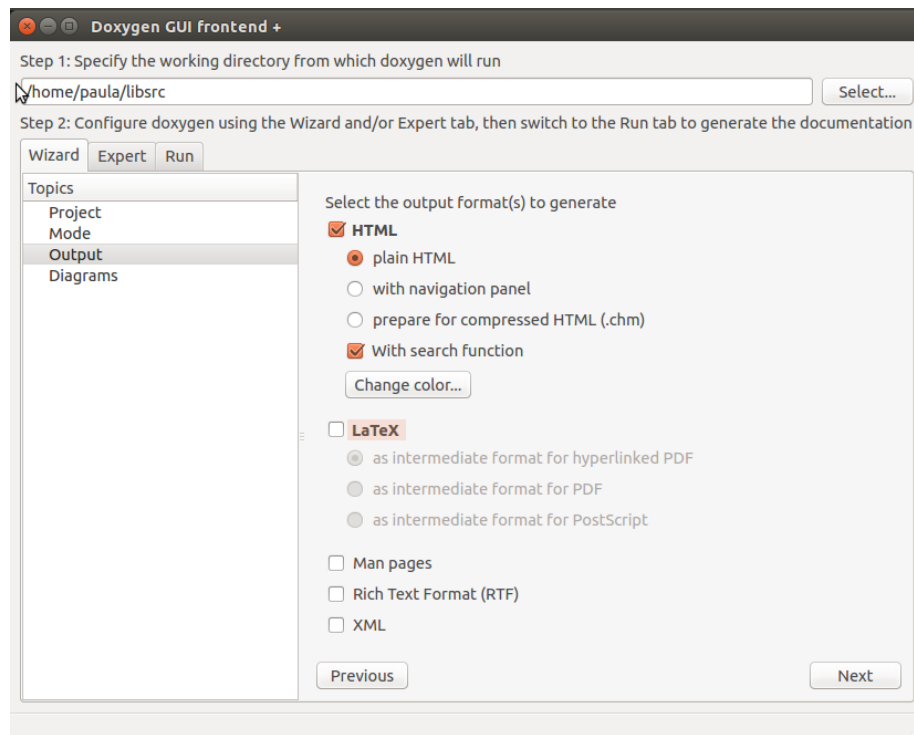


Figura 25: Salida de la documentación en HTML

7. Por último ir a la pestaña run y ejecutar el proyecto, aquí también podrá visualizar como quedará el html

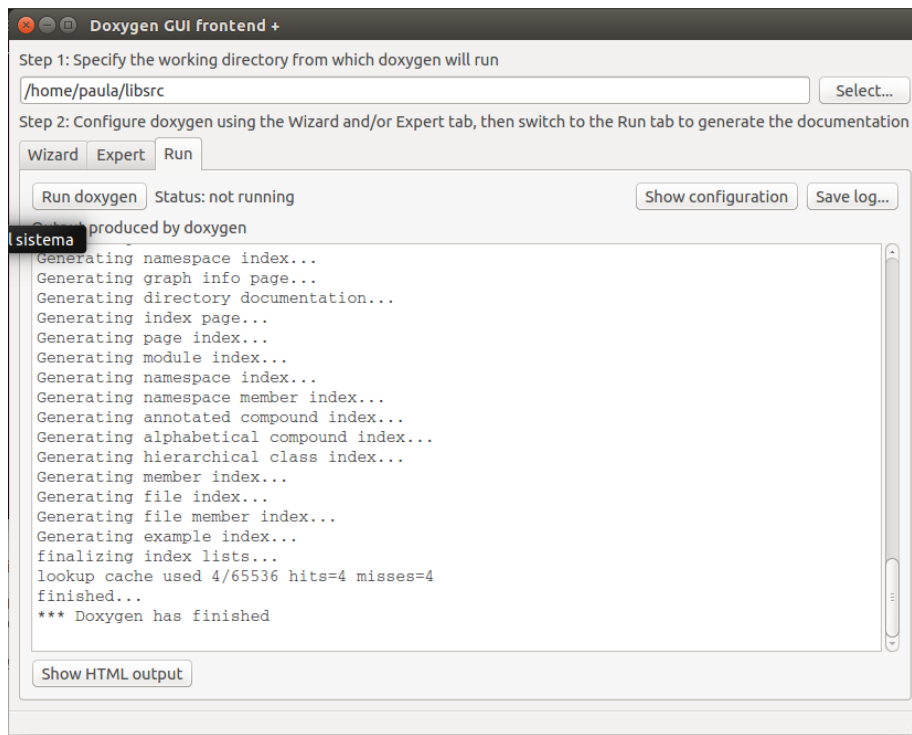


Figura 26: Ejecutar el Proyecto

8. AL final se puede obtener algo como:

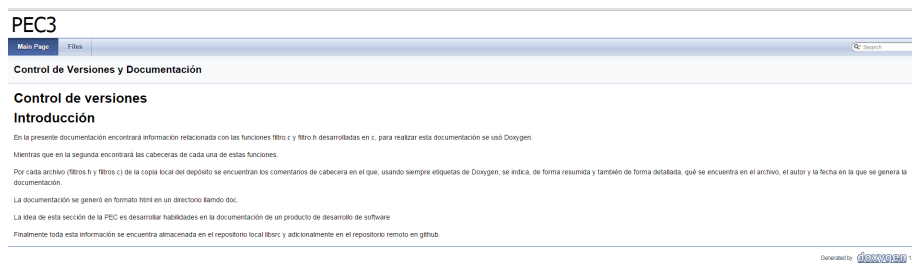


Figura 27: Documentación en HTML



Figura 28: Documentación en HTML



Figura 29: Documentación en HTML

9. Se hace el commit al github

```

paula@Bernies:~/libsrc$ git push -u origin master
Username for 'https://github.com': pfigueroapuoc
Password for 'https://pfigueroapuoc@github.com':
Counting objects: 79, done.
Compressing objects: 100% (78/78), done.
Writing objects: 100% (78/78), 132.67 KiB | 0 bytes/s, done.
Total 78 (delta 21), reused 0 (delta 0)
To https://github.com/pfigueroapuoc/libsrc.git
   eab2de5..9a74aed  master -> master
Branch master set up to track remote branch master from origin.

```

Figura 30: Agregando los cambios al repositorio

<p>pfigueroapuoc Actualiza con documentación filtros.c y filtros.h y se sube la carpet... ...</p>		Latest commit 9a74aed 31 minutes ago
Archivos	Actualiza con documentación filtros.c y filtros.h y se sube la carpet...	31 minutes ago
doc	Actualiza con documentación filtros.c y filtros.h y se sube la carpet...	31 minutes ago
README	Inicializa el proyecto y se añade README filtro.c y filtros.h	12 hours ago
filtros.c	Actualiza con documentación filtros.c y filtros.h y se sube la carpet...	31 minutes ago
filtros.h	Actualiza con documentación filtros.c y filtros.h y se sube la carpet...	31 minutes ago

README
<p>Este es el repositorio GIT de Paula Andrea Figueroa aqui encontrará dos archivos:</p> <pre> filtros.c filtros.h </pre>

Figura 31: Agregando los cambios al repositorio

Al final con otra configuración se pueden obtener los siguientes resultados como se muestra en la imagen:

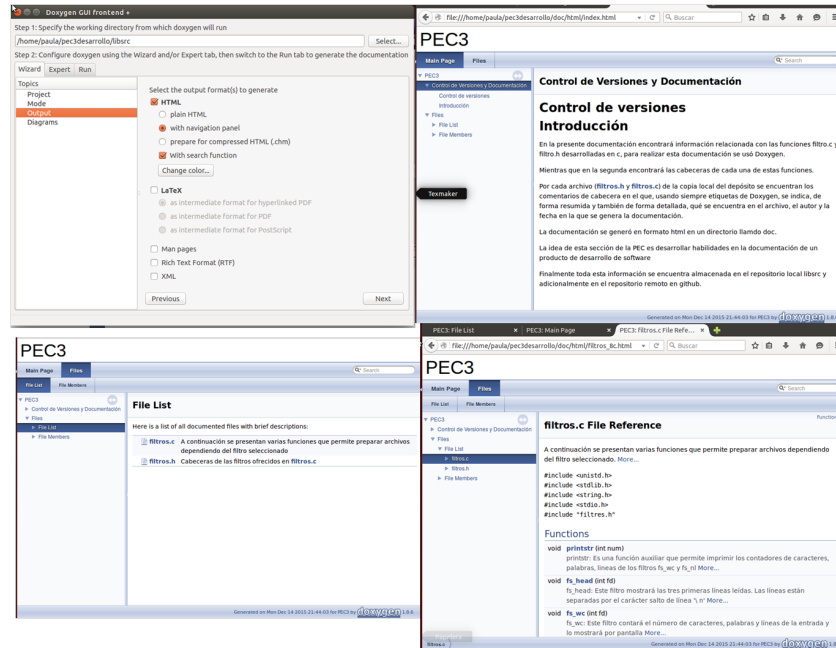


Figura 32: Configuración final de la documentación con Doxygen

4. Documento Latex en el repositorio

1. Se crea una carpeta llamada `descripcion` _ proceso en el repositorio local y en ella se copia el TeX, las imágenes y el documento generado en pdf.

```
paula@Bernies:~/libsrc/descripcion_proceso$ ls -al
total 860
drwxrwxr-x 3 paula paula 4096 dic 6 00:27 .
drwxrwxr-x 6 paula paula 4096 dic 6 00:27 ..
drwxrwxr-x 2 paula paula 4096 dic 6 00:18 imagenes
-rw-rw-r-- 1 paula paula 845978 dic 6 00:26 PEC3.pdf
-rw-r--r-- 1 paula paula 17047 dic 6 00:26 PEC3.tex
```

Figura 33: Copiando a `descripcion` _ proceso

2. Posteriormente se hace `adiciona` y se realiza un `commit` para incluir esta información y se aplica un `push` para montarlo al repositorio

```

create mode 100644 descripcion_proceso/imagenes/uso.png
paula@Bernies:~/libsrc$ git push -u origin master
Username for 'https://github.com': pfigueroapuoc
Password for 'https://pfigueroapuoc@github.com':
Counting objects: 41, done.
Compressing objects: 100% (40/40), done.
Writing objects: 100% (40/40), 1.73 MiB | 148.00 KiB/s, done.
Total 40 (delta 1), reused 0 (delta 0)
To https://github.com/pfigueroapuoc/libsrc.git
 9a74aed..c33eb31 master -> master
Branch master set up to track remote branch master from origin.

```

Figura 34: Copiando a descripcion _ proceso

3. Se actualiza el repositorio

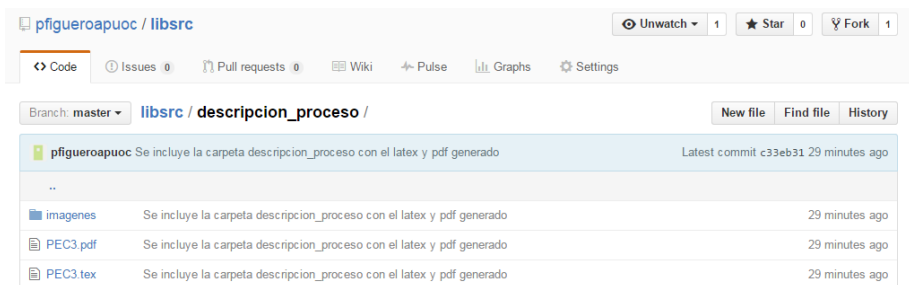


Figura 35: Copiando a descripcion _ proceso

5. Conclusiones

Después de haber realizado la práctica se puede concluir que:
Para el caso del controlador de versiones:

- Los controladores de versiones son muy importantes en cualquier proyecto de desarrollo de software pues facilita tener a disposición cualquier versión y controlar los cambios que se han realizado en el proyecto.
- Git es un controlador de versiones fácil de instalar y manejar, aunque se realiza por consola los pasos prácticamente son muy intuitivos y se obtienen resultados muy claros.
- La integración con github ofrece facilidad de publicar proyectos de desarrollo disponibles para toda la comunidad.
- El hecho que para poder configurar los usuarios permitidos en los proyectos Github requiera pagos implica un inconveniente en el caso que se desee restringir o controlar los cambios que se realizan en el proyecto y las copias

y ventas que se hagan con el, es recomendable marcar cada componente del proyecto con la licencia que ayude a protegerlo.

- Git y github tienen ciertos inconvenientes con las modificaciones al tiempo que realicen los colaboradores pero ofrece soluciones de permitir guardar los cambios detectados aunque el desarrollador deba modificarlo manualmente.

Para el caso de la documentación:

- Es fácil obtener un manual/documentación estructurado en html de un desarrollo y siendo así es fácil incluirla en cualquier desarrollo.
- Al igual de latex se debe tener conocimiento de las etiquetas y del lenguaje manejado, aunque se encuentra muy bien documentado en su página principal.
- La parte gráfica facilita la compilación aunque también se puede realizar por consola.

Para el caso de latex:

- Es estable, multiplataforma, es gratuito y permite estructurar fácilmente los documentos
- Se debe tener especial cuidado en la forma de poner las etiquetas, pues una etiqueta mal cerrada, un parámetro mal escrito implica un error en la compilación y no será posible ver el documento.
- La curva de aprendizaje es bastante amplia, al no ser tan gráfico como los demás procesadores presenta un grado de dificultad.
- Hasta que no se compile no se tiene una visión del documento, lo que representa una dificultad y requiere de tiempo en caso que no se tenga habilidad se demorará más en generar un documento.
- Los gráficos no se insertan tan fácilmente, presenté dificultad respecto a ubicarlos donde quería en este caso debería tener en el `\beginfigure` el parámetro H entre corchetes.