



ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

“PROPUESTA DE UN MÉTODO PARA EL SEGUIMIENTO DE PROYECTOS DE FIN DE CURSO EN MATERIAS DE LENGUAJES DE PROGRAMACIÓN UTILIZANDO GITHUB EN CARRERAS DE INGENIERÍA ELECTRÓNICA.”

AUTOR:

SALAZAR ZÁRATE JAVIER EDUARDO

Proyecto de investigación presentado ante el Instituto de Posgrado y Educación Continua de la
ESPOCH, como requisito parcial para la obtención del grado de:

MAGISTER EN INFORMÁTICA EDUCATIVA

Riobamba – Ecuador

Enero 2016



ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

CERTIFICACIÓN:

EL TRIBUNAL DE TRABAJO DE TITULACION CERTIFICA QUE:

El Proyecto de Investigación, titulado “PROPUESTA DE UN MÉTODO PARA EL SEGUIMIENTO DE PROYECTOS DE FIN DE CURSO EN MATERIAS DE LENGUAJES DE PROGRAMACIÓN UTILIZANDO GITHUB EN CARRERAS DE INGENIERÍA ELECTRÓNICA”, de responsabilidad del Ing. Javier Eduardo Salazar Zárate, ha sido prolijamente revisado y se autoriza su presentación.

Tribunal:

Dr. Juan Vargas Guambo, M.Sc.

PRESIDENTE

FIRMA

Ing. Blanca Hidalgo Ponce, M. Sc.

DIRECTORA

FIRMA

Dra. Narcisa Salazar Álvarez, M. Sc.

MIEMBRO

FIRMA

Ing. Byron Vaca Barahona, PhD.

MIEMBRO

FIRMA

COORDINADOR SISBIB ESPOCH

FIRMA

Riobamba, enero 2016

DERECHOS INTELECTUALES

Yo, **JAVIER EDUARDO SALAZAR ZÁRATE**, declaro que soy responsable de las ideas, doctrinas y resultados expuestos en el presente Proyecto de Investigación, y que el patrimonio intelectual generado por el mismo pertenece exclusivamente a la Escuela Superior Politécnica de Chimborazo.

FIRMA

060309374-1

TABLA DE CONTENIDO

CONTENIDO	Páginas
RESUMEN	xiv
SUMMARY	xv
CAPÍTULO I	
1. INTRODUCCIÓN	1
1.1. Formulación del problema	2
1.2. Justificación	4
1.3. Objetivos	6
<i>1.3.1. General</i>	<i>6</i>
<i>1.3.2. Específicos</i>	<i>6</i>
1.4. Hipótesis	6
CAPÍTULO II	
2. MARCO DE REFERENCIA	8
2.1. Definición de Método	8
2.2. Tipos de método	9
2.3. Sistemas de control de versiones	12
<i>2.3.1. Sistemas de control de versiones local</i>	<i>12</i>
<i>2.3.2. Sistemas de control de versiones centralizado</i>	<i>13</i>
<i>2.3.3. Sistemas de control de versiones distribuido</i>	<i>13</i>
2.4. Git	14
<i>2.4.1. Funcionamiento</i>	<i>14</i>
<i>2.4.2. Instalación</i>	<i>18</i>

2.4.3.	<i>Conceptos en Git</i>	19
2.4.4.	<i>Comandos</i>	19
2.5.	Github	20
2.5.1.	<i>La popularidad de GitHub</i>	21
2.5.2.	<i>Más allá del versionamiento de archivos</i>	24
2.5.3.	<i>Características</i>	25
2.5.4.	<i>Permisos para usuarios</i>	34
2.6.	GitHub en la educación	35

CAPÍTULO III

3.	DISEÑO DE INVESTIGACIÓN	40
3.1.	Diseño de la investigación	40
3.2.	Tipo de investigación	40
3.3.	Población	41
3.4.	Muestra	41
3.5.	Métodos	41
3.6.	Técnicas	42
3.7.	Instrumentos de evaluación	42
3.8.	Aplicación del método	42
3.9.	Procedimientos	43
3.10.	VARIABLES e indicadores	44
3.11.	Análisis de las variables	47
3.11.1.	<i>Indicadores de la variable independiente</i>	47
3.11.2.	<i>Indicadores de la variable dependiente</i>	47

CAPÍTULO IV

4.	RESULTADOS Y DISCUSIÓN	49
----	-------------------------------------	----

4.1.	Procedimiento general	49
4.2.	Presentación de resultados	50
4.3.	Demostración de la hipótesis	62
4.3.1.	<i>Planteamiento</i>	62
4.3.2.	<i>Población</i>	63
4.3.3.	<i>Selección del nivel de significación</i>	63
4.3.4.	<i>Descripción de la muestra</i>	63
4.3.5.	<i>Especificación del estadístico</i>	64
4.4.	Comprobación	65
4.5.	Conclusión de la hipótesis	69

CAPÍTULO V

5.	PROPUESTA	73
5.1.	Introducción	73
5.2.	Objetivos	73
5.3.	Descripción de escenario	74
5.4.	Definición del método	74
5.5.	Lineamientos	74
5.6.	Estructura del método	75
5.6.1.	<i>Roles</i>	75
5.6.2.	<i>Artefactos</i>	75
5.6.2.1.	<i>ARTEFACTO 01: Definición de proyecto</i>	75
5.6.2.2.	<i>ARTEFACTO 02: Manual de GitHub</i>	76
5.6.2.3.	<i>ARTEFACTO 03: Definición de equipo</i>	77
5.6.2.4.	<i>ARTEFACTO 04: Retroalimentación</i>	77
5.6.2.5.	<i>ARTEFACTO 05: Registro de avance</i>	78

5.6.2.6. ARTEFACTO 06: Cuestionario final.....	80
5.6.2.7. ARTEFACTO 07: Rúbrica de evaluación.....	81
5.6.3. Actividades.....	82
5.6.4. Procedimiento	82
5.7. Ventajas y Desventajas	85
CONCLUSIONES	87
RECOMENDACIONES	90
BIBLIOGRAFÍA	
ANEXOS	

INDICE DE TABLAS

Tabla 1-2:	Los tipos de investigación y el método.	9
Tabla 2-2:	Comandos relevantes en Git.....	19
Tabla 3-2:	Inferencias sociales basadas en la actividad en GitHub.....	22
Tabla 4-2:	Top 3 de proyectos en GitHub por lenguaje de programación.	23
Tabla 1-3:	Operacionalización conceptual de variables	44
Tabla 2-3:	Operacionalización metodológica de variables.....	45
Tabla 1-4:	Grupos muestrales de estudiantes seleccionados para el estudio.....	64
Tabla 2-4:	Grados de calidad de proyectos de los estudiantes del periodo 2014 ..	66
Tabla 3-4:	Grados de calidad de proyectos de los estudiantes del periodo 2014 ..	66
Tabla 4-4:	Valores de la estadística descriptiva para los grupos muestrales	67
Tabla 5-4:	Contraste para la diferencia de medias de los dos grupos muestrales..	69
Tabla 1-5:	Ventajas y desventajas del método MESEPP	85

INDICE DE FIGURAS

Figura 1-2:	Lista de issues del proyecto bootstrap en GitHub.	26
Figura 2-2:	Wiki del proyecto bootstrap en GitHub.....	27
Figura 3-2:	Vista del pulso de actividades del proyecto bootstrap en GitHub.....	28
Figura 4-2:	Vista de contribuciones del proyecto bootstrap en GitHub.	29
Figura 5-2:	Vista de commits del proyecto bootstrap en GitHub.....	30
Figura 6-2:	Vista de frecuencia de código del proyecto bootstrap en GitHub.	31
Figura 7-2:	Vista Push Card del proyecto bootstrap en GitHub.....	32
Figura 8-2:	Vista de milestones del proyecto bootstrap en GitHub.	32
Figura 9-2:	Vista de labels del proyecto bootstrap en GitHub.	33

INDICE DE GRÁFICOS

Gráfico 1-2:	Sistema de control.....	12
Gráfico 2-2:	Sistema de control de versiones centralizado	13
Gráfico 3-2:	Sistema de control de versiones distribuido.....	14
Gráfico 4-2:	Versionamiento basado en cambios.....	15
Gráfico 5-2:	Versionamiento basado en archivos.....	16
Gráfico 6-2:	Principales secciones de un proyecto en Git.....	17
Gráfico 7-2:	Estado de los archivos en su tiempo de vida en Git.....	18
Gráfico 8-2:	Tareas de LMSs potenciadas por GitHub	38
Gráfico 1-4:	Total de calidad en proyectos periodo 2014 y 2015	50
Gráfico 2-4:	Componente grupal de calidad proyectos periodo 2014 y 2015.....	52
Gráfico 3-4:	Índices del componente grupal de calidad de los proyectos	53
Gráfico 4-4:	Porcentaje del cumplimiento de requerimientos por día	54
Gráfico 5-4:	Componente individual de calidad de los proyectos periodo	55
Gráfico 6-4:	Índices del componente individual de calidad de proyectos.....	56
Gráfico 7-4:	Nivel de actividad Equipo 1	57
Gráfico 8-4:	Nivel de actividad Equipo 2.....	57
Gráfico 9-4:	Nivel de actividad Equipo 3.....	58
Gráfico 10-4:	Nivel de actividad Equipo 4.....	59
Gráfico 11-4:	Nivel de actividad Equipo 5.....	59
Gráfico 12-4:	Nivel de actividad Equipo 6.....	60
Gráfico13-4:	Nivel de actividad Equipo 7.....	61
Gráfico 14-4:	Nivel de actividad Equipo 8.....	61
Gráfico 15-4:	Nivel de actividad Equipo 9.....	62

Gráfico 16-4: Gráfico de caja y vigote para los dos grupos	68
Gráfico 17-4: Representación gráfica de valores calculados para el estadígrafo de contraste	70

DEDICATORIA

A Dios, que me devolvió la vida y la salud cuando sentía que las perdía. Sólo un poder tan grande y una bondad divina pudieron hacer ese milagro.

Para mi esposa Naty, fuente inagotable de amor y bondad, su soporte incondicional es mi aliento, su simple presencia mi inspiración.

Para mis papis Juanito y Anita y mis hermanos Sandrita y Marce, que son mi familia, mi refugio y mi ejemplo de amor y dedicación.

Javier.

AGRADECIMIENTO

Agradezco a mi esposa, docente de los cursos en los que se desarrolló el estudio. Su ayuda fue indispensable para completar cada una de las tareas previstas en esta investigación.

A la Ing. Ms. Blanquita Hidalgo, directora de tesis. Sus revisiones detalladas, su guía y recomendaciones dieron la forma y el matiz necesario para este proyecto.

A Dra. Narcisa Salazar, por su soporte especializado en el desarrollo estadístico del estudio y al Ing. Byron Vaca Ph.D. por su asesoría para la estrategia de análisis y presentación de resultados.

RESUMEN

Se propuso el método de seguimiento y evaluación de proyectos de fin de curso de materias de Lenguajes de Programación MESEPP, para carreras de Ingeniería Electrónica. El método propuesto presenta un procedimiento basado en el uso de GitHub, una herramienta para el control de versiones de archivos. El estudio sugirió que la aplicación de este método mejoraría la calidad de los proyectos de fin de curso desarrollados y presentados por los estudiantes. Los recursos técnicos utilizados fueron Git, GitHub, Software R, Visio, laboratorios de la Facultad de Informática y Electrónica (FIE) de la Escuela Superior Politécnica de Chimborazo (ESPOCH), institución donde se llevó a cabo el estudio. MESEPP fue aplicado en uno de dos grupos de estudiantes de las Escuelas de Ingeniería Electrónica de la FIE para realizar un contraste de diferencia de sus medias. Para ello se utilizó un estadígrafo de contraste unilateral basado en la distribución normal, con nivel de significación de 5% y valor crítico de 1.65. El valor del estadígrafo fue de 1.33, resultando menor al valor crítico y llevando a concluir en el estudio que no se tiene evidencia suficiente para afirmar que la aplicación del método propuesto mejora la calidad de los proyectos de fin de curso. Se recomienda realizar nuevas investigaciones para refinar el método propuesto y obtener los efectos esperados en esta investigación.

Palabras clave: <PROPUESTA METODOLÓGICA>, <EVALUACIÓN EDUCATIVA>, < PROYECTO DE FIN DE CURSO>, <MÉTODO [MESEPP]>, <SISTEMA DE VERSIONAMIENTO [GITHUB]>, <LENGUAJES DE PROGRAMACIÓN>

ABSTRACT

The method of monitoring and evaluation of projects to course materials MESEPP programming was proposed, for electronic engineering careers. The proposed method presents a method based on the use of GitHub, a tool for version control file. The study suggested that the application of this method would improve the quality of end-of-course projects developed and presented by the students. The technical resources used were Git, GitHub, Software R, Visio, laboratories of the faculty of computer and electronics (FIE) Higher Polytechnic School of Chimborazo (ESPOCH), Institution which was held on study. MESEPP was applied in one of two groups of students from the Schools of Electronic engineering of the FIE, for a contrast difference of her stockings. For this, a unilateral Statistician contrast based on the normal distribution was used, with significance level of 5% critical value of 1.65. Statistician value was 1.33, lower than the critical value results and led to the conclusion in the study that there is no evidence of ongoing projects for the end of course. We recommend further research to refine the proposed project and achieve the expected effects in this research method.

Key words: <METHODOLOGICAL PROPOSAL>, <EDUCATIONAL ASSESSMENT>, <END OF COURSE PROJECT>, <METHOD MESEPP>, <GITHUB VERSIONING SYSTEM>, <PROGRAMMING LANGUAGES>

CAPÍTULO I

1. INTRODUCCIÓN

En este trabajo de investigación se propone un método para el seguimiento y evaluación de los proyectos de fin de curso de la materia de Lenguajes de Programación en Carreras de Ingeniería Electrónica.

El objetivo de la aplicación del método es aumentar la calidad de los proyectos de fin de curso desarrollados por los estudiantes de la materia. El estudio se realiza en las Escuelas de Ingeniería Electrónica de la Facultad de Informática y Electrónica (FIE) de la Escuela Superior Politécnica de Chimborazo (ESPOCH).

En el Capítulo I se describe el problema existente en el proceso de ejecución y evaluación de los proyectos de fin de curso en materias de lenguajes de programación, se presenta la necesidad de incorporar un método que guíe este proceso y se definen los objetivos que se pretenden alcanzar como parte del estudio.

En el Capítulo II se realiza la definición de método, se desarrolla una revisión teórica acerca de los diferentes tipos de método, se describen las funcionalidades y características generales de las herramientas Git y GitHub y se realiza un estudio de su uso y aplicación en el ámbito de la educación.

En el Capítulo III se detalla el procedimiento utilizado para la investigación, se define el tipo de investigación y se determina la muestra. Se presentan adicionalmente los métodos y técnicas utilizadas.

En el Capítulo IV se presentan los resultados obtenidos en la investigación y se utiliza la estadística inferencial para desarrollar el procedimiento para la demostración de la hipótesis.

En el Capítulo V se realiza la propuesta del método de seguimiento y evaluación de proyectos de programación (MESEPP) soportado por la plataforma GitHub.

1.1. Formulación del problema

En la Escuela Superior Politécnica de Chimborazo (ESPOCH) al igual que en otras instituciones de educación superior, se utilizan diferentes métodos de trabajo por parte de los docentes para la valoración y evaluación de los estudiantes.

El proyecto de fin de curso es uno de los instrumentos de enseñanza/aprendizaje ampliamente utilizado por los docentes en las carreras de ingeniería para fortalecer el afianzamiento de conocimientos y potenciar cualidades como la creatividad y el ingenio.

El proyecto no sólo fortalece cualidades propias de una ingeniería, también potencia en los estudiantes habilidades y destrezas como:

- Capacidad en la resolución de problemas.
- Abstracción teórica y aplicación a la realidad.
- Organización personal y grupal.
- Interacción social.
- Liderazgo.
- Participación y comunicación.
- Toma de decisiones y otros.

El uso del proyecto de fin de curso en materias de carreras de ingeniería es pertinente, no obstante su aplicación trae consigo algunos problemas.

El proyecto es un trabajo en equipo, sin embargo, algunos estudiantes no llegan a beneficiarse de él. Su esfuerzo sigue siendo individual. Ciertos equipos se vuelven dispares debido a que algunos miembros resuelven un alto porcentaje del proyecto mientras que otros se convierten en observadores pasivos.

Algunos equipos concentran su atención en la calificación, entonces utilizan proyectos existentes o heredados para lograr sus objetivos. Algunos estudiantes se benefician de

manera individual de la ciencia que aporta el proyecto, pero descuidan la colaboración con su equipo.

El proyecto es un proceso, sin embargo, no es tratado como tal. Los proyectos se visibilizan a los evaluadores sólo días previos a la finalización del semestre.

El docente o el comité de docentes dispuesto para la evaluación, valora en unos cuantos minutos el esfuerzo individual y grupal que los estudiantes han realizado sobre el proyecto y definen una calificación para cada miembro del equipo. Algunos docentes con astucia identifican a los miembros pasivos quienes reciben una calificación que los penaliza.

El docente y el comité no interactúan periódica y progresivamente con los equipos. El día de la presentación podría ser la primera y la última vez que los docentes conozcan del proyecto.

En las carreras de Ingeniería Electrónica de la Facultad de Informática y Electrónica de la ESPOCH existen materias en las cuales los docentes utilizan el proyecto de fin de curso como un instrumento de aprendizaje. Su uso pretende recoger sus beneficios, sin embargo los lleva a involucrarse también con sus problemas.

De manera específica en la materia de Lenguajes de Programación I se define un proyecto de programación, con equipos de tres (3) personas que implementan una aplicación que intente resolver de manera efectiva un problema planteado por el docente.

Entendiendo las especificidades que mantiene un proyecto de lenguajes de programación, en el que un programa de computadora con líneas de código aportadas por cada uno de los miembros del equipo es el producto resultante, el presente trabajo de investigación pretende definir, proponer y validar un método para el seguimiento del proyecto de fin de curso, con el fin de mitigar los problemas inherentes a su ejecución. Como un apoyo tecnológico y a la vez como una parte integrante del método, se utilizará “GitHub”, la plataforma para versionamiento de archivos más popular del mercado de software y de uso gratuito para proyectos de código abierto.

Con el desarrollo del trabajo se pretende dar respuesta a la siguiente pregunta de investigación:

¿Permitirá un método de seguimiento de proyectos de fin de curso aplicado en la materia de Lenguajes de Programación y soportado por GitHub, mejorar la calidad de los proyectos en carreras de Ingeniería Electrónica?

A pesar de que la estimación de la calidad es subjetiva para diferentes escenarios, el presente trabajo sugerirá la estrategia de ponderación de la calidad de un proyecto de fin de curso en las materias de Lenguajes de Programación.

Así mismo las respuestas a las siguientes preguntas pretenden complementar el contexto de investigación:

¿Existen otras experiencias con la utilización de GitHub en carreras de ingeniería, que soporten su validez y su uso para esta investigación?

¿Cuáles son las funcionalidades de GitHub que ayudan en el seguimiento de proyectos de fin de curso en materias de lenguajes de programación?

¿Cuáles son los resultados obtenidos en la ejecución de proyectos de fin de curso cuando se utiliza un método de seguimiento basado en el uso de GitHub?

¿Cuáles deberían ser los componentes de un método para el seguimiento de proyectos de fin de curso que permita mitigar sus problemas?

El estudio será realizado sobre dos grupos de estudiantes de la materia de Lenguajes de Programación I de las carreras de Ingeniería Electrónica de la ESPOCH. Mientras el primer grupo ha utilizado un esquema tradicional para el seguimiento del proyecto de fin de curso, el segundo grupo utilizará un método definido y propuesto como parte de este trabajo de investigación.

1.2. Justificación

Siendo los proyectos de fin de curso altamente recomendables para el aprendizaje y afianzamiento de conocimientos de la materia de Lenguajes de Programación, es

necesario adoptar algún procedimiento que permita mitigar los problemas inherentes y que permita potenciar las bondades intrínsecas a su ejecución.

“A priori” un método para la ejecución y seguimiento del proyecto, que guíe al docente y a los estudiantes, que incluya un componente de seguimiento y retroalimentación diario, y que utilice a GitHub como una herramienta de soporte tecnológico, parece una propuesta conveniente para la mitigación de los problemas.

Con el seguimiento diario se podría tener una valoración más acertada del verdadero trabajo individual y grupal de los estudiantes, se podría encontrar errores o deficiencias de los estudiantes de manera temprana, se podría ayudar a los estudiantes en las especificidades de su proyecto, se podría dar una retroalimentación personalizada y a tiempo para cada equipo de trabajo.

El seguimiento diario podría fomentar la motivación del estudiante, el grado de compromiso con su proyecto y equipo, la ejecución a tiempo del proyecto, el avance progresivo y uniforme de sus actividades.

Encontrar un método de seguimiento diario, simple pero efectivo para la ejecución y seguimiento de los proyectos de fin de curso en la materia de Lenguajes de Programación de las carreras de Ingeniería Electrónica de la FIE motiva y justifica el presente trabajo de investigación.

A nivel general la justificación del proyecto se enmarca dentro de la línea de investigación “Herramientas computacionales para la enseñanza” de entre las delineadas por la Maestría en Informática Educativa grado al cual se pertenece el presente proyecto de investigación, y se alinea con el “Programa para el desarrollo de aplicaciones de software para procesos de gestión y administración pública y privada. Educación.” sugerido por la línea de investigación “Tecnologías de la información, comunicación y procesos industriales” de la ESPOCH.

Como un complemento la investigación pretende aportar dentro del sub área de “Tecnologías de la información y comunicación” perteneciente al área de investigación de “Ciencias de la producción e innovación” de la SENESCYT, alcanzando así el

Objetivo 2 del Plan Nacional del Buen Vivir (Secretaría Nacional de Planificación y Desarrollo - Ecuador, 2009), el cual dicta:

OBJETIVO 2: Mejorar las capacidades y potencialidades de la ciudadanía.

Política 2.2. Mejorar progresivamente la calidad de la educación, con un enfoque de derechos, de género, intercultural e inclusiva, para fortalecer la unidad en la diversidad e impulsar la permanencia en el sistema educativo y la culminación de los estudios.

Fomentar la evaluación contextualizada de los logros educativos de los estudiantes.

1.3. Objetivos

1.3.1. General

Proponer un método para el seguimiento de proyectos de fin curso de materias de Lenguajes de Programación en carreras de Ingeniería Electrónica.

1.3.2. Específicos

- Estudiar otras experiencias con la utilización de GitHub en carreras de ingeniería, que soporten su validez y su uso para esta investigación.
- Estudiar la plataforma GitHub para determinar las funcionalidades que ayuden en el seguimiento de proyectos de fin de curso en materias de lenguajes de programación.
- Aplicar el método propuesto en el seguimiento de los proyectos de fin de curso de la materia de Lenguajes de Programación I en las carreras de Ingeniería Electrónica de la ESPOCH y analizar sus resultados.
- Proponer un método para el seguimiento de proyectos de fin de curso de materias de Lenguajes de Programación en carreras de Ingeniería Electrónica, que incluya a GitHub como la herramienta de soporte tecnológico.

1.4. Hipótesis

Hipótesis de investigación

La utilización de un método que incorpore a GitHub como herramienta de soporte tecnológico en el seguimiento de proyectos de fin de curso de materias de Lenguajes de Programación en carreras de Ingeniería Electrónica mejora la calidad de los proyectos.

CAPÍTULO II

2. MARCO DE REFERENCIA

2.1. Definición de Método

La Real Academia de la Lengua define a la palabra método como el “modo de decir o hacer con orden, modo de obrar o proceder, hábito o costumbre que cada uno tiene y observa” (Real Academia Española, 2001). Un método refiere entonces la forma en la que se puede realizar una actividad, tarea o trabajo para alcanzar un objetivo particular.

En general el método puede ser definido como el camino o procedimiento viable para conseguir un fin propuesto (Gutiérrez, 2006; citado en Coria, et al. 2013), esto es, el método constituye una serie de pasos sistematizados que contribuyen al logro de ciertos objetivos.

Es importante notar que las definiciones de método no asumen su éxito, es decir, un método no es método por ser eficaz o eficiente, lo es tan sólo por el hecho de definir el camino para alcanzar su objetivo.

El método, sin embargo, difiere del concepto de metodología. “La metodología proviene del griego *methodos* (camino a seguir) y *logos* (estudio o tratado de)” (Coria, et al., 2013, p. 7). La metodología es entonces “el estudio de los métodos” (Bunge, 2005) y por tanto se encuentra en una escala de abstracción mayor. Si el método presenta y describe un conjunto de pasos, tareas o actividades a ejecutarse para completar el objetivo, la metodología lo estudia, analiza la coherencia de sus actividades, el logro de los objetivos, su ámbito de aplicación.

El método está intrínseco en un proceso de investigación científica, el cual se constituye por tres condiciones i) un producto, como objeto de estudio basado en teorías y hechos; ii) un método, como un conjunto de acciones destinadas al descubrimiento y dirigidas a la validación del conocimiento, y iii) condiciones de realización, como los elementos interpuestos entre el investigador y su objeto (Samaja, 2004).

Se puede afirmar entonces que la investigación utiliza métodos y no metodologías para su desarrollo (Bunge, 2005).

Sin embargo, “la secuencia de pasos, el seguimiento de lineamientos heurísticos y la contundencia de las reglas de métodos se aplican con diferente rigor dependiendo del tipo de investigación” (Coria, et al., 2013). Algunos tipos de investigación junto con el método utilizado y su producto final se presentan en la Tabla 1-2:

Tabla 1-2: Los tipos de investigación y el método.

Tipo de investigación	Método	Producto final
Investigación básica (o científica) e investigación Aplicada (o tecnológica)	Método científico Otros métodos y técnicas de apoyo.	Conocimiento científico Desarrollo tecnológico
Tipo de Aplicada (o Técnica)	Distintos métodos	Solución de problemas prácticos (controlar ciertos sectores escogidos de la realidad, mediante el diseño de dispositivos técnicos y la elaboración de planes o cursos de acción que tengan algún valor práctico para algún grupo social)

Fuente: (Coria, et al., 2013)

La ciencia intenta descubrir leyes o realizar desarrollos tecnológicos para entender la realidad, mientras que la técnica pretende controlar ciertos sectores de la realidad con ayuda de conocimientos de todo tipo, principalmente los científicos. Ambas parten de problemas, pero si los problemas científicos se entienden como puramente cognoscitivos los técnicos se entienden como prácticos. (Torres & Navarro 2007; citados en Coria et al., 2013)

2.2. Tipos de método

Los métodos de investigación pueden ser de dos tipos.

Primero están aquellos que constituyen actividades humanas de carácter interno, llamados métodos de pensamiento (Sierra, 2001), que como lo describen algunos autores (Del Cid, et al., 2007), son procesos lógicos que utiliza la razón para relacionar datos, como los métodos analítico, el deductivo, el inductivo, entre otros.

Por otro lado están aquellos que constituyen actividades humanas de carácter externo, llamados métodos de acción, que a su vez pueden ser aquellos que buscan transformar la realidad y aquellos que buscan conocerla (Sierra, 2001).

En el área de la investigación se pueden encontrar distintos métodos, en dependencia del contexto, el fin y los objetivos de la investigación. Coria (et al., 2013) lo refleja en el siguiente listado:

- *De tipo general*
 - *Método experimental.*
 - *Método de muestreo al azar.*
 - *Método estadístico en general (estadística descriptiva, correlacional, inferencial, probabilidad)*
 - *Método de aproximaciones sucesivas.*
 - *Otros.*
- *De pensamiento*
 - *Abstracción*
 - *Deducción*
 - *Inducción*
 - *Análisis*
 - *Síntesis.*
 - *Definición.*
 - *Clasificación.*
 - *Comparación.*
 - *Otros.*
- *De acción*
 - *Del conocimiento del mundo o la realidad.*
 - *Realidad presente – Científicos.*
 - *Realidad histórica – No científicos.*
 - *Métodos didácticos.*
 - *Métodos de trabajo y producción (transformación de la realidad):*
 - *Toma de decisiones bajo condiciones de certidumbre y riesgo:*
 - *Programación lineal.*

- *Método gráfico para dos variables.*
- *Método simplex.*
- *Método de transporte.*
- *Toma de decisiones bajo condiciones de incertidumbre y riesgo:*
- *Decisiones con base en probabilidades.*
- *Decisiones con base en valores monetarios: matriz de utilidad.*
- *Árboles de decisión*
- *Pronósticos*
- *Técnicas cualitativas (escenarios, técnicas Delphi)*
- *Técnicas cuantitativas (series de tiempo, regresión)*
- *Programación de proyectos*
- *Gráfica de Gantt*
- *Program Evaluation and Review Technique (PERT)*
- *Critical Path Method (CPM)*
- *Administración estratégica*
- *Matriz de fortalezas-debilidades, amenazas-oportunidades (FODA)*
- *Matriz de posición estratégica y evaluación de acción (PE y EA)*
- *Matriz del Boston Consulting Group (BCG)*
- *Matriz de Planeación Estratégica Cuantitativa (MPEC)*
- *Administración de la calidad total*
- *Benchmarking*
- *Justo a tiempo*
- *Total Quality Management (TQM)*
- *Mejora continua*
- *Producción*
- *Planeación de requerimientos (MRPI, MRPII, ERP)*
- *Distribución física para operaciones (Lay out)*
- *Métodos de cola y líneas de espera*
- *Métodos de inventarios (ABC, EOQ)*
- *Simulación Monte Carlo*
- *Ingeniería de control moderno:*
- *Transformada de Laplace*
- *Modelado matemático de sistemas dinámicos*
- *Análisis de la respuesta transitoria y estacionaria*
- *Análisis del lugar de las raíces*
- *Análisis y diseño de sistemas de control en el espacio de estado*
- *Otros*

2.3. Sistemas de control de versiones

“Los sistemas de control de versiones son sistemas que registran los cambios sucedidos en el tiempo, de uno o varios archivos, con el fin de recuperar después versiones específicas de ellos” (Chacon S, 2014, p. 27).

A pesar de que los sistemas de control de versiones son utilizados en mayor porcentaje para el control de archivos de código fuente de aplicaciones informáticas, en realidad pueden ser utilizados para cualquier tipo de archivo.

2.3.1. Sistemas de control de versiones local

Son sistemas que mantienen una base de datos local en el computador del usuario, en ella mantienen el registro de los cambios sobre los archivos.

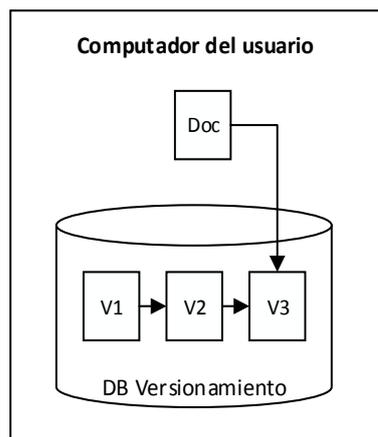


Gráfico 1-2: Sistema de control de versiones local

Fuente: (Chacon S, 2014, p. 28)

El Gráfico 1-2 representa el computador del usuario y su base de datos local que registra las diferentes versiones de un archivo. Mientras la base de datos local mantiene el historial completo de los cambios de versiones, el archivo real “Doc” tiene el contenido de la última versión “V3”. El usuario cada vez que lo requiera puede tener acceso a versiones anteriores del archivo, pues toda esta información la mantiene en su mismo computador.

2.3.2. Sistemas de control de versiones centralizado

En este tipo de sistemas un servidor es quien mantiene el registro de todos los cambios sobre los archivos. En las computadoras de los usuarios se mantiene una única versión de archivos.

Para acceder a otras versiones de archivos o su historial se depende de la disponibilidad del servidor central. Si el servidor central se daña o la base de datos que mantiene el registro de las versiones de archivos se corrompe toda la información se pierde.

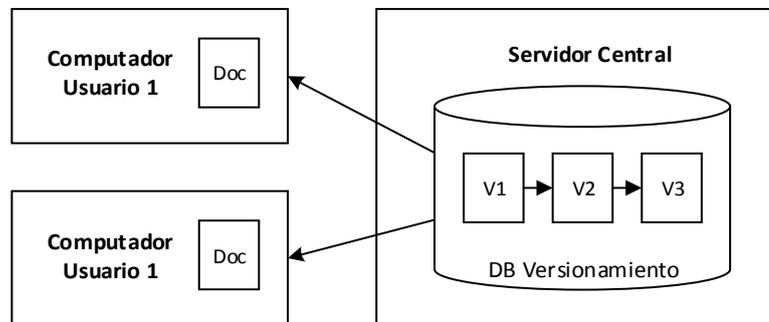


Gráfico 2-1: Sistema de control de versiones centralizado

Fuente: (Chacon S, 2014, p. 29)

2.3.3. Sistemas de control de versiones distribuido

En este tipo de sistemas la información de versiones se mantiene en un servidor central, a la vez, la misma y completa información se mantiene en las computadoras de los usuarios. Si la información del servidor central se corrompe, el copiar la información de cualquiera de los computadores de los usuarios de vuelta al servidor central permite recuperar toda la información.

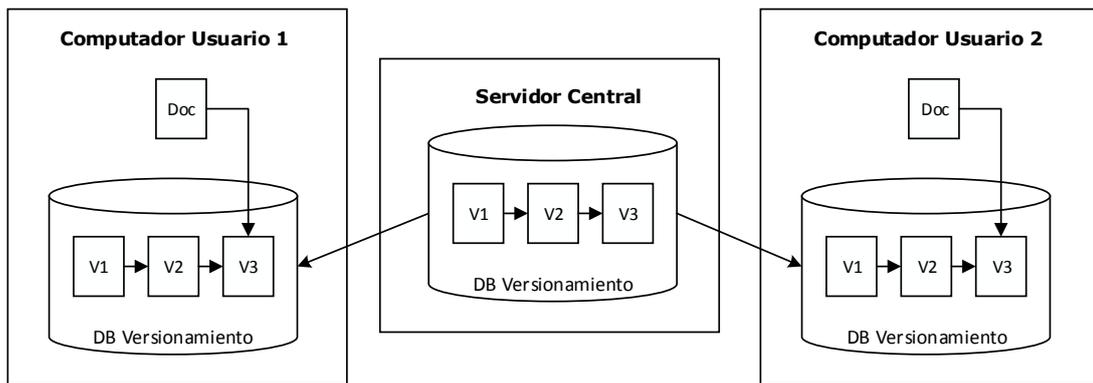


Gráfico 3-2: Sistema de control de versiones distribuido

Fuente: (Chacon S, 2014, p. 30)

2.4. Git

Git es una herramienta gratuita y de código abierto que se encuentra en la categoría de sistemas de control de versiones distribuido (Git, 2015).

Herramientas como Subversion, Perforce, CVS, SourceSafe, Team Foundation son otras del mercado que tienen capacidades de controlar versiones de archivos.

Git tiene su nacimiento en el año 2005, como una iniciativa de la comunidad Linux, y en especial de su creador Linus Torvalds para crear una herramienta que reemplace a la propietaria que hasta ese momento era utilizada en el sistema Linux denominada como BitKeeper (Chacon S, 2014).

Git puede instalarse en cualquier sistema operativo y mantiene la historia de cambios de manera descentralizada, esto es, que la historia se encuentra registrada y almacenada en cada una de las computadoras de sus autores (SitePoint Pty. Ltda., 2014).

2.4.1. Funcionamiento

Algunos sistemas de control de versiones como Bazaar, CVS, Perforce, Subversión, tienen un esquema de versiones basado en cambios. Estos sistemas almacenan la información como una lista de archivos y los cambios realizados sobre estos archivos a lo largo del tiempo (Chacon S, 2014).

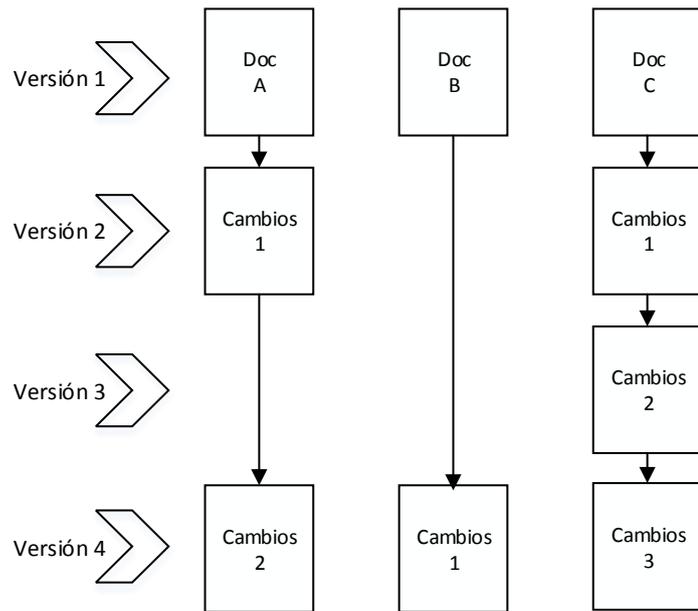


Gráfico 4-2: Versionamiento basado en cambios

Fuente: (Chacon S, 2014, p. 32)

Por otro lado Git utiliza un esquema basado en archivos. Git almacena la información como una foto copia del árbol de archivos que observa en un momento dado.

Cada vez que el usuario guarda un estado de su proyecto, Git saca una foto copia de cómo se encuentran en ese momento exacto los archivos y crear una referencia a esa foto copia.

Como una forma de mejorar su eficiencia, si un archivo no ha cambiado Git no guarda el archivo nuevamente, únicamente realiza un enlace al archivo previo que ya se encuentre almacenado (Chacon S, 2014).

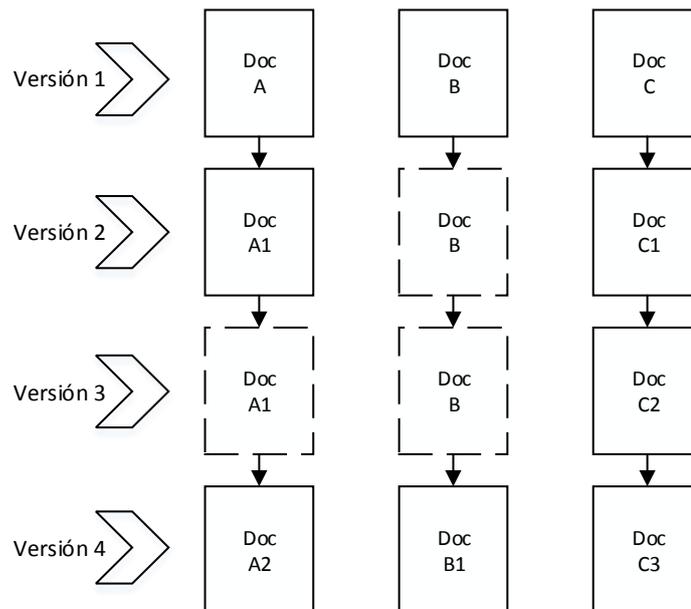


Gráfico 5-2: Versionamiento basado en archivos

Fuente: (Chacon S, 2014, p. 32)

En el Gráfico 5-2, los documentos con línea entrecortada, representan simples enlaces al documento original que en la nueva versión no ha variado. Esta estrategia con la que Git maneja las versiones lo distingue de otros sistemas de versiones y le permiten ofrecer funcionalidades que con otros esquemas sería imposible.

Debido a que Git almacena en el computador del usuario el historial completo de cambios, muchas de sus operaciones se las puede realizar sin tener una conexión con un servidor central. Tareas como revisar el historial, comparar la versión actual de un archivo con una versión de hace una semana y regresar a esa versión, son tareas que pueden ser ejecutadas de manera local y por lo tanto de forma veloz.

Git mantiene una integridad de los archivos. Los archivos siempre son observados para capturar todos sus cambios y registrarlos en su historial. Ningún archivo puede tener un contenido diferente al que ha sido almacenado por Git. Para identificar la versión de un archivo se utiliza un checksum.

El mecanismo que GitHub utiliza para realizar el checksum es denominado como SHA-1 hash. Esto es un cadena de 40 caracteres compuesta por caracteres hexadecimales (0-9

y a-f) y calculada basada en los contenidos de un archivo o la estructura del directorio (Chacon S, 2014, p. 33).

Git tiene tres estados principales en los que los archivos pueden estar en un momento dado: entregado (committed), modificado (modified) y ensayado (staged). El entregado significa que la información ha sido almacenada de manera segura en la base de datos local.

A la acción de almacenar un conjunto de cambios en la base de datos local se la denomina commit. Modificado significa que se han realizado cambios sobre los archivos pero que aún no han sido almacenados en la base de datos de Git. Ensayado significa que un archivo modificado has sido marcado en la versión actual para ir a la base de datos de Git en el próximo commit.

Esto nos lleva a definir las tres secciones principales de un proyecto Git: El directorio Git, el directorio de trabajo, y el área de ensayo (Chacon S, 2014, p. 34).

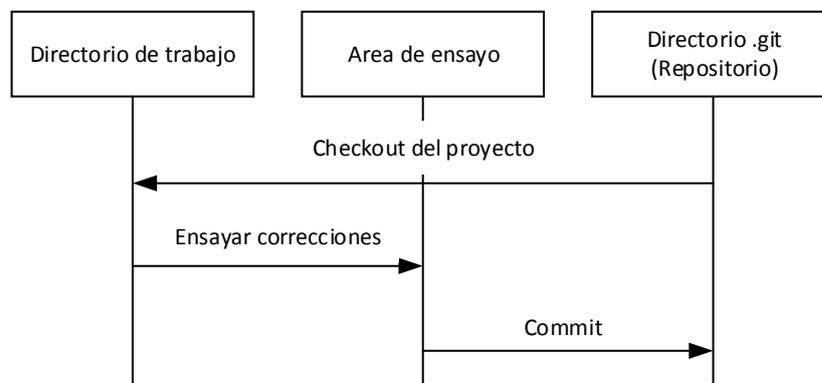


Gráfico 6-2: Principales secciones de un proyecto en Git

Fuente: (Chacon S, 2014, p. 35)

En Git el usuario puede configurar de manera explícita archivos para los cuales no se necesita que se realice un seguimiento.

Aquellos archivos a los cuales sí se realiza un seguimiento, pueden estar en su tiempo de vida en tres estados: No modificado, modificado, ensayado.

Cuando se clona un repositorio, todos los archivos estarán en el estado “no modificado”. Cuando un archivo es editado Git lo detecta automáticamente y los marca en un estado de “modificado”.

El usuario puede marcar a este archivo modificado como “ensayado” para posteriormente realizar un commit y con ello registrar los cambios en la base de datos local. Al realizar un commit, el archivo pasará nuevamente a un estado de “no modificado”. Cada vez y para cada archivo este ciclo se repite.

Los nuevos archivos creados por el usuario empiezan en un estado en el que no se los realiza seguimiento, hasta que el usuario los marca como “ensayados”. Una vez que se realice el commit, estos nuevos archivos son marcados como “no modificados” y ahora se dará seguimiento de ellos utilizando el mismo flujo anterior. El Gráfico 7-2 muestra los estados de los archivos en su tiempo de vida.

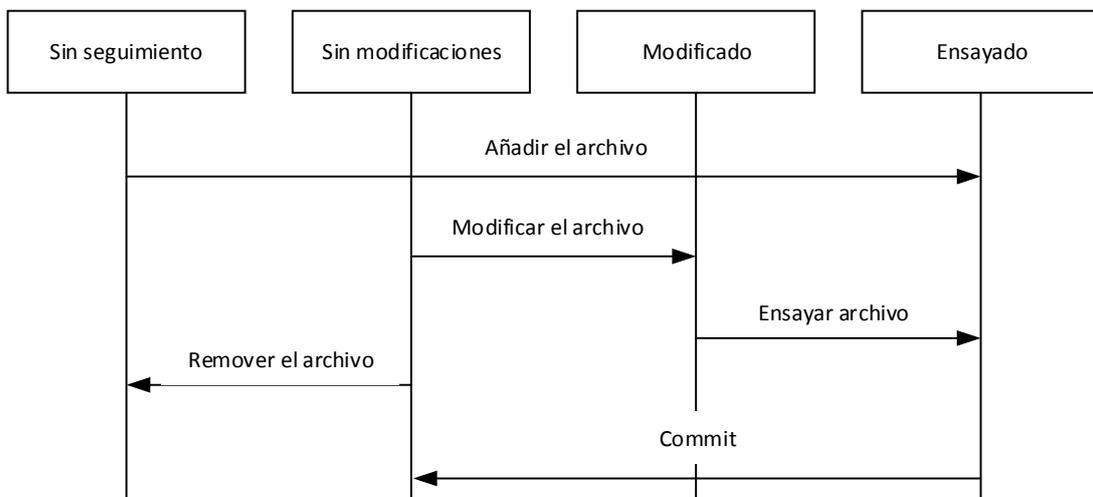


Gráfico 7-2: Estado de los archivos en su tiempo de vida en Git

Fuente: (Chacon S, 2014, p. 45)

2.4.2. Instalación

Git se puede encontrar como un paquete de instalación para cada una de las distintas distribuciones Linux y como un programa ejecutable de extensión .exe para las diferentes

versiones del sistema operativo Windows. Existen así mismo versiones disponibles para sistemas operativos MacOS y Solaris.

2.4.3. Conceptos en Git

Git al igual que otros sistemas de control de versiones utiliza conceptos que definen su funcionalidad. A continuación se presentan algunos de ellos:

Branch

Es un ramal, una bifurcación de una versión del proyecto. Puede ser utilizada para tomar una nueva dirección del proyecto, para realizar cambios que luego puedan ser incorporados de regreso al proyecto principal o para realizar ensayos de cambios.

Clonar

Es la acción de copiar un repositorio existente. Un usuario clona el branch principal o algún otro branch secundario, para tener una copia local sobre la cual puede empezar a trabajar y realizar todos los cambios que sean necesarios.

Repositorio

Un repositorio es un grupo nombrado de archivos que serán manejados como un solo proyecto. La jerarquía de archivos de este proyecto o repositorio se la denomina como “working tree”. Al primer nivel de la jerarquía de archivos existe una carpeta de nombre .git que contiene toda la información del repositorio y su historial, es decir la base de datos local de Git (Chacon S, 2014).

2.4.4. Comandos

La Tabla 2-2 presenta los comandos más utilizados en un ciclo de versionamiento en Git.

Tabla 2-2: Comandos relevantes en Git

Add	Añade los cambios de uno o varios archivos al área de ensayo. Esto es, los cambios observados en el momento que se ha ejecutado este comando serán registrados en la base de datos local en el próximo commit.
Commit	Es la acción de registrar los cambios de uno o varios archivos en la base de datos local de Git. Los archivos en este estado, están listos para ser enviados hacia un repositorio remoto.
Diff	Permite ver las diferencias entre dos árboles de archivos. Por ejemplo permite ver los cambios entre el directorio de trabajo y el área de ensayo o entre el área de ensayo y el último commit.
Pull	Cuando se trabaja con repositorios remotos permite traer la última versión remota, la más actual, hacia el repositorio local.
Push	Cuando se trabaja con repositorios remotos, envía la última versión que se hizo commit en el repositorio local, hacia el repositorio remoto.
Merge	Combina los cambios entre dos branches diferentes. Esta es una de las formas a través de las cuales se puede incorporar los cambios de un branch previo hacia el ramal original.
Rebase	En un procedimiento similar al merge, sin embargo con este se logra que la historia de cambios de dos branches, a pesar de que los cambios se ejecuten en paralelo, se registre y se vea como un conjunto de cambios realizados de manera secuencial.
Revert	Revierte los cambios que se hayan realizado a través del último commit.
Status	Detalla el estado de cada uno de los archivos en el directorio de trabajo. Este comando listará los archivos que hayan sido modificados y ensayados.

Realizado por: Javier Salazar, 2015

2.5. Github

GitHub es una plataforma web comercial lanzada en el año 2008 (Yu, et al., 2014), diseñada para facilitar el almacenamiento centralizado, la trazabilidad y colaboración sobre archivos de proyectos. Actualmente mantiene más de 28.1 millones de repositorios (GitHub, Inc., 2015).

Los usuarios pueden utilizar a GitHub para mantener repositorios remotos de sus repositorios locales, para ello Git es la piedra angular. Esta última es la herramienta que los colaboradores de un proyecto deberán instalar en sus computadores para manejar y sincronizar sus repositorios locales con los servicios especializados de GitHub.

GitHub es de uso gratuito para proyectos open source o de código abierto y es pago, bajo un esquema de licenciamiento, para proyectos de ámbito privado, cuyo código no deba ser abierto o no quiera ser público.

Muchos de los proyectos bandera del ámbito del open source se encuentran hospedados en GitHub.

2.5.1. La popularidad de GitHub

La plataforma ha tenido una curva creciente de popularidad en las comunidades, especialmente de desarrolladores. Individuos y corporaciones utilizan a GitHub como la herramienta primaria para el control y versionamiento de sus proyectos de software.

La popularidad de GitHub puede analizarse desde algunos puntos de vista. En un estudio realizado acerca del uso de GitHub sus autores, perciben la preferencia de GitHub basada en el efecto líder y el comportamiento rebaño (Yu, et al., 2014, p. 3):

Efecto de líder: Algunos desarrolladores disfrutan de trabajar con personas de alta reputación como Linus Torvalds que mantiene algunos de sus proyectos en GitHub.

Comportamiento Rebaño: Muchos desarrolladores utilizan GitHub solamente porque muchas de las personas a su alrededor utilizan y hablan de GitHub frecuentemente.

Así mismo su popularidad puede ser concebida por su capacidad para recrear la interacción social que los participantes de un equipo pueden tener de manera presencial.

El interés, la atracción, la motivación y el trabajo en equipo se ven potenciados cuando las personas van conociendo de manera transparente el comportamiento y trabajo de sus compañeros de equipo. GitHub logra ese tipo de transparencia y comunicación social a través de su plataforma.

En GitHub un colaborador puede realizar inferencias acerca de otro, en base a la información visibilizada y transparentada por la herramienta. Los desarrolladores “combinan estas inferencias en estrategias efectivas para coordinar proyectos, mejorar sus habilidades técnicas y gestionar su reputación” (Dabbish, et al., 2012, p. 1).

En un estudio realizado acerca de la transparencia y colaboración en repositorios de código abierto, se realizó entrevistas a varios desarrolladores usuarios de GitHub un resumen de sus resultados se presentan en la Tabla 3-2.

Tabla 3-2: Inferencias sociales basadas en la actividad en GitHub

Información visible	Inferencias sociales	Cita representativa
Actualidad (información recientemente ingresada) y volumen de actividad.	Interés y nivel de compromiso.	“Este pana de Mongoid es una máquina, sigue y sigue poniendo código.”
Secuencia de acciones en el tiempo.	Intención atrás de una acción	“Las aportaciones cuentan una historia. Indican la dirección hacia la cual estás tratando de ir con el código... revelando que quieres hacer.”
Atención a artefactos y personas.	Importancia a la comunidad	“Un proyecto con un número de personas viéndolo o interesados en él, obviamente es un mejor proyecto que otro que no tenga a nadie interesado en él.”
Información detallada acerca de una acción.	Impacto y relevancia personal	“Si existe algo en un aporte de un proyecto que me imposibilita un requerimiento o característica tengo la oportunidad de aportar en él.”

Fuente: (Dabbish, et al., 2012, p. 4)

Gracias a su gran popularidad GitHub, vive con una gran cantidad de proyectos de diferentes lenguajes de programación, diversos dominios de aplicación, una gran cantidad de líneas de código, un gran número de commits y muchos errores que se reportan y se corrigen en cada uno de los proyectos.

Para ejemplificar su popularidad, se presenta en la Tabla 4-2 los 3 proyectos más populares hospedados en GitHub para cada uno de los 19 lenguajes de programación más destacados existentes en esta plataforma:

Tabla 4-2: Top 3 de proyectos en GitHub por lenguaje de programación.

Lenguaje	Proyectos
C	linux, git, php-src
C++	node-webkit, phantomjs, mongo
C#	SignalR, SparkleShare, ServiceStack
Objective-C	AFNetworking, GPUImage, RestKit
Go	docker, lime, websocketd
Java	storm, elasticsearch, ActionBarSherlock
CoffeeScript	coffee-script, hubot, brunch
JavaScript	bootstrap, jquery, node
TypeScript	bitcoin, litecoin, qBittorrent

Fuente: (Ray, et al., 2014, p. 2)

Tabla 4-2: Top 3 de proyectos en GitHub por lenguaje de programación (Continuación)

Lenguaje	Proyectos
Ruby	rails, gitlabhq, homebrew
Php	laravel, CodeIgniter, symfony
Python	flask, django, reddit
Perl	gitolite, showdown, rails-dev-box
Clojure	LightTable, leiningen, clojurescript
Erlang	ChicagoBoss, cowboy, couchdb
Haskell	pandoc, yesod, git-annex
Scala	Play20, spark, scala

Fuente: (Ray, et al., 2014, p. 2)

2.5.2. Más allá del versionamiento de archivos

Como una línea de acción paralela GitHub se convierte también en una comunidad para visibilizar proyectos y personas. Mientras los proyectos pueden llamar la atención de inversionistas, sus autores pueden atraer la mirada que empresas que busquen contratarlos.

Muchos desarrolladores de software y proyectos multimedia han convertido a GitHub en una vitrina interesante de sus proyectos, ideas y trabajos, que a más de visibilizarlos al mundo, son un aporte fundamental para la creación de otros proyectos.

Pero no son sólo los autores originales quienes aportan a su proyecto, nuevos interesados pueden contribuir de manera directa o indirecta.

De manera directa si los propietarios del proyecto les entregan los permisos necesarios para que contribuyan.

De manera indirecta el interesado puede clonar el proyecto, crear un nuevo ramal y contribuir con sus cambios al proyecto original si así lo deseara. Se va formando entonces una red ilimitada de colaboradores que dan vida a ideas, que comparten talento y que colaboran con la comunidad.

2.5.3. Características

A continuación se presentan algunas de las funcionalidades y características que GitHub ofrece como parte de su servicio. Para la presentación de gráficos se ha tomado como referencia el proyecto bootstrap, un framework popular para la creación de aplicaciones multi dispositivo que es de código abierto y que se encuentra hospedado en la plataforma GitHub.

Búsquedas

Los navegantes de internet pueden buscar proyectos de su interés, en cualquier línea de negocio, como el desarrollo de sistemas, diseño, multimedia, música, etc.

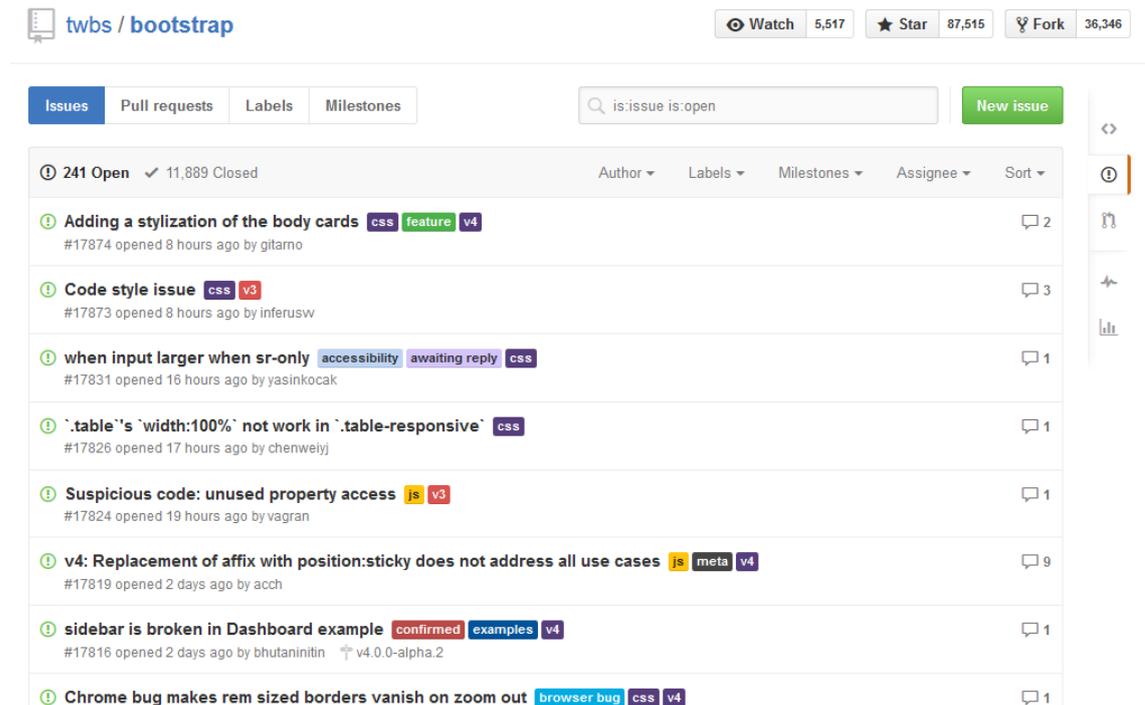
Pull Request

Es un aporte, cambio o mejora sugerido por un colaborador. Para ejemplificarlo, en el caso de un archivo de código para un proyecto de desarrollo de software, un “Pull Request” está formado por un conjunto de líneas de código que el colaborador aporta al proyecto original.

La persona o el equipo que soporta el proyecto revisa, analiza, tal vez prueba y valida que la contribución sea adecuada a las condiciones y necesidades del proyecto y que sea coherente con el cumplimiento de ciertos objetivos técnicos, funcionales y de negocio y si todo es correcto aceptará el “Pull Request”. Las nuevas líneas de código ahora son parte del proyecto.

Issues

Es una forma de mantener un seguimiento de tareas, mejoras y errores de los proyectos. Es una especie de email, excepto que los issues pueden ser compartidos y discutidos con el resto del equipo.



The screenshot shows the GitHub interface for the 'twbs / bootstrap' repository. At the top, there are buttons for 'Watch' (5,517), 'Star' (87,515), and 'Fork' (36,346). Below the repository name, there are tabs for 'Issues', 'Pull requests', 'Labels', and 'Milestones'. A search bar contains the text 'is:issue is:open' and a 'New issue' button. The main content area displays a list of issues with the following details:

Issue Title	Labels	Comments
Adding a stylization of the body cards	css, feature, v4	2
Code style issue	css, v3	3
when input larger when sr-only	accessibility, awaiting reply, css	1
.table's width:100% not work in .table-responsive	css	1
Suspicious code: unused property access	js, v3	1
v4: Replacement of affix with position:sticky does not address all use cases	js, meta, v4	9
sidebar is broken in Dashboard example	confirmed, examples, v4	1
Chrome bug makes rem sized borders vanish on zoom out	browser bug, css, v4	1

Figura 1-2: Lista de issues del proyecto bootstrap en GitHub.

Fuente: (GitHub, Inc., 2015)

Wikis

Son una forma de presentar información hacia los interesados del proyecto de una forma adecuada. Cada repositorio en GitHub tiene un wiki. En el wiki se pueden crear tantas páginas como sean necesarias. En cada una de las páginas se mantiene la pista de los cambios que se han realizado así como del colaborador que ha realizado el cambio.

Bootstrap



Bootstrap is a sleek, intuitive, and powerful front-end framework for faster and easier web development, created by [Mark Otto](#) and [Jacob Thornton](#), and maintained by the [core team](#) with the massive support and involvement of the community.

To get started, check out <http://getbootstrap.com/>

Table of contents

- [Quick start](#)
- [Bugs and feature requests](#)
- [Documentation](#)
- [Contributing](#)
- [Community](#)
- [Versioning](#)
- [Creators](#)
- [Copyright and license](#)

Figura 2-2: Wiki del proyecto bootstrap en GitHub.

Fuente: (GitHub, Inc., 2015)

Fork

Es una operación que permite realizar una copia de un proyecto. Este procedimiento permite utilizar un proyecto como un punto de partida de uno propio. El nuevo proyecto, puede servir como un puente con el original, al cual se puede aportar con mejoras y correcciones a través de Pull Requests.

Cuadro de control con gráficos de actividad

Pulso: Presenta de manera sintetizada los issues, pull request y commits que ha tenido un proyecto en un periodo determinado de tiempo.

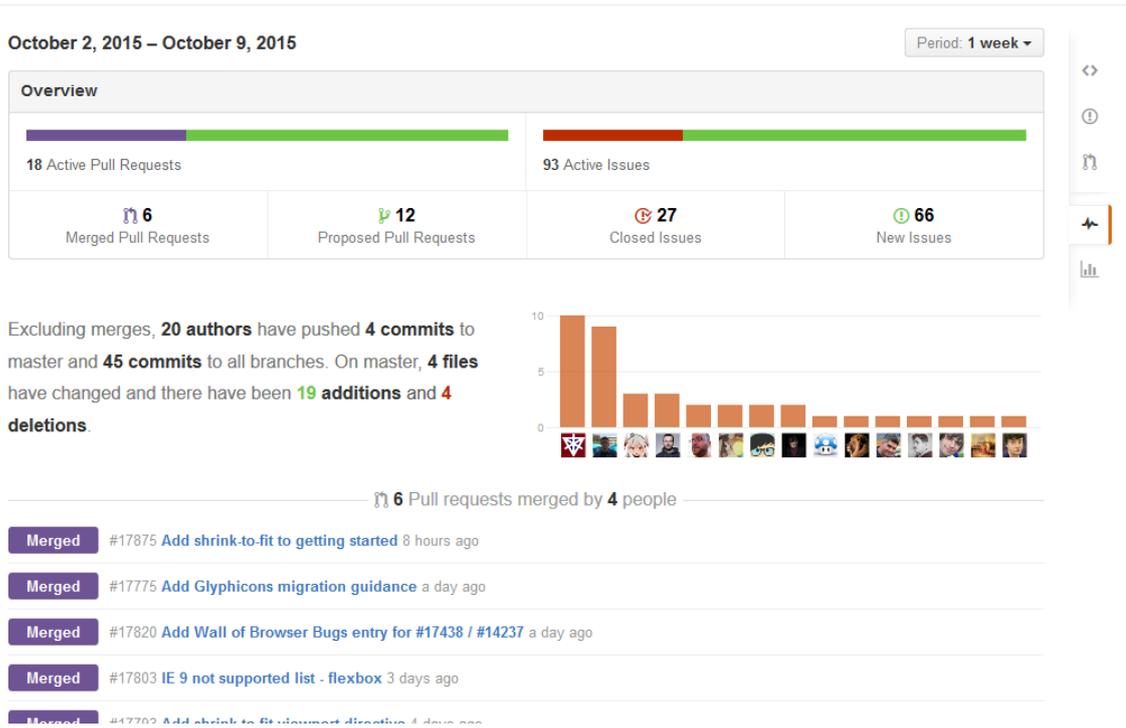


Figura 3-2: Vista del pulso de actividades del proyecto bootstrap en GitHub.

Fuente: (GitHub, Inc., 2015)

Contribuciones: Presenta las contribuciones que se han realizado sobre un proyecto en un periodo determinado de tiempo. Presenta las contribuciones totales así como los aportes individuales de cada uno de sus colaboradores.

Contributors

Commits

Code frequency

Punch card

Network

Members

Apr 24, 2011 – Oct 10, 2015

Contributions to master, excluding merge commits

Contributions: Commits

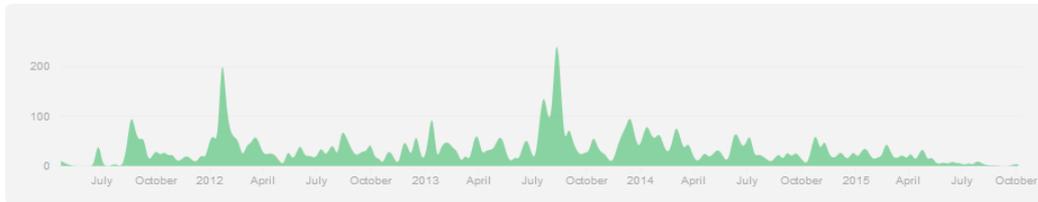


Figura 4-2: Vista de contribuciones del proyecto bootstrap en GitHub.

Fuente: (GitHub, Inc., 2015)

Commits: Presenta el número de commits que ha tenido un proyecto en un periodo determinado de tiempo. La información presenta la frecuencia de commits por fecha.

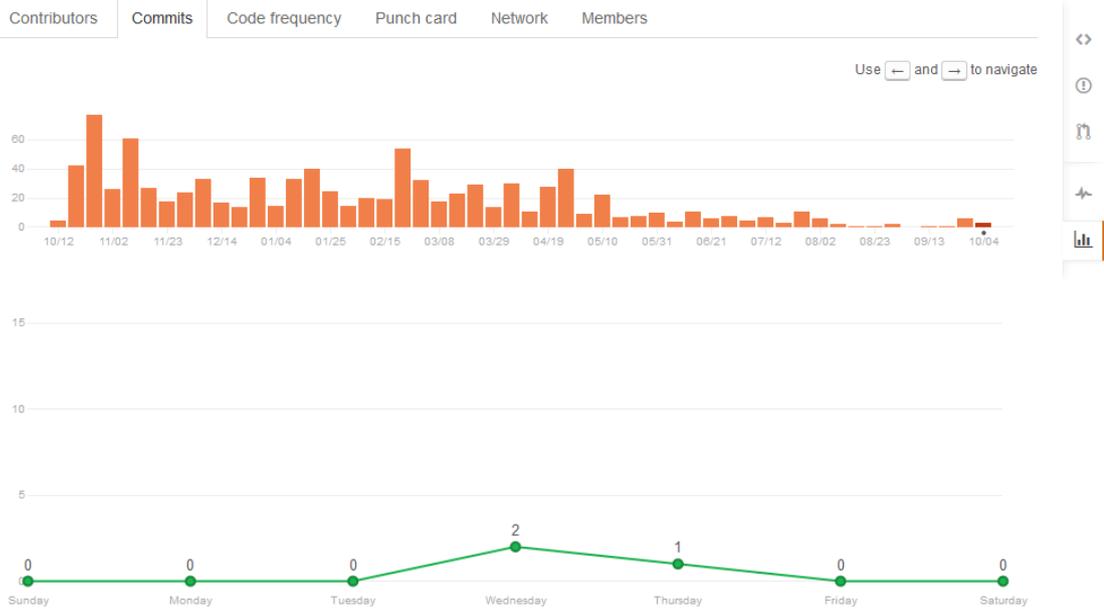


Figura 5-2: Vista de commits del proyecto bootstrap en GitHub.

Fuente: (GitHub, Inc., 2015)

Frecuencia de código: Presenta un resumen del número de líneas añadidas al proyecto y el número de líneas eliminadas del proyecto. Esta información es presentada por fechas.



Figura 6-2: Vista de frecuencia de código del proyecto bootstrap en GitHub.

Fuente: (GitHub, Inc., 2015)

Push Card: Es un informe que presenta la distribución de la actividad sobre el proyecto por días y por horas del día.

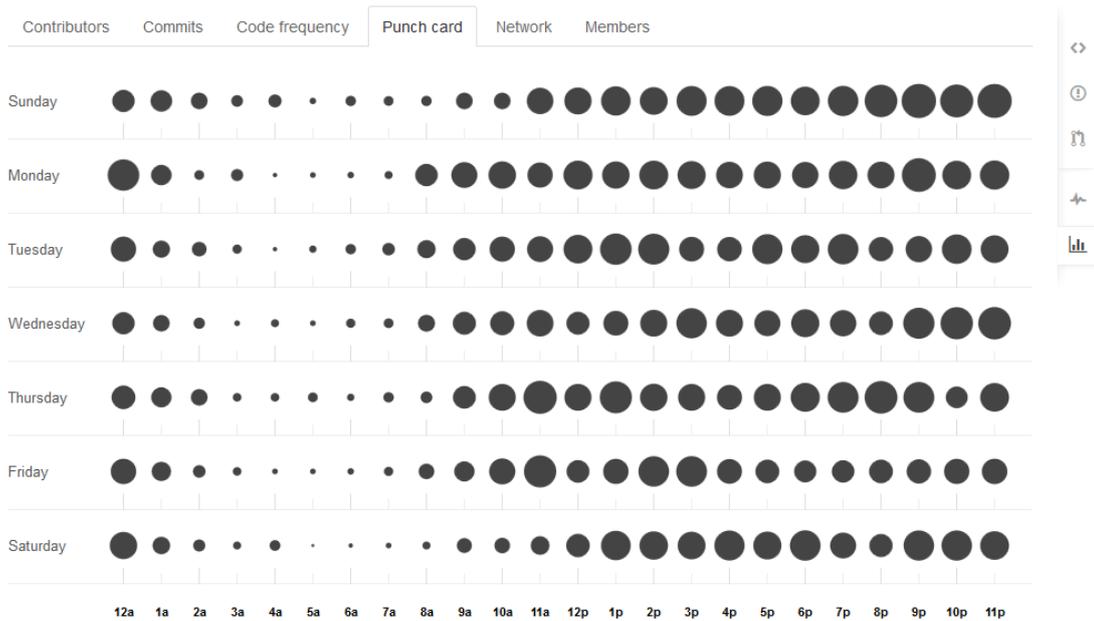


Figura 7-2: Vista Push Card del proyecto bootstrap en GitHub.

Fuente: (GitHub, Inc., 2015)

Milestones

Son una forma de clasificar los issues. Ejemplos de milestones utilizados en algunos proyectos son: “Lanzamiento en versión Beta”, “Segunda quincena de octubre”, “Rediseño del proyecto”, “v3.0”.

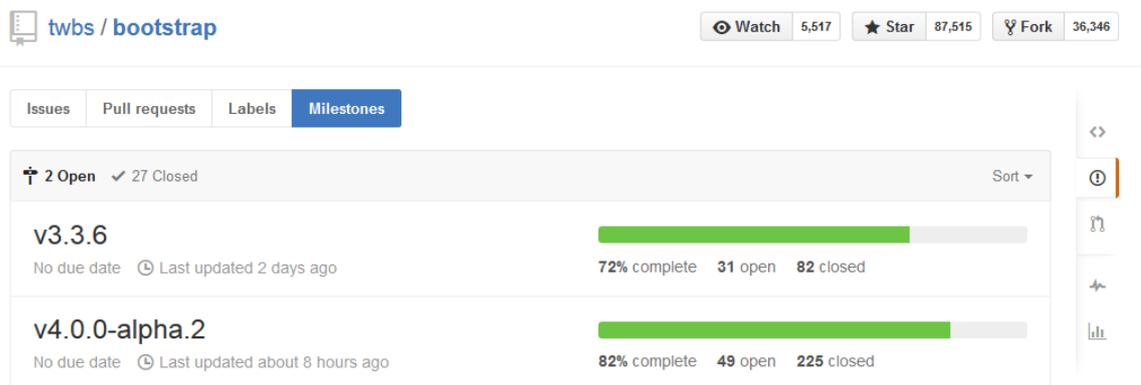
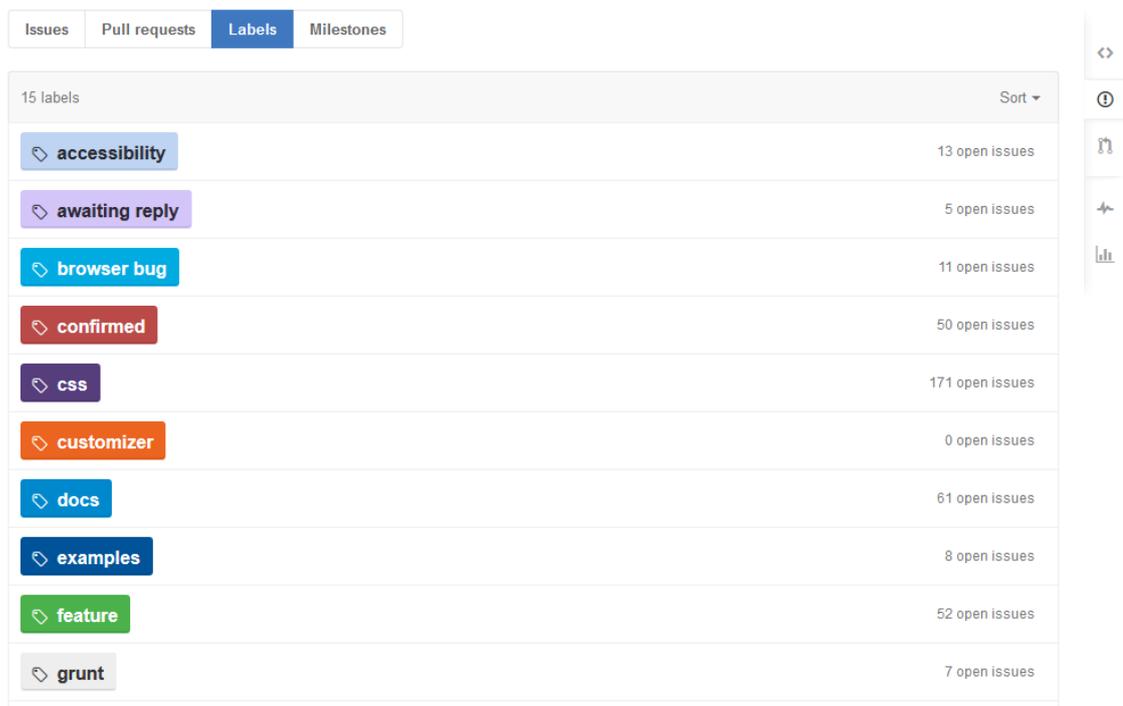


Figura 8-2: Vista de milestones del proyecto bootstrap en GitHub.

Fuente: (GitHub, Inc., 2015)

Labels

Son una forma de clasificar los issues. Generalmente son utilizados como “tipos” de issues. Algunos ejemplos son “Nueva característica”, “Error”, etc.



Label	Open Issues
accessibility	13
awaiting reply	5
browser bug	11
confirmed	50
css	171
customizer	0
docs	61
examples	8
feature	52
grunt	7

Figura 9-2: Vista de labels del proyecto bootstrap en GitHub.

Fuente: (GitHub, Inc., 2015)

Assignees

Son usuarios del equipo a quienes se asignan los issues.

Actividades Sociales

GitHub tiene funcionalidades que permiten realizar actividades como las que se pueden encontrar en redes sociales. Comportamientos como “seguir” a un usuario de GitHub u “observar” proyectos se encuentran disponibles. Con ellas se puede obtener notificaciones de las actividades que realiza un usuario o que se realiza sobre un proyecto respectivamente.

2.5.4. Permisos para usuarios

Cuando un usuario se registra en el la plataforma de GitHub obtiene una cuenta de usuario con la que posteriormente podrá ingresar al servicio.

Una cuenta de usuario sobre un repositorio, puede tener alguna de los siguientes niveles de permisos (GitHub, 2015):

Propietario de repositorio

Tiene control completo sobre el repositorio. Únicamente un usuario puede ser propietario de un repositorio. Adicionalmente de los permisos permitidos para un colaborador el propietario puede realizar las siguientes actividades:

- Añadir colaboradores.
- Cambiar la visibilidad del repositorio, de privado a público y viceversa.
- Eliminar el repositorio.

Colaborador

Un colaborador de un proyecto puede realizar las siguientes actividades:

- Push, para escribir.
- Pull, para leer.
- Fork, para copiar un repositorio.
- Aplicar etiquetas (labels) e hitos.
- Abrir, cerrar, re-abrir y asignar issues.
- Editar y eliminar comentarios de commits, pull requests e issues.
- Combinar y cerrar pull requests.
- Enviar pull requests desde forks de un repositorio.
- Crear y editar wikis.
- Crear y editar releases.
- Removerse a sí mismo como colaborador del repositorio.

2.6. GitHub en la educación

La amplia popularidad que ha venido manteniendo GitHub principalmente en la comunidad de desarrolladores de software, ha visibilizado también su importancia a interesados en resolver problemas y potenciar capacidades en otras áreas.

En el ámbito de la educación GitHub está capturando la atención de los actores de los procesos de enseñanza aprendizaje.

Esto es debido a que GitHub no solamente soporta las principales características presentes en los LMSs (Learning Management Systems) tradicionales, sino que también “provee características adicionales de las cuales los educadores y los estudiantes pueden beneficiarse: revelando transparencia de las actividades, estimulando la participación y la colaboración, y soportando el re uso y la compartición de los materiales y los conocimientos del curso” (Zagalsky, et al., 2015, p. 2).

Aunque no está diseñado específicamente para uso académico, GitHub “está diseñado teniendo los conceptos de texto, compartición, colaboración y libertad en mente, y debido a su naturaleza social y colaborativa, es un recurso consistente con la ideología de educación liberal que crecerá en utilidad a la medida que las comunidades académicas hagan uso de ella” (Shaffer, 2013).

Y aunque el uso de GitHub puede ser una opción interesante para los docentes, principalmente para aquellos que imparten cátedra en materias relacionadas con el desarrollo de software, se debe encontrar una estrategia para mitigar la complejidad que su uso puede involucrar para los estudiantes, pues “el uso de GitHub trae consigo retos, tal como la dificultad en la curva de aprendizaje de los estudiantes” (Zagalsky, et al., 2015, p. 2).

En un reportaje dedicado sobre el portal educativo de GitHub (Sawers, 2014) se estima que son ya 1.200 cursos y 70.000 estudiantes registrados en el servicio.

Existen algunas experiencias documentadas acerca del uso de GitHub en el área de la educación, la mayoría de ellas en el ámbito de desarrollo del software. Sin embargo y referido por algunos autores (Zagalsky, et al., 2015), el uso de herramientas de versionamiento

de código no debería ser restringido únicamente a carreras o materias dedicadas al desarrollo de software.

Otras carreras, principalmente las de ingeniería pueden ser beneficiadas por el uso de estas herramientas. Las experiencias positivas de docentes que ya hacen uso de manera regular de este tipo de herramientas pueden ser motivadores para que otros docentes independientemente de su área se sumen al grupo de aquellas que van encontrando grandes beneficios en su utilización.

En la Universidad de Valdosta, Estados Unidos, se realizó una investigación empírica del uso de Git en proyectos de software en la clase “CS 4900 Senior Seminar” (Xu, 2012). Este curso es análogo a una clase de “Proyecto Integrador” para estudiantes próximos a graduarse en una carrera de Ingeniería en Sistemas en nuestra localidad. En este tipo de clases el aprendizaje es basado en la ejecución de proyectos.

En el estudio referido, los resultados obtenidos por el uso de Git son positivos. Las ventajas son importantes, especialmente para estudiantes próximos a graduarse, pues participarán en un ambiente profesional en el cual posiblemente utilicen estas herramientas de versionamiento.

En los cursos de ingeniería el manejo de proyectos para los estudiantes generalmente es caótico, en un ambiente en el que se requiere coordinar las actividades y aportes de cada uno de los miembros del equipo.

El mantener el proyecto sincronizado y articulado es un proceso que involucra tiempo y esfuerzo de los estudiantes y que posiblemente involucre caer en errores, sea con una acción manual o valiéndose de herramientas como el correo electrónico u otras como Google Docs, Google Drive, One Drive, etc. El uso de Git y GitHub involucra ventajas en el proceso de recolección de asignaciones, volviendo el procedimiento más sencillo y ordenado que la recolección manual o el soportado por aplicaciones como las dispuestas por Google (Xu, 2012).

Josh Davis, ingeniero en los servicios web Elastic Beanstalk de Amazon, en la primavera del 2014, desempeñando el rol de asistente de aprendizaje en la materia Principios de Leguajes de Programación en una Universidad de los Estados Unidos, aplica un

procedimiento para la asignación de tareas y la evaluación de trabajos de sus estudiantes utilizando GitHub.

Su experimento lo lleva a desarrollar herramientas que ayuden con el registro de estudiantes y con el manejo de repositorios GitHub para las asignaciones de tareas de clase. Las experiencias de su aplicación de GitHub en el aula las describe en su blog personal (Josh, 2014).

En un estudio basado en entrevistas directas a docentes que utilizan GitHub en su aula de clases (Zagalsky, et al., 2015), sus autores describen que usando GitHub como una plataforma de presentación hizo sencillo a su entrevistados monitorear el progreso, la actividad y la participación de los estudiantes.

Hoy en día, estas iniciativas de utilización de herramientas tecnológicas para el seguimiento de proyectos pueden facilitarse gracias a los servicios en la nube existentes para el hospedaje de proyectos, eliminando la carga administrativa que involucra tener infraestructura, servidores y personal de soporte propios dentro de la institución (Lawrance, et al., 2013). Entre algunos de estos servicios se encuentran: Assembla, Bitbucket, Github, Google Code, ProjectLocker, Sourceforge y Unfuddle.

Assembla, Bitbucket y Github son tres de los servicios que utilizan Git. Si el hospedaje de un proyecto sobre uno de estos servicios tuviera algún inconveniente se puede hospedar rápidamente y sin complejidades el mismo repositorio Git local sobre otro de estos servicios. (Lawrance, et al., 2013)

Adicionalmente Assembla, Bitbucket and Github ofrecen un servicio gratuito para proyectos de uso académico en el que los repositorios y toda su información se mantiene privada y de uso exclusivo para los colaboradores del proyecto.

Y aunque los sistemas de versionamiento no pueden reemplazar del todo a un LMS, pues carecen de herramientas tales como pruebas y evaluaciones en línea, listas de distribución por email, manejo de paquetes como SCORM, etc. con herramientas como GitHub se puede cubrir y mejorar el flujo utilizado para la publicación de materiales y para las asignaciones de tareas.

El Gráfico 8-2 permite identificar las tareas potenciadas por GitHub por encima de un LMS tradicional:

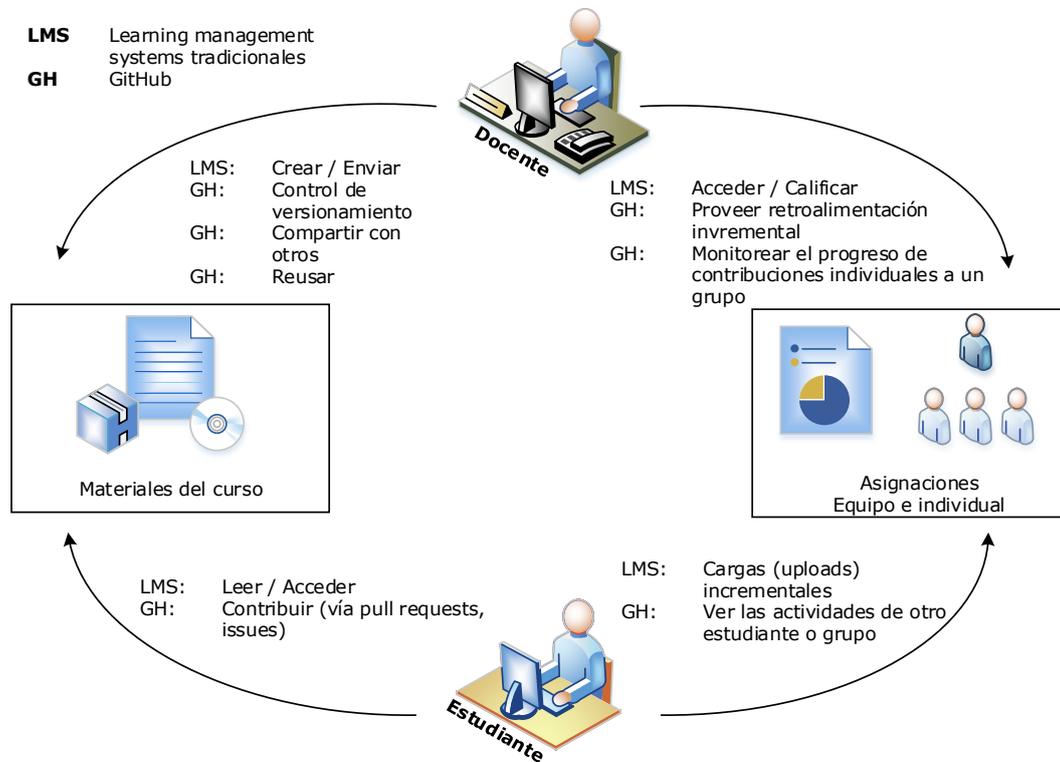


Gráfico 8-2: Tareas de LMSs potenciadas por GitHub

Fuente: (Zagalsky, et al., 2015, p. 5)

En su experiencia Lawrance (et al., 2013), recomienda que en un currículo de Ciencias de la Computación se debería trasladar la cátedra de versionamiento de código desde materias de alto nivel como la Ingeniería de Software hacia materias de niveles inferiores como fundamentos de programación.

Sumergir a los estudiantes en la práctica de control de versiones desde etapas tempranas es de mucho beneficio para todos los actores del proceso de enseñanza y aprendizaje en estas áreas académicas.

Pero no sólo los docentes pueden percibir los beneficios del uso de herramientas de versionamiento de código. En áreas como el desarrollo de software, los estudiantes que se han integrado al ámbito laboral y que han requerido: armar su CV, presentarse a una entrevista de trabajo, o abordar su trabajo utilizando estas herramientas, entienden y agradecen su uso temprano en el aula de clases (Lawrance, et al., 2013).

CAPÍTULO III

3. DISEÑO DE INVESTIGACIÓN

3.1. Diseño de la investigación

La presente investigación es cuasi experimental. Se pretende modificar las condiciones habituales en las que un docente realiza el seguimiento del proyecto de fin de curso de Lenguajes de Programación en las carreras de Ingeniería Electrónica de la FIE, sin tener un control total sobre todo el contexto del experimento.

Los grupos de estudiantes sobre los cuales se ha realizado el estudio son los matriculados en la materia de Lenguajes de Programación I en los periodos marzo – agosto 2014 y marzo – agosto 2015 en dos (2) carreras de Ingeniería Electrónica diferentes.

En este caso las muestras no pueden ser consideradas como aleatorias. Por el proceso de matriculación los grupos son previamente constituidos.

El primer grupo ha utilizado un esquema tradicional para el seguimiento del proyecto de fin de curso, el segundo grupo ha utilizado el método MESEPP definido y propuesto en este proyecto de investigación.

3.2. Tipo de investigación

El presente trabajo de investigación es de tipo exploratorio. Se pretende incluir un método en el seguimiento del proyecto de fin de curso de la materia de Lenguajes de Programación I, en búsqueda del incremento de su calidad.

En el proyecto no se cuestiona el tiempo y el esfuerzo que el uso del método propuesto implica para el docente. Tampoco se cuestiona la complejidad que podría significar para los estudiantes el uso de una nueva plataforma como GitHub. Nuevos estudios podrán profundizar estas temáticas.

3.3. Población

La población del estudio está constituida por los estudiantes de la materia de Lenguajes de Programación de las carreras de Ingeniería Electrónica de la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo. Cada nuevo semestre existen nuevos estudiantes en esta población, a la cual se la entiende como de número indefinido.

3.4. Muestra

Entendiéndose la población del estudio, los grupos de estudiantes sobre los cuales se ha realizado el experimento han sido seleccionados por los grupos previamente constituidos en base a un proceso de matriculación.

Se ha tomado el conjunto de estudiantes matriculados en la materia de Lenguajes de Programación en la Escuela de Ingeniería Electrónica en Telecomunicaciones y Redes (EIEETR) en el periodo marzo 2014 – agosto 2014 y el conjunto de estudiantes matriculados en la materia de Lenguajes de Programación en la Escuela de Ingeniería Electrónica en Control y Redes Industriales (EIECRI) en el periodo marzo 2015 – agosto 2015.

Mientras el primer grupo ha utilizado un esquema tradicional para el seguimiento del proyecto de fin de curso, en el segundo grupo se ha utilizado el método MESEPP definido y propuesto en este proyecto de investigación.

3.5. Métodos

En la investigación ha sido utilizado el método Científico - Modelo general, que incluye el planteamiento del problema, la formulación de la hipótesis, el levantamiento de la información, el análisis e interpretación de datos, la comprobación de la hipótesis y la difusión de los resultados de la investigación.

También ha sido utilizado el método deductivo para definir el método MESEPP, específico para el seguimiento y evaluación de proyectos de materias de lenguajes de

programación, basándose en procedimientos generales utilizados para seguimiento de proyectos fuera del área de la educación.

Adicionalmente el método inductivo ha sido utilizado para desarrollar las conclusiones generales presentadas en este proyecto de investigación, basadas en los resultados específicos del experimento sobre los grupos muestrales.

3.6. Técnicas

Por el ámbito de la experimentación, la observación es la técnica principal utilizada en esta investigación. La misma ha permitido la ponderación de cada uno de los indicadores de la calidad de los proyectos de fin de curso.

Para la comprobación de la hipótesis se ha utilizado el análisis estadístico y la estadística inferencial.

3.7. Instrumentos de evaluación

El instrumento de evaluación, utilizado para calcular la calidad de los proyectos de fin de cursos en los dos grupos muestrales, es la rúbrica de evaluación presentada en el Anexo A. Este instrumento representa a su vez el ARTEFACTO 07: Rúbrica de evaluación del método MESEPP, nuevo método propuesto a través del presente trabajo de investigación.

Este instrumento ha sido validado previo a su aplicación por los miembros del tribunal del presente trabajo de investigación.

3.8. Aplicación del método

Dentro del ámbito de las carreras de Ingeniería Electrónica de la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo, es viable la aplicación del método propuesto.

El método no tiene requerimientos sofisticados, es abierto y gratuito. Para el acceso y utilización de la plataforma GitHub son requeridos únicamente un computador u otro dispositivo y acceso a internet, los dos son instrumentos garantizados en la ESPOCH.

El método define e incluye todos los artefactos necesarios para su aplicación. Para la ejecución de las actividades es requerida la participación del docente y de los estudiantes, actividades enmarcadas siempre en el ámbito curricular de la materia de Lenguajes de Programación.

Si la aplicación del nuevo método en el grupo de prueba mejora la calidad de los proyectos, el método propuesto por este proyecto de investigación será validado y así mismo la hipótesis de investigación será aceptada.

3.9. Procedimientos

Para la aplicación y validación del nuevo método se ha tomado como muestra el grupo de veinte y ocho (28) estudiantes de la Escuela de Ingeniería Electrónica en Control y Redes Industriales de la ESPOCH matriculados en la materia de Lenguajes de Programación I en el periodo lectivo marzo 2015 – agosto 2015.

El procedimiento permitirá comparar los resultados de calidad de los proyectos respecto a la calidad de los proyectos de los veinte y ocho (28) estudiantes de la Escuela de Ingeniería Electrónica en Telecomunicaciones y Redes matriculados en la misma materia de Lenguajes de Programación I en el periodo lectivo marzo 2014 – agosto 2014.

Este último grupo considerado para el estudio ha utilizado un esquema tradicional para el seguimiento de los proyectos de fin de curso. En el cuál el docente envía el proyecto, los estudiantes definen el tiempo que dedican al proyecto y el ritmo de avance. Siempre tienen a disposición al profesor en horas de consulta para resolver alguna inquietud.

El proyecto se presenta en un día específico al final del semestre a través de una exposición que incluye diapositivas, una exposición oral y la presentación del programa computacional.

Para el experimento, un proyecto ha sido enviado a todos los equipos del primer grupo de estudio y otro proyecto diferente ha sido enviado a todos los equipos del segundo grupo de estudio.

A ambos grupos se los ha valorado con la misma rúbrica de evaluación. Las ponderaciones registradas en la rúbrica han sido analizadas para obtener conclusiones y realizar la comprobación de la hipótesis de investigación.

3.10. Variables e indicadores

En base a la hipótesis de investigación planteada en el presente trabajo la cual dicta: *La utilización de un método que incorpore a GitHub como herramienta de soporte tecnológico en el seguimiento de proyectos de fin de curso de materias de Lenguajes de Programación en carreras de Ingeniería Electrónica mejora la calidad de los proyectos*, se determinan las siguientes variables:

Variable Independiente: *Utilización de un método que incorpore GitHub.*

Variable Dependiente: *Calidad del proyecto de fin de curso.*

Habiéndose determinado las variables, se presente a continuación el detalle de su operacionalización conceptual y metodológica. Nótese que los indicadores de la variable dependiente *Calidad del proyecto de fin de curso* coinciden con los ponderadores de la rúbrica de evaluación sugerida por el nuevo método propuesto y que constituye el instrumento de evaluación en el presente trabajo de investigación.

Operacionalización conceptual

Tabla 1-3: Operacionalización conceptual de variables

Variable	Tipo	Concepto
<i>Utilización de un método que incorpore GitHub.</i>	- Independiente - Cualitativa	Inclusión de un método como un componente para el seguimiento de proyectos de fin de curso.
<i>Calidad del proyecto de fin de curso.</i>	- Dependiente - Cuantitativa	Valor porcentual de eficiencia y de eficacia del proyecto de fin de curso.

Realizado por: Javier Salazar, 2015

Operacionalización metodológica

Tabla 2-3: Operacionalización metodológica de variables

Variable	Indicador	Índice	Ponderador	Técnica	Fuente de verificación / Instrumento
<i>Utilización de un método que incorpore GitHub.</i>	Uso	¿Se ha utilizado un método que incorpore a GitHub para el seguimiento del proyecto de fin de curso?	- Sí - No	Entrevista	Encuesta
<i>Calidad del proyecto de fin de curso</i>	Eficiencia	¿El equipo repartió entre sus integrantes equitativamente el esfuerzo del proyecto?	- No 0% - En Parte 10% - Sí 20%	Observación	Código fuente proyecto
		¿El equipo repartió equitativamente el tiempo dedicado al proyecto?	- No 0% - En parte 10% - Sí 20%	Observación	Código fuente proyecto

Realizado por: Javier Salazar, 2015

Tabla 2-3: Operacionalización metodológica de variables (Continuación)

Variable	Indicador	Índice	Ponderador	Técnica	Fuente de verificación / Instrumento
<i>Calidad del proyecto de fin de curso</i>	Eficacia	¿El equipo ha completado todos los requerimientos especificados para el proyecto?	- No 0% - Parcialmente 10% - Sí 20%	Observación	Código fuente proyecto
		¿Las partes de código creados, respetan la teoría y consideraciones presentadas en clase?	- No 0% - Parcialmente 10% - Sí 20%	Observación	Código fuente proyecto
		¿La presentación final del proyecto fue satisfactoria?	- No 0% - Parcialmente 5% - Sí 10%	Observación	Presentación final del proyecto
		¿Los estudiantes respondieron correctamente la pregunta realizada por el profesor en el momento de su presentación final?	- No 0% - Parcialmente 5% - Sí 10%	Observación	Presentación final del proyecto

Realizado por: Javier Salazar, 2015

3.11. Análisis de las variables

El presente trabajo de investigación propone un nuevo método para el seguimiento y evaluación de los proyectos de fin de curso de la materia de Lenguajes de Programación. Para validarlo, el nuevo método ha sido aplicado a un grupo de estudiantes. Como sugiere la hipótesis, se espera que la aplicación del nuevo método permita mejorar la calidad de los proyectos.

3.11.1. Indicadores de la variable independiente

Para la comprobación de la hipótesis, en el primer grupo de estudiantes no se ha utilizado un método que incorpore GitHub y en el segundo grupo de estudiantes sí se ha utilizado un método que incorpore GitHub en la ejecución de los proyectos de fin de curso. Esta incorporación o no del nuevo método representa la variable independiente del estudio. A continuación se describen los indicadores.

Indicador Utilización de un método que incorpore GitHub

Este indicador responde a la pregunta ¿Se ha utilizado un método que incorpore a GitHub para el seguimiento del proyecto de fin de curso? La respuesta a esta pregunta es dicotómica. Para el primer grupo de estudio la respuesta es no, para el segundo grupo de estudio la respuesta es sí.

3.11.2. Indicadores de la variable dependiente

Para determinar si la calidad de los proyectos mejora de un grupo de estudio a otro, para ambos se ha determinado la calidad de sus proyectos, utilizando el mismo conjunto de indicadores. La calidad representa la variable dependiente del estudio. A continuación se describen los indicadores.

Indicador Eficiencia

En la ejecución de proyectos de desarrollo mientras menos tiempo, esfuerzo y dinero se involucre en el proyecto, su ejecución es más eficiente.

Para determinar su analogía en proyectos de la academia, en donde el dinero no es un ponderador, el tiempo y las características del proyecto son establecidos de manera fija y constante, para el presente proyecto la eficiencia se la ha querido representar en base a la respuesta a las siguientes preguntas: ¿El equipo repartió entre sus integrantes equitativamente el esfuerzo del proyecto?, ¿El equipo repartió equitativamente el tiempo dedicado al proyecto?

Indicador Eficacia

En la ejecución de proyectos la eficacia puede ser vista como el cumplimiento efectivo de los objetivos prefijados para el proyecto.

En su analogía para los proyectos de fin de curso en las materias de Lenguajes de Programación la eficacia se la ha querido representar en base a la respuesta a las siguientes preguntas ¿El equipo ha completado todos los requerimientos especificados para el proyecto?, ¿Las partes de código creados, respetan la teoría y consideraciones presentadas en clase?, ¿La presentación final del proyecto fue satisfactoria?, ¿Los estudiantes respondieron correctamente la pregunta realizada por el profesor en el momento de su presentación final?

CAPÍTULO IV

4. RESULTADOS Y DISCUSIÓN

4.1. Procedimiento general

En el presente trabajo de investigación se propone el método MESEPP para el seguimiento y evaluación de los proyectos de fin de curso de la materia de Lenguajes de Programación de las carreras de Ingeniería Electrónica.

A través del estudio se pretende demostrar que la utilización de este nuevo método permite mejorar la calidad de los proyectos de fin de curso desarrollados y presentados por los estudiantes de la materia.

Para lograr esto, se utilizó una estrategia de contraste para la diferencia de medias de dos (2) grupos distintos de muestras distintas de estudiantes dentro de una misma población.

En un primer grupo de veinte y ocho (28) estudiantes pertenecientes al periodo lectivo marzo – agosto del 2014, se utilizó un seguimiento tradicional, basado en revisiones en horas de clase, asesoramiento y soporte en horas de atención a estudiantes y una revisión mayor del proyecto el día de la presentación final.

Para la evaluación se utilizó la rúbrica sugerida por MESEPP. Las rúbricas de evaluación con los resultados obtenidos por cada grupo y estudiante se presentan en el Anexo G. En este capítulo y para claridad del lector a este grupo de estudiantes se los denominará como “Periodo 2014”.

En un segundo grupo de veinte y ocho (28) estudiantes pertenecientes al periodo lectivo marzo – agosto 2015, se utilizó el método MESEPP, por un periodo total de veinte y dos (22) días. Los documentos resultados de la aplicación del método se presentan en la sección de anexos del presente documento. A este grupo de estudiantes se los denominará como “Periodo 2015”.

El uso de la misma rúbrica de evaluación para los dos (2) grupos de estudiantes permite contrastar los resultados. El valor porcentual total obtenido por un estudiante representa, según lo sugerido por MESEPP, el grado de calidad con la que el estudiante ejecutó el proyecto.

El promedio de los grados de calidad de todos los estudiantes del periodo 2014 representa el grado de calidad de los proyectos para dicho periodo lectivo. La misma ponderación aplica para el periodo 2015.

4.2. Presentación de resultados

Los resultados obtenidos revelan de manera directa que la utilización del método MESEPP no mejora la calidad de los proyectos de fin de curso de la materia de lenguajes de programación.

El Gráfico 1-4 presenta el grado de calidad obtenido en cada uno de los grupos de estudio.

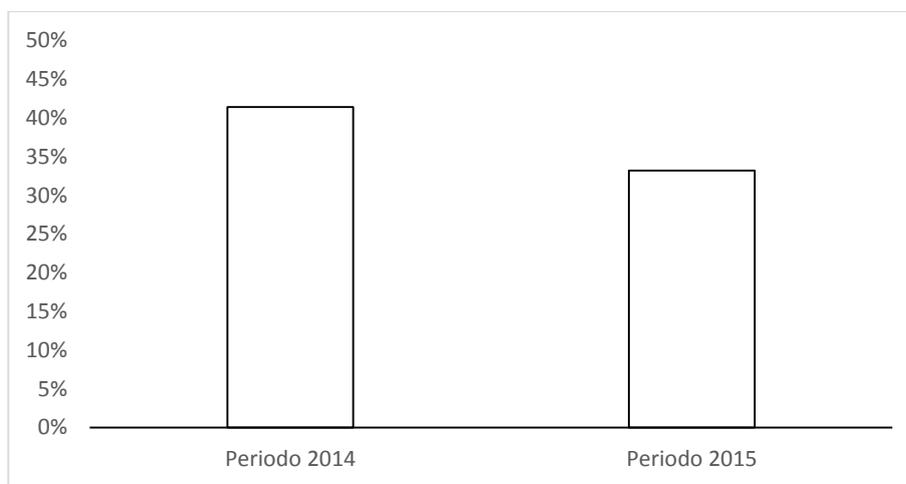


Gráfico 1-4: Total de calidad en proyectos periodo 2014 y 2015

Realizado por: Javier Salazar, 2015

El grado de calidad del Periodo 2014 con un valor del 41% es mayor que el ponderado para el Periodo 2015 que apenas alcanza un valor del 33%. Basado en estos valores cuantitativos se puede afirmar que la utilización del método propuesto en el periodo 2015 no mejoró la calidad de los proyectos.

A primera vista esto haría suponer también que el método tradicional para el seguimiento y evaluación de proyectos es mejor que el método propuesto, sin embargo, una lectura más detallada permite discutir y analizar algunos detalles:

- A pesar de que para los dos (2) grupos de estudio se utilizó la misma rúbrica de evaluación, los procedimientos para su ponderación representan dos percepciones distintas:

- o Para el periodo 2014 la ponderación es más subjetiva, debido a que se tiene menos información disponible para el docente. Las revisiones de aula y la presentación final son insuficientes para tener una percepción objetiva del trabajo individual y grupal de los estudiantes.

La rúbrica de evaluación es llenada con base a esta subjetividad y puede ser sobrevalorada.

- o Para el periodo 2015 el método propuesto brinda objetividad, ofreciendo al docente una trazabilidad completa del trabajo individual y grupal de sus estudiantes, no sólo en periodos cortos de aula o en la presentación final del proyecto sino durante todo el periodo de ejecución.

Esta visibilidad del trabajo de los estudiantes recae en una ponderación más acertada, más real, que evita sobrestimar el trabajo de estudiantes que no aporten con el proyecto, de equipos donde un estudiante sea el único ejecutor o de equipos que distribuyan ineficientemente su esfuerzo o su tiempo.

La rúbrica de evaluación es llenada con base a esta objetividad y sus resultados pueden disminuir considerablemente respecto a una ponderación subjetiva.

Se puede evidenciar también que en general la calidad de los proyectos de los estudiantes es baja, pues siendo que el valor de la más alta calidad esperado y deseado por el docente es del 100%, ninguno de los dos periodos lectivos supera siquiera el 50%.

Componente Grupal

La rúbrica tiene componentes de evaluación tanto grupal como individual. El Gráfico 2-4 presenta la comparación entre los componentes grupales de los dos (2) periodos de

estudio. Nótese que el valor máximo que pueden alcanzar las barras del siguiente gráfico son 70% que es el porcentaje sugerido por MESEPP para este componente grupal.

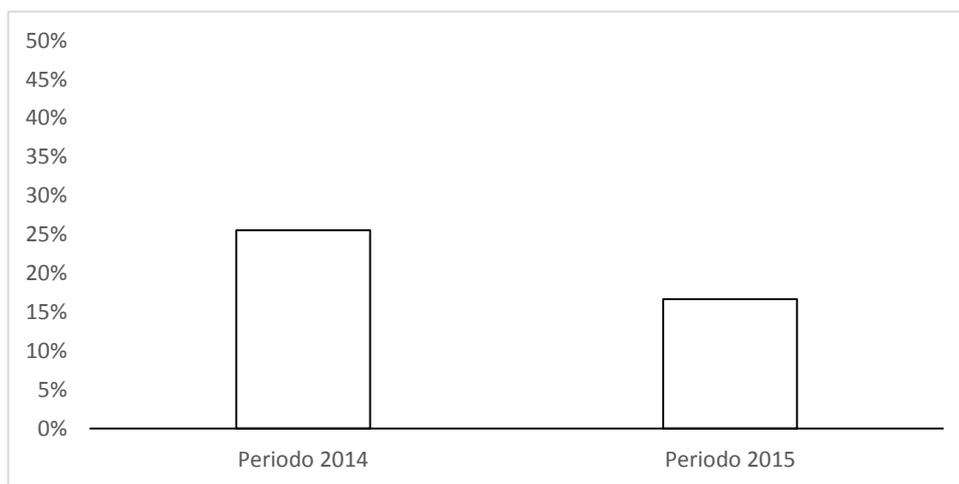


Gráfico 2-4: Componente grupal de calidad proyectos periodo 2014 y 2015

Realizado por: Javier Salazar, 2015

En el Gráfico 4-2, al igual que en el general, se puede afirmar que el grado de calidad sigue siendo mejor para el periodo 2014 con un valor del 26% por encima del periodo 2015 con un valor del 17%. Y mientras estos valores cuantitativos permiten realizar esta afirmación, la diferencia de objetividad entre los dos periodos antes discutida, permite considerar el punto de vista que, los resultados en el periodo 2015 simplemente evidencian con objetividad la ineficiencia del trabajo en equipo de los estudiantes.

Desglosando el componente grupal a más detalle, tal como lo presenta el Gráfico 3-4, se puede evidenciar que en los dos (2) periodos los estudiantes reparten ineficientemente el esfuerzo del proyecto con un valor ponderado del 2% sobre 20%. Esto significa que dentro de un mismo equipo hay estudiantes que hacen más esfuerzo y completan más cantidad de funcionalidad del proyecto, mientras que otros aportan poco o en ocasiones nada al proyecto.

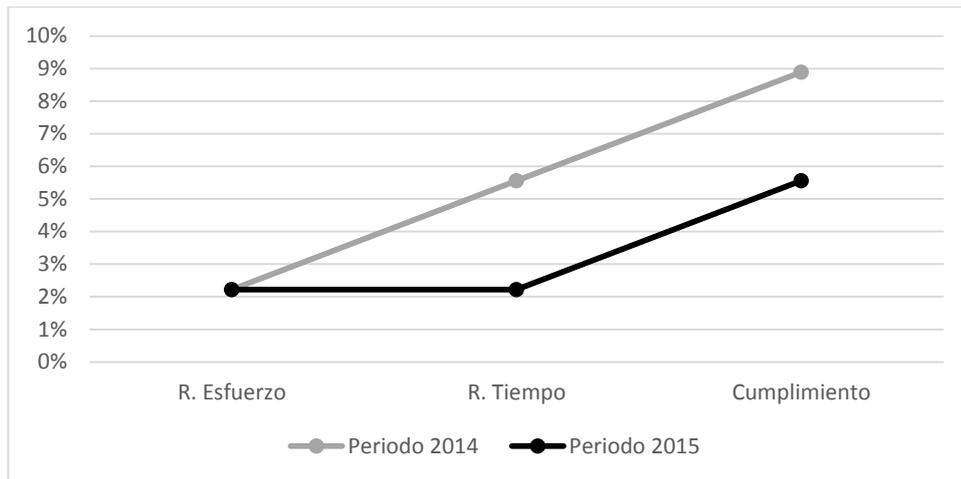


Gráfico 3-4: Índices del componente grupal de calidad de los proyectos periodo 2014 y 2015

Realizado por: Javier Salazar, 2015

La repartición del tiempo se ha ponderado mejor para el periodo 2014 con un valor de 6% sobre 20% con respecto al 2% del periodo 2015. Un equipo es mejor ponderado cuando realiza avances periódicos, progresivos y sostenidos del proyecto.

Se puede observar sin embargo que estas ponderaciones de la repartición del tiempo son bajas para los dos periodos, lo que da a notar que los estudiantes no reparten bien su tiempo, y como se analizará más adelante, tienden a realizar el mayor esfuerzo en los últimos días del periodo designado para la ejecución del proyecto.

Y así como los índices de repartición de esfuerzo y de tiempo refieren la eficiencia con la que los estudiantes ejecutan el proyecto, el índice del cumplimiento presentado en el Gráfico 3-4 refiere la eficacia. Es la capacidad de los estudiantes de cumplir con todos los requerimientos solicitados para el proyecto.

En esta valoración, la percepción de cumplimiento también fue mayor para el periodo 2014 con un valor del 9% sobre 20% contra el valor del 6% del año 2015. Nótese así mismo que estos valores son bajos, pues en ninguno de los dos (2) periodos los estudiantes, en promedio, cumplieron siquiera el 50% de los requerimientos solicitados para sus proyectos.

Para el periodo 2015, gracias al uso MESEPP, se puede presentar información adicional no visible en la rúbrica de evaluación. El Gráfico 4-4 presenta el cumplimiento de los requerimientos (del 0% al 100% de requerimientos) de los proyectos por cada uno de los equipos.

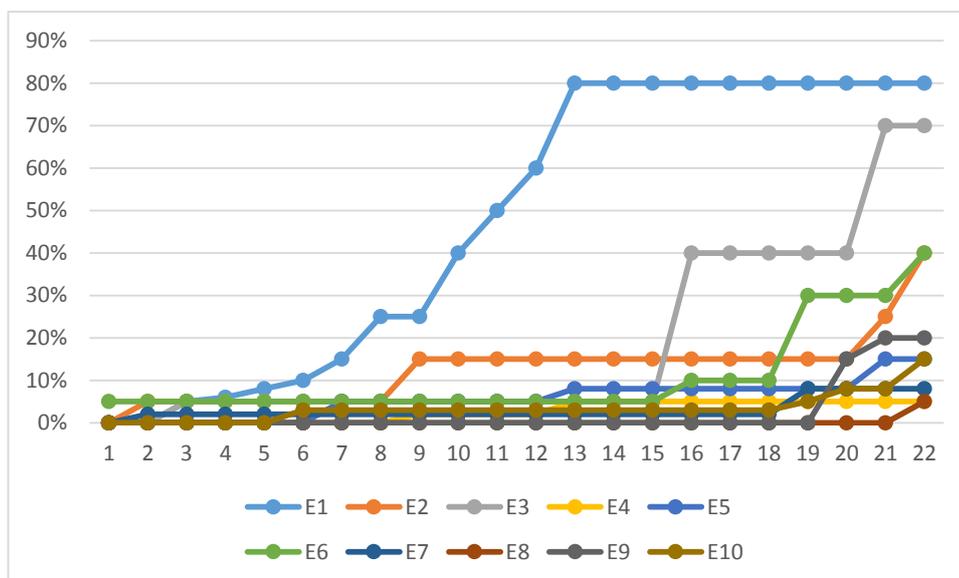


Gráfico 4-1: Porcentaje del cumplimiento de requerimientos por día de proyectos del periodo 2015.

Realizado por: Javier Salazar, 2015

El equipo 1 (E1) es un caso excepcional, realiza un trabajo dedicado desde el día 1 hasta el día 13, día en el que completa ya el 80% del proyecto.

El equipo 3 (E3) realiza un trabajo incremental la última semana. En esta semana completa ya el 70% del proyecto.

El resto de equipos presentan un patrón de ejecución similar: no realizan ningún avance sino hasta los últimos días, en donde completan un bajo porcentaje del proyecto menor o igual al 40%.

Esta información evidencia los hábitos en la ejecución de los proyectos por parte de los estudiantes, donde las actividades se realizan en los últimos días acordados con el

docente. En esos escasos días, el esfuerzo y el tiempo son insuficientes para cubrir con el 100% de los requerimientos exigidos por el proyecto.

Componente Individual

El Gráfico 5-4 presenta la calidad de los proyectos en su componente individual, para los dos (2) periodos lectivos en estudio. En el gráfico el valor máximo de las barras puede alcanzar el 30% que es el valor sugerido por MESEPP:

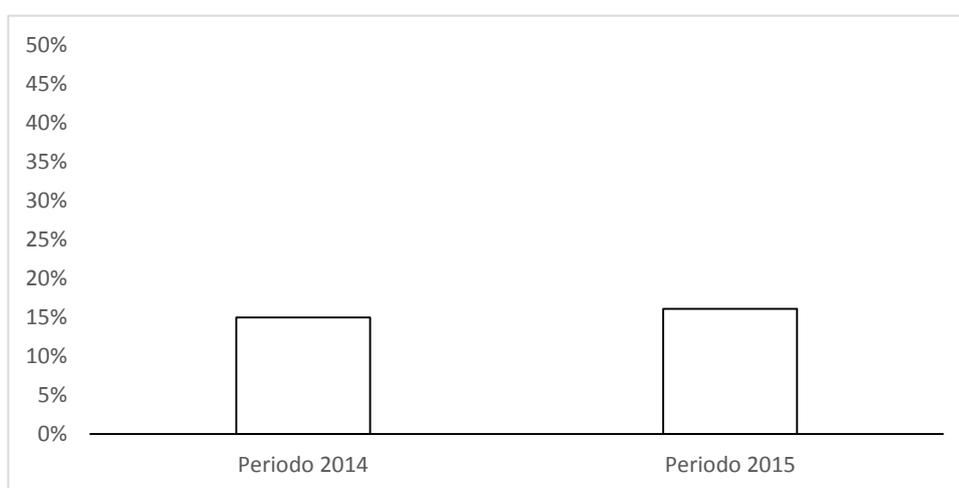


Gráfico 5-4: Componente individual de calidad de los proyectos periodo 2014 y 2015.

Realizado por: Javier Salazar, 2015

Con el Gráfico 5-4 se puede afirmar que en lo que respecta al componente individual, es decir el trabajo por estudiante, el periodo 2015 presenta mejores resultados con un valor del 16% siendo un punto porcentual mayor respecto al 15% del periodo 2014.

Nótese sin embargo, que esto no significa que todos los estudiantes hayan hecho un mejor trabajo. La información detallada recogida en el periodo 2015, visibiliza que el trabajo destacado de unos pocos estudiantes ha elevado el valor de la media presentada en el gráfico.

Si desglosamos este componente individual, como se presenta en el Gráfico 6-4, podemos notar que para el periodo 2015 la calidad del código es superior con un valor de 11% sobre 20% por encima del valor de 8% sobre 20% del periodo 2014. El índice de código indica si los estudiantes han utilizado los conceptos de programación enseñados en clase y los han utilizado bien.

Por otro lado el índice que pondera el desenvolvimiento individual de los estudiantes en su presentación final fue mejor para el periodo 2014 con un valor de 7% sobre 10% por encima del valor de 5% sobre 10% del periodo 2015.

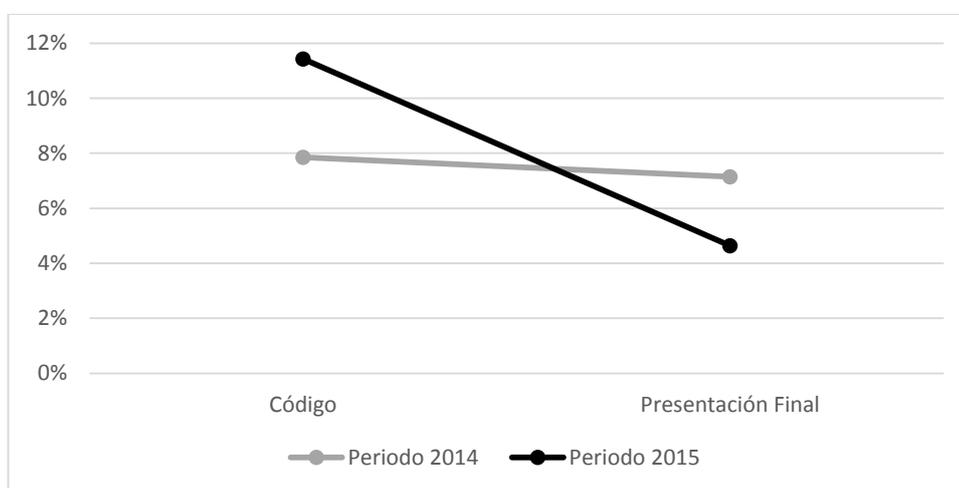


Gráfico 6-4: Índices del componente individual de calidad de proyectos periodo 2014 y 2015

Realizado por: Javier Salazar, 2015

Actividad en el Periodo 2015

Gracias a la utilización de MESEPP, se ha podido obtener información más rica para el periodo 2015. Esta información revela el trabajo destacado de unos pocos estudiantes así como el esfuerzo concentrado en los últimos días disponibles para la ejecución del proyecto.

A continuación se presentan los gráficos en los que el eje de las Y representa el nivel de actividad de un estudiante con valores: 0 ninguna, 1 baja, 2 media y 3 alta y el eje de las

X representa los días designados para el proyecto que para el periodo 2015 son de un total de veinte y dos (22).

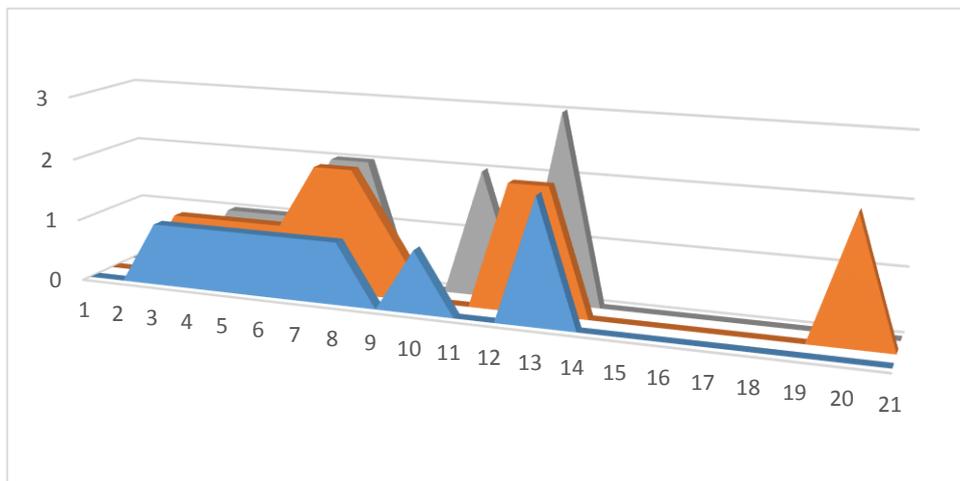


Gráfico 7-4: Nivel de actividad Equipo 1

Realizado por: Javier Salazar, 2015

El Gráfico 7-4 es del equipo con mejor grado de calidad en el periodo 2015. Se puede observar un trabajo desde el inicio del proyecto por parte de los tres participantes. Al día 13 tenían cubierto ya el 80% del proyecto. Las buenas ponderaciones obtenidas por estos tres (3) estudiantes permiten de cierta manera mejorar los resultados en las medias del periodo 2015.

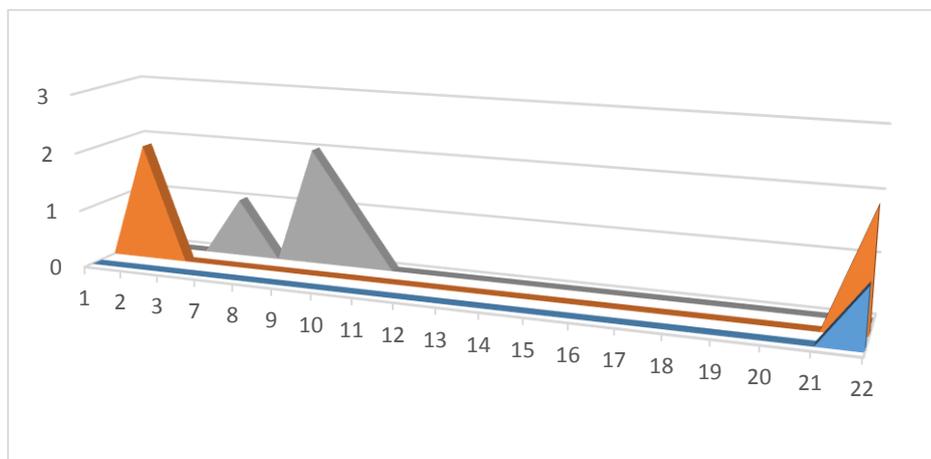


Gráfico 8-4: Nivel de actividad Equipo 2

Realizado por: Javier Salazar, 2015

El Gráfico 8-4 representa un equipo que realizó un esfuerzo mínimo. Se puede apreciar algo de esfuerzo en los primeros días y algo de esfuerzo en los últimos.

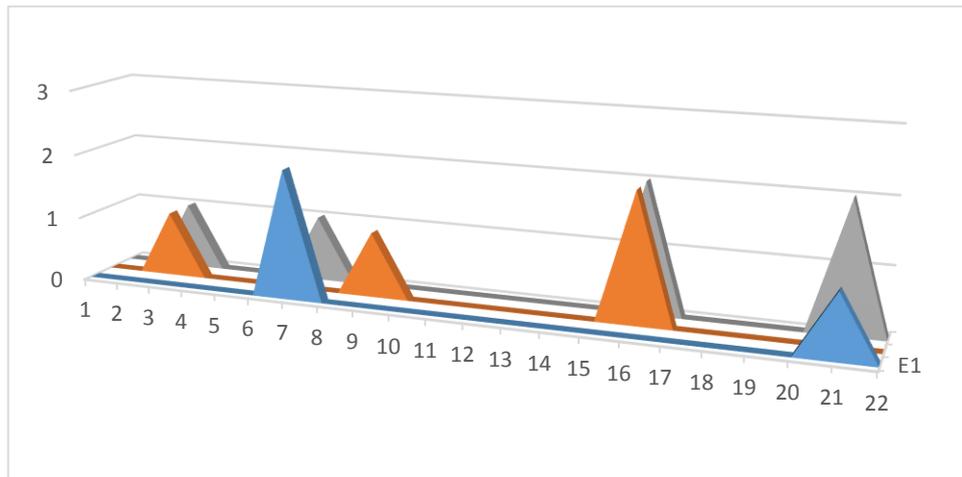


Gráfico 9-4: Nivel de actividad Equipo 3

Realizado por: Javier Salazar, 2015

El Gráfico 9-4 es el de un equipo en donde todos los integrantes participan, sin embargo su participación es escasa, y aunque repartida durante el periodo de ejecución del proyecto, no es continuada y sostenida.

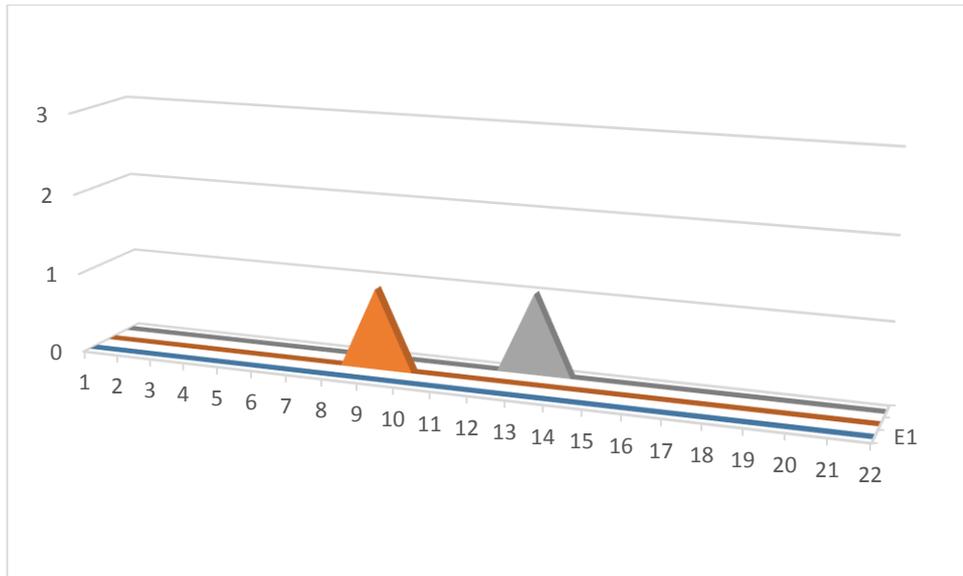


Gráfico 10-4: Nivel de actividad Equipo 4

Realizado por: Javier Salazar, 2015

El Gráfico 10-4 es de un equipo en el que dos integrantes interactuaron únicamente una vez en el proyecto, en estas dos ocasiones su participación es baja e insignificante. Un tercer miembro del equipo definitivamente no participa.

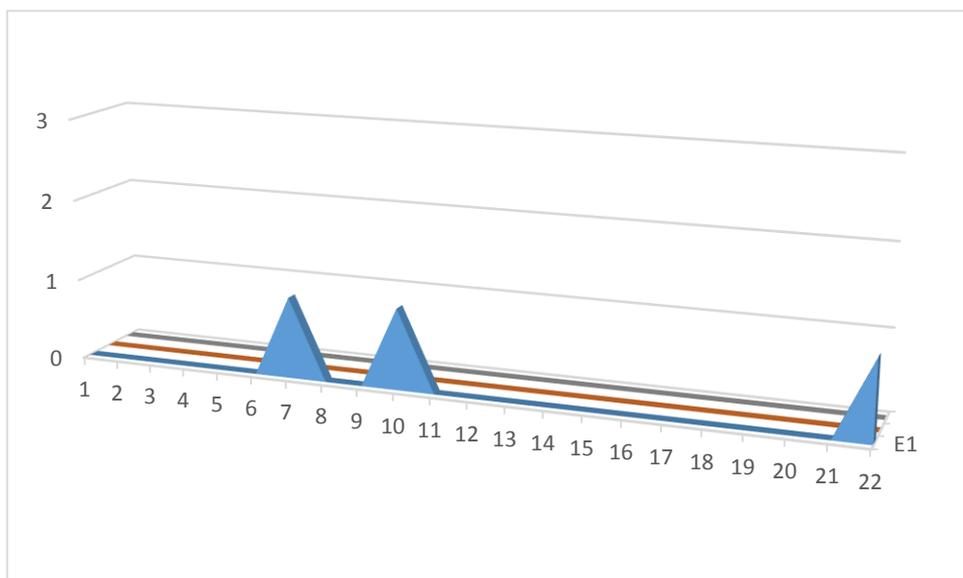


Gráfico 11-4: Nivel de actividad Equipo 5

Realizado por: Javier Salazar, 2015

El Gráfico 11-4 muestra un equipo en el que participa un único miembro. Su participación sin embargo es escasa. Nótese como se realiza una participación en los últimos días del proyecto.

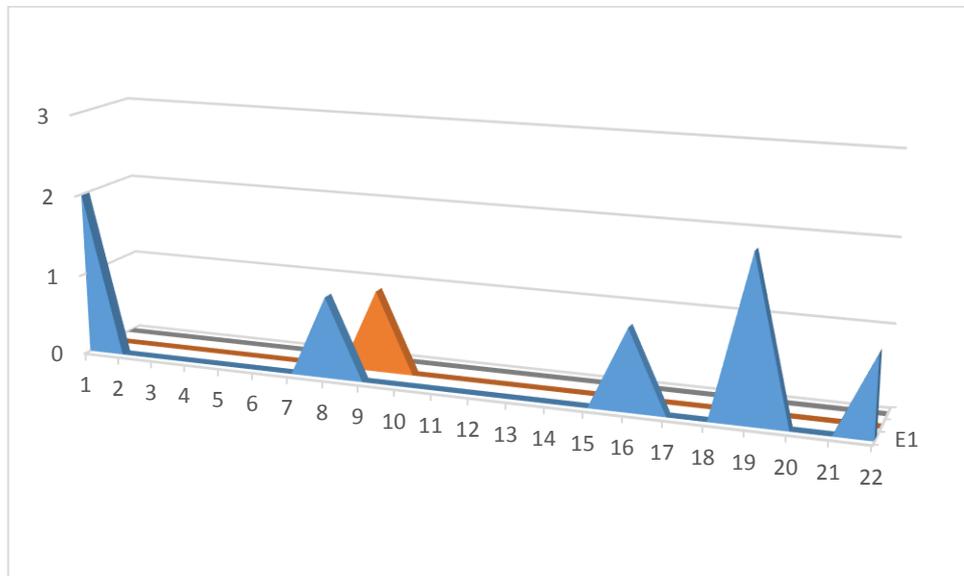


Gráfico 12-4: Nivel de actividad Equipo 6

Realizado por: Javier Salazar, 2015

El Gráfico 12-4 muestra un equipo en el que sólo un estudiante es el protagonista. Su esfuerzo es medio y no continuado. Nótese como en este caso también se realiza un gran esfuerzo en los últimos días del proyecto.

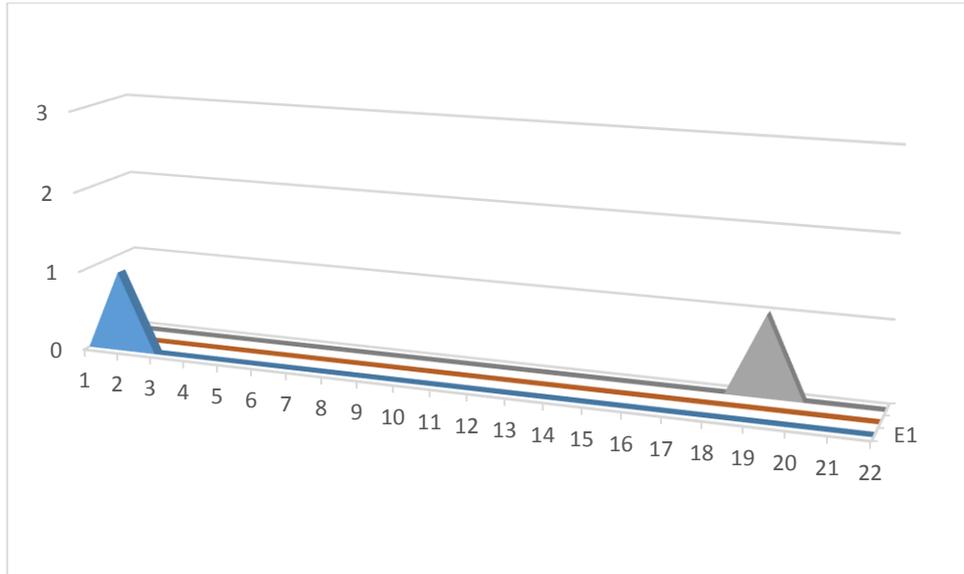


Gráfico 13-4: Nivel de actividad Equipo 7

Realizado por: Javier Salazar, 2015

El Gráfico 13-4 visualiza un equipo que no hizo ningún esfuerzo sobre el proyecto. Un miembro tiene una participación baja al inicio del proyecto, otro tiene una participación baja en los últimos días del proyecto. Un tercero no participa.

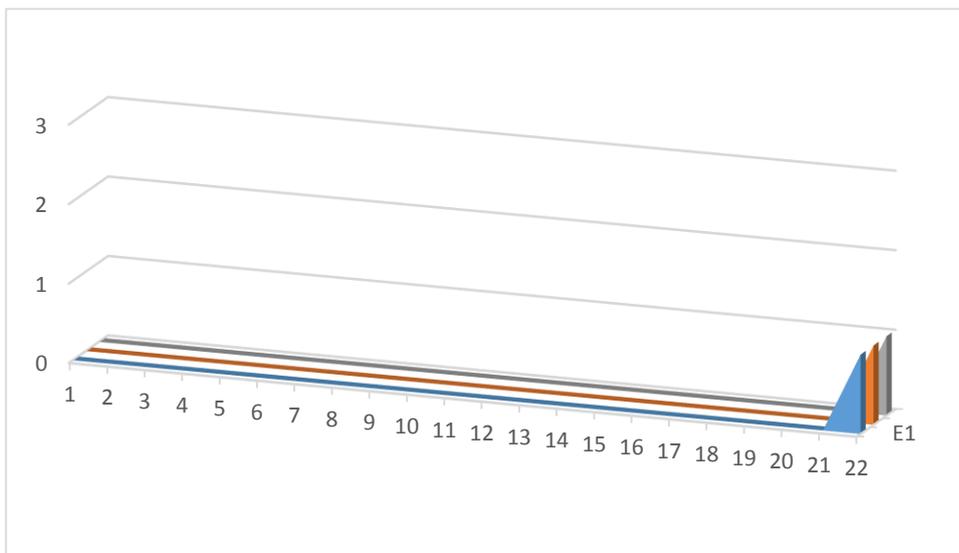


Gráfico 14-4: Nivel de actividad Equipo 8

Realizado por: Javier Salazar, 2015

El Gráfico 14-4 es un ejemplo de tres (3) miembros de un equipo que decidieron trabajar los últimos días del proyecto. Sin embargo su participación es baja y obviamente insuficiente para cubrir los requerimientos del proyecto.

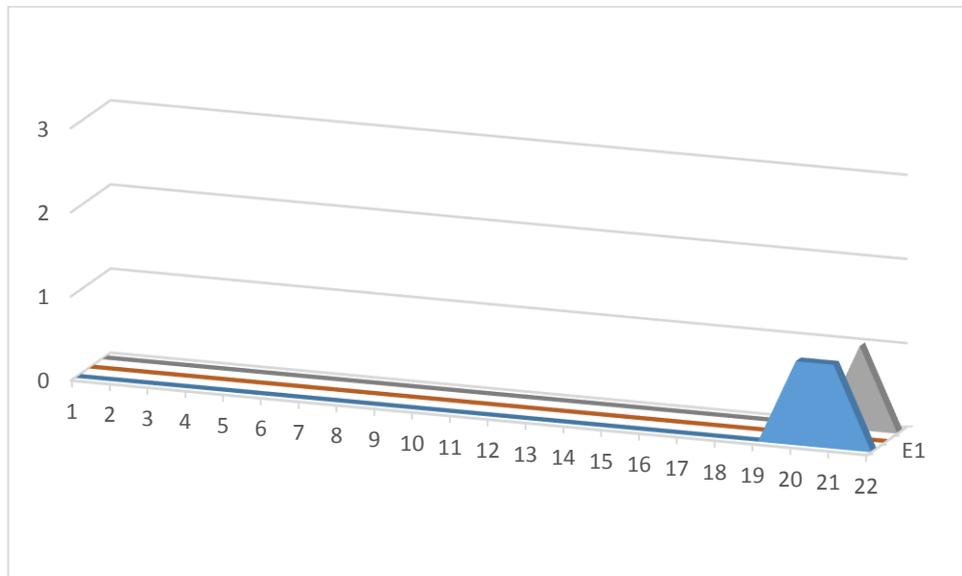


Gráfico 15-4: Nivel de actividad Equipo 9

Realizado por: Javier Salazar, 2015

El Gráfico 15-4 es un ejemplo más de un equipo que decidió participar en los últimos días del proyecto. En este caso dos son los estudiantes que realizan una baja participación. Un tercer integrante no participa en el proyecto.

4.3. Demostración de la hipótesis

A continuación se presenta de manera sistematizada el procedimiento aplicado para la demostración de la hipótesis:

4.3.1. Planteamiento

Hipótesis Nula H_0 : La utilización de un método que incorpore a GitHub como herramienta de soporte tecnológico en el seguimiento de proyectos de fin de curso de materias de Lenguajes de Programación en carreras de Ingeniería Electrónica no mejora la calidad de los proyectos.

Hipótesis de investigación H_a : La utilización de un método que incorpore a GitHub como herramienta de soporte tecnológico en el seguimiento de proyectos de fin de curso de materias de Lenguajes de Programación en carreras de ingeniería electrónica mejora la calidad de los proyectos.

La representación estadística de la hipótesis nula y de la alternativa o de investigación sería el de un caso unilateral, tal como sigue:

$$H_0: \mu_2 \leq \mu_1$$

$$H_a: \mu_2 > \mu_1$$

4.3.2. Población

La población del presente estudio son los estudiantes de la Escuela de Ingeniería Electrónica en Telecomunicaciones y Redes y los estudiantes de la Escuela de Ingeniería Electrónica en Control y Redes Industriales de la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo, matriculados en la materia de Lenguajes de Programación I.

Se supone la distribución de la probabilidad para la población como una distribución normal.

4.3.3. Selección del nivel de significación

Para el presente estudio se ha utilizado un nivel de significación de $\alpha = 0.05$ (5%)

4.3.4. Descripción de la muestra

Para el estudio han sido seleccionados dos grupos muestrales. En el siguiente cuadro se describe sus características:

Tabla 1-4: Grupos muestrales de estudiantes seleccionados para el estudio.

	Grupo 1	Grupo 2
Escuela	Escuela de Ingeniería Electrónica en Telecomunicaciones y Redes	Escuela de Ingeniería Electrónica en Control y Redes Industriales
Periodo Lectivo	marzo 2014 – agosto 2014	marzo 2015 – agosto 2015
Semestre	Segundo	Segundo
Materia	Lenguajes de Programación I	Lenguajes de Programación I
Syllabus	Mismo que del Grupo 2	Mismo que del Grupo 1
Número de estudiantes que completaron el semestre y que fueron finalmente considerados en la muestra	28	28
Total de estudiantes originalmente matriculados en la materia.	29	30
Método de seguimiento de proyectos de fin de curso	Tradicional	Propuesto

Realizado por: Javier Salazar, 2015

Los datos experimentales se corresponden con experimentos planificados en los que se asignan dos tratamientos distintos a una población de estudiantes. En este tipo de diseños es necesario que todas las características que no intervienen en el diseño y puedan modificar la respuesta, estén controlados y sean similares en los dos grupos a comparar.

Los grupos están conformados por todos los estudiantes matriculados en los periodos lectivos marzo 2014 – agosto 2014 y marzo 2015 – agosto 2015.

Siendo que las muestras se han extraído desde la misma población las varianzas de las muestras son desconocidas pero iguales: $\sigma_1 = \sigma_2 = \sigma$

4.3.5. Especificación del estadístico

Para una prueba t de Student, los grados de libertad serían de $n_1 + n_2 - 2$, que para el conjunto de datos observados sería igual a 56, valor que se afirma como muestra grande (mayor a 30). Se descarta entonces la utilización de la prueba t Student.

Bajo el supuesto que la cuasi-varianza muestral es para este experimento un buen estimador de la varianza poblacional, se sugiere entonces el siguiente estadígrafo basado en la distribución normal:

$$Z = \frac{(\bar{x}_2 - \bar{x}_1)}{\sqrt{\frac{\hat{\delta}_2^2}{n_2} + \frac{\hat{\delta}_1^2}{n_1}}}$$

Donde:

\bar{x}_1 : Media del Grupo 1.

\bar{x}_2 : Media del Grupo 2.

$\hat{\delta}_1^2$: Varianza del Grupo 1.

$\hat{\delta}_2^2$: Varianza del Grupo 2.

n_1 : Número de estudiantes del Grupo 1.

n_2 : Número de estudiantes del Grupo 2.

4.4. Comprobación

A continuación se presenta un resumen de los grados de calidad del proyecto de fin de curso de cada uno de los estudiantes de los grupos muestrales:

Tabla 2-4: Grados de calidad de proyectos de los estudiantes del periodo 2014

0%	0%	50%	50%	50%	50%	20%
20%	20%	80%	70%	80%	20%	30%
20%	50%	30%	30%	50%	30%	50%
30%	70%	70%	50%	60%	40%	40%

Realizado por: Javier Salazar, 2015

Tabla 3-4: Grados de calidad de proyectos de los estudiantes del periodo 2014

90%	90%	80%	25%	35%	35%	20%
40%	50%	25%	25%	25%	35%	5%
5%	50%	40%	20%	10%	0%	10%
30%	30%	30%	30%	15%	25%	55%

Realizado por: Javier Salazar, 2015

Para este conjunto de datos se presenta la Tabla 4-4 que reúne los datos calculados en base a la estadística descriptiva:

Tabla 4-4: Valores de la estadística descriptiva para los grupos muestrales

	Periodo 2014	Periodo 2015	Total
Media	41%	33%	37%
Desviación estándar	21.72%	23.26%	23%
Error estándar	4.11%	4.40%	3%
N	28	28	56
Mínimo	0%	0%	0%
Máximo	80%	90%	90%

Realizado por: Javier Salazar, 2015

A continuación se presenta el Gráfico 16-4 de caja y bigote basada en los datos de la estadística descriptiva:

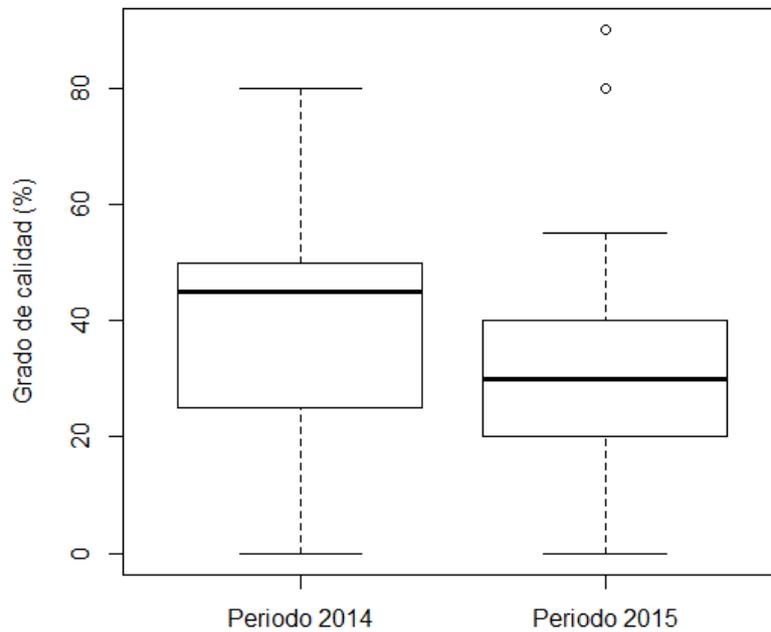


Gráfico 16-4: Gráfico de caja y vigote para los dos grupos muestrales

Realizado por: Javier Salazar, 2015

La simple inspección visual nos muestra que a diferencia de lo esperado, la calidad de los proyectos no ha aumentado con la aplicación del nuevo método sugerido.

Se aplica a continuación la estadística inferencial para demostrar formalmente este hecho para toda la población en estudio.

Tabla 5-4: Contraste para la diferencia de medias de los dos grupos muestrales

Hipótesis	$H_0: \mu_2 \leq \mu_1$ $H_a: \mu_2 > \mu_1$	
Nivel de significación	α	$\alpha = 0.05$ (5%)
Estadígrafo de contraste	$Z = \frac{(\bar{x}_2 - \bar{x}_1)}{\sqrt{\frac{\hat{\delta}_2^2}{n_2} + \frac{\hat{\delta}_1^2}{n_1}}}$	$Z = \frac{(41 - 33)}{\sqrt{\frac{471.95}{28} + \frac{541.13}{28}}} = 1.33$
Distribución del estadígrafo cuando la hipótesis nula es cierta	$N(0, 1)$	$N(0, 1)$
Valores críticos al 5%	Z_α	$Z_{0.05} = 1.64$
Región de aceptación (de la hipótesis nula)	$\{Z/Z \leq z_\alpha\}$	$\{Z/Z \leq 1.64\}$
Región crítica	$\{Z/Z > z_\alpha\}$	$\{Z/Z > 1.64\}$

Realizado por: Javier Salazar, 2015

4.5. Conclusión de la hipótesis

A continuación se presenta el Gráfico 17-4 que resume los datos calculados en base a la estadística inferencial y que permite llegar a la conclusión sobre la hipótesis presentada para el presente trabajo de investigación:

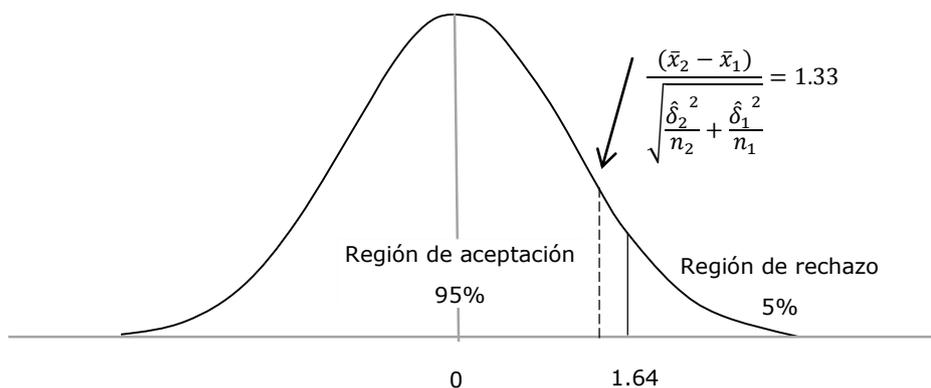


Gráfico 17-4: Representación gráfica de valores calculados para el estadígrafo de contraste.

Realizado por: Javier Salazar, 2015

Por los resultados calculados y presentados en esta sección, se formaliza a continuación las conclusiones:

Decisión estadística

El valor del estadígrafo de contraste pertenece a la región de aceptación, por lo tanto se acepta la hipótesis nula.

Conclusión no estadística

La evidencia encontrada es insuficiente para afirmar con alto grado de certeza que la utilización de un método que incorpore a GitHub como herramienta de soporte tecnológico en el seguimiento de proyectos de fin de curso de materias de Lenguajes de Programación en carreras de Ingeniería Electrónica mejora la calidad de los proyectos, sin embargo, el estudio ha permitido determinar que su utilización evidentemente mejora el proceso de seguimiento de los proyectos.

La siguiente lista evidencia algunas de las ventajas del método MESEPP sobre el método tradicional:

Potencia el proceso de educación:

- Permite un seguimiento continuo durante todo el tiempo de ejecución del proyecto.
- Realza una enseñanza y una evaluación de tipo formativa.
- Potencia una enseñanza personalizada.
- Evidencia deficiencias en el aprendizaje y/o en la aplicación de conceptos presentados en el aula.

Valora en más el trabajo colaborativo y el esfuerzo distribuido:

- El 70% de la ponderación total del proyecto está basado en el trabajo grupal. El 30% en el trabajo individual.
- El 80% de la ponderación total del proyecto está basado en el esfuerzo continuo y progresivo durante todo el tiempo de ejecución del proyecto. El 20% en la presentación final de proyecto.

Permite una evaluación objetiva:

- Evidencia a los estudiantes que no han colaborado en el proyecto.
- Visibiliza a los estudiantes que han realizado la mayor cantidad de actividades.
- Evidencia la distribución de la complejidad del proyecto entre los miembros del equipo.
- Evidencia la distribución del esfuerzo durante los días dispuestos para la ejecución del proyecto.

Visibiliza los hábitos de estudio y trabajo de los estudiantes:

- Visualiza los días, de entre todos los dispuestos para la ejecución del proyecto, en los que los estudiantes concentran su esfuerzo sobre el proyecto.
- Visualiza las horas del día en las que los estudiantes realizan un mayor aporte sobre el proyecto.
- Visualiza los días de la semana en los que los estudiantes realizan la mayor cantidad de trabajo.

Facilita el trabajo colaborativo en proyectos de software:

- Permite el almacenamiento progresivo y periódico de los avances del proyecto en un repositorio centralizado de acceso universal para los estudiantes y el docente.
- Facilita la regresión o referencia a versiones anteriores del proyecto.
- Permite realizar un trabajo paralelo por parte de los estudiantes sobre un mismo proyecto e inclusive sobre un mismo archivo del proyecto.

Transparenta la ejecución de las actividades:

- Evidencia a detalle las líneas de código añadidas, eliminadas o modificadas por cada estudiante.
- Visibiliza el proyecto a los miembros del mismo equipo, a los miembros de otros equipos y al docente.

Potencia el seguimiento y la guía a los estudiantes:

- Facilita una retroalimentación periódica y progresiva.
- Permite el encaminamiento temprano de los esfuerzos de los estudiantes.
- Evidencia los requerimientos bien o mal entendidos por los estudiantes.

Automatiza las actividades:

- Automatiza el proceso de colaboración sobre el proyecto software.
- Automatiza el versionamiento de cada uno de los archivos y recursos del proyecto.
- Automatiza el registro de actividad de cada uno de los estudiantes y de cada uno de los grupos.

CAPÍTULO V

5. PROPUESTA

5.1. Introducción

En esta sección se presenta la propuesta de un nuevo método de seguimiento y evaluación de proyectos de fin de curso para proyectos de lenguajes de programación en carreras de Ingeniería Electrónica. Al nuevo método se lo ha identificado con las siglas MESEPP provenientes del nombre “Método de seguimiento y evaluación de proyectos de programación”.

MESEPP nace de la motivación por potenciar los beneficios que puedan obtener los estudiantes con la ejecución de los proyectos de fin de curso.

5.2. Objetivos

Con la utilización del nuevo método se pretende:

- Realizar un seguimiento continuo a los estudiantes durante el periodo de ejecución de los proyectos, promoviendo una repartición equitativa del trabajo entre los miembros de los equipos y una repartición equitativa de su esfuerzo durante el tiempo de ejecución del proyecto.
- Obtener mayor información y más relevante de la ejecución de los proyectos de fin de curso, que permita al docente realizar una evaluación más certera y objetiva de sus estudiantes.
- Utilizar una herramienta tecnológica que facilite la colaboración de los estudiantes en sus equipos de trabajo y que permita, a la vez, transparentar sus actividades grupales e individuales.
- Utilizar herramientas tecnológicas que brinden las facilidades para el seguimiento de los proyectos pero con un bajo costo de inversión para docentes y estudiantes.

5.3. Descripción de escenario

El proyecto de fin de curso de la materia de Lenguajes de Programación en las carreras Ingeniería Electrónica se realiza como una asignación adicional que el docente envía a sus estudiantes en las últimas semanas o días del periodo lectivo vigente.

El proyecto ayuda a los estudiantes a poner en práctica en una aplicación software, los conocimientos adquiridos durante el curso. El proyecto es ejecutado en equipos de hasta tres (3) estudiantes con algunas excepciones cuando existen deserciones de estudiantes durante el semestre.

Un mismo proyecto es enviado a todos los equipos, sus requerimientos específicos son definidos y delineados por el docente. Al final del proyecto, en el día acordado, los estudiantes realizan una presentación de su proyecto. El docente califica de manera individual y grupal la participación, ejecución y presentación del proyecto.

5.4. Definición del método

MESEPP define los participantes, actividades y artefactos a ser utilizados por parte de los docentes de Lenguajes de Programación para guiar sus tareas de seguimiento y evaluación de los proyectos de fin de curso o semestre.

El grupo objetivo que ha sido considerado para la definición y desarrollo del método es el compuesto por estudiantes de la materia de Lenguajes de Programación I en las Carreras de Ingeniería Electrónica de la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo.

Sin embargo nótese que la aplicación del método puede ser extendido a otros segmentos en donde el escenario tenga características similares al presentado en esta investigación.

5.5. Lineamientos

Para el desarrollo del método MESEPP se han planteado lineamientos que deban dirigir y enmarcar su desarrollo. Los lineamientos permitirán alcanzar los objetivos planteados para el uso del nuevo método.

Los siguientes son los lineamientos definidos para el nuevo método:

1. Que incluya un componente de retroalimentación a los estudiantes.
2. Que incluya un procedimiento de calificación de los proyectos.
3. Que esté soportado por una herramienta informática que facilite las tareas de revisión del código fuente de los proyectos que tenga un bajo costo de inversión para docentes y estudiantes.

5.6. Estructura del método

Basado en el escenario, objetivos y lineamientos definidos para el método MESEPP, a continuación se presenta su desarrollo.

5.6.1. Roles

- a. Docente:* Profesor de la materia de Lenguajes de Programación en una carrera de Ingeniería Electrónica, que realiza la planificación, seguimiento y evaluación de los proyectos de fin de curso de su asignatura.
- b. Estudiante:* Estudiante de la materia de Lenguajes de Programación de una carrera de Ingeniería Electrónica, que ejecuta el proyecto de fin de curso de la asignatura.

5.6.2. Artefactos

5.6.2.1. ARTEFACTO 01: Definición de proyecto

Título del proyecto: Incluir un título del proyecto que de primera vista indique claramente el ámbito del proyecto.

Objetivo de la aplicación: Definir cuál es el objetivo general, que, la suma de todos los requerimientos de la aplicación informática permitirá alcanzar.

Objetivos en la materia: Definir los objetivos de la materia que se pretende alcanzar con la ejecución de los requerimientos específicos de la aplicación informática propuesta. Esto es importante por las siguientes consideraciones:

- Se da un sentido de consistencia al estudiante, al afirmar que no más que los temas tratados en clase son los que le permitirán abordar y completar el proyecto.
- Entregan una pista al estudiante de los temas generales que se espera que aplique en el proyecto. Si los requerimientos no son suficientes para el estudiante para determinar los tópicos de clase que debe aplicar en el proyecto, estos objetivos le marcarán el camino correcto a seguir.

Descripción General: Describir de manera general el sistema.

Requerimientos Específicos: Detallar uno a uno los requerimientos que se espera que ejecuten en el programa informático.

Fecha de Inicio: Definir el día que se iniciará el proyecto. Esta fecha es importante pues es la fecha en la que el docente se compromete también a empezar su seguimiento diario sobre el avance del proyecto.

Fecha de Fin: Definir el día en el que se recibirá el último aporte al proyecto utilizando la plataforma GitHub. Esta fecha es importante pues en este día el docente realizará la última revisión del avance del proyecto y hasta allí completará el 80% de la calificación de cada uno de los estudiantes.

Presentación final: Definir la fecha, hora, lugar y condiciones en los que se va a realizar la presentación final del proyecto de fin de curso. En esta presentación cada estudiante defenderá el 20% restante de la calificación total de su proyecto. Como parte de las condiciones y en dependencia de la complejidad del proyecto definir el tiempo dispuesto para su presentación y defensa. El tiempo sugerido es 15 a 25 minutos.

Método: Indicar de manera detallada el método que se ejecutará para el seguimiento del proyecto.

5.6.2.2.ARTEFACTO 02: Manual de GitHub

El manual de GitHub está narrado y detallado en base a las necesidades previstas para la ejecución del proyecto de fin de curso de la materia de Lenguajes de Programación I. El documento completo se lo presenta en el Anexo A.

5.6.2.3.ARTEFACTO 03: Definición de equipo

Equipo No: [No]

Líder: [Apellidos] [Nombres]

Identificación /Código	Apellidos	Nombres	Nombre Preferido	Email	Usuario GitHub

5.6.2.4.ARTEFACTO 04: Retroalimentación

Equipo No: [No]

[APELLIDOS 1] [NOMBRES 1]

Día 1	[dd/mm/yyyy]
[Retroalimentación]	
Día 2	[dd/mm/yyyy]
[Retroalimentación]	
Día N*	[dd/mm/yyyy]
[Retroalimentación]	

[APELLIDOS N] [NOMBRES N]

Día 1	[dd/mm/yyyy]
[Retroalimentación]	
Día 2	[dd/mm/yyyy]
[Retroalimentación]	
Día N*	[dd/mm/yyyy]
[Retroalimentación]	

* Fecha que en el *Artefacto 01: Definición de proyecto* haya sido definida como la *Fecha de Fin*.

5.6.2.5.ARTEFACTO 05: Registro de avance

Equipo No: [No]

AVANCE SOBRE EL TOTAL DEL PROYECTO

Día 1	Día 2	Día 3	Día 4	Día 5	Día 6	Día 7	Día 8
%	%	%	%	%	%	%	%

Día 9	Día 10	Día 11	Día 12	Día 13	Día 14	Día 15	Día 16
%	%	%	%	%	%	%	%

ACTIVIDAD INDIVIDUAL

[APELLIDOS 1] [NOMBRES 1]

	Actividad respecto al equipo				Observaciones
	Alta	Media	Baja	Ninguna	
Día 1					
Día 2					
Día N*					

[APELLIDOS N] [NOMBRES N]

	Actividad respecto al equipo				Observaciones
	Alta	Media	Baja	Ninguna	
Día 1					
Día 2					
Día N*					

* Fecha que en el *Artefacto 01: Definición de proyecto* haya sido definida como la *Fecha de Fin*.

5.6.2.6.ARTEFACTO 06: Cuestionario final

Equipo No: [No]

[APELLIDOS 1] [NOMBRES 1]

Antecedentes:

Pregunta:

Ámbito de respuesta:

Respuesta entregada:

[APELLIDOS N] [NOMBRES N]

Antecedentes:

Pregunta:

Ámbito de respuesta:

Respuesta entregada:

5.6.2.7.ARTEFACTO 07: Rúbrica de evaluación

Durante la ejecución del proyecto

Porcentaje total del proyecto: 80%

	Valoración Grupal (60%)		
	0%	10%	20%
El equipo repartió entre sus integrantes equitativamente el esfuerzo del proyecto.	No. Algunos miembros del equipo hicieron más esfuerzo que otros.	En parte. En cantidad de líneas de código la repartición fue casi equitativa, pero algunos ejecutaron los requerimientos más complejos y otros sólo los más sencillos.	Sí. El equipo repartió el proyecto de manera equitativa tanto en cantidad como en complejidad del proyecto.
El equipo repartió equitativamente el tiempo dedicado al proyecto.	No. En los primeros días casi no se observó trabajo, mientras que en los últimos se completó la mayor cantidad del proyecto.	En parte. Se observó participación continua en el proyecto, pero no fue suficiente para completar todos los requerimientos.	Sí. Se observó un trabajo continuo y sostenido desde los primeros días hasta los últimos. El proyecto fue terminado. Es más se terminó el proyecto antes de la finalización de los días previstos para su ejecución.
El equipo ha completado todos los requerimientos especificados para el proyecto.	No. No se han completado todos los requerimientos.	Parcialmente. No se han completado todos los requerimientos, pero los ejecutados funcionan bien.	Sí. Se han completado todos los requerimientos y todos funcionan bien.

	Valoración Individual (20 %)		
	0%	10%	20%
Las partes de código creadas, respetan la teoría y consideraciones presentadas en clase (*).	No. Se descuidaron temas y consideraciones tratadas en clase. El código no es óptimo. Es más tiene errores que ocurren u ocurrirán bajo ciertas condiciones.	Parcialmente. En algunos temas se han tomado en cuenta las consideraciones tratadas en clase, en otras secciones se han descuidado.	Sí totalmente. Se ha tenido cuidado en utilizar los temas y consideraciones tratados en clase. En algunos casos se han utilizado otras estrategias producto de una investigación del estudiante.

En la presentación final del proyecto

Porcentaje total del proyecto: 20%

	Valoración Grupal (10%)		
	0%	5%	10%
La presentación final del proyecto fue satisfactoria.	No. No se observa preparación de la presentación. Fue desordenada y no se dio realce al proyecto.	Parcialmente. Se nota una preparación algo improvisada de la presentación.	Sí. Se nota una preparación adecuada de la presentación. Tiene responsables asignados, tiempos coherentes y una secuencia de presentación que da realce al proyecto.

	Valoración Individual (10%)		
	0%	5%	10%
El estudiante respondió correctamente la pregunta realizada por el profesor.	No. La pregunta no fue respondida, la respuesta no se entendió o se respondió algo diferente a lo preguntado.	Parcialmente. Se respondió la pregunta parcialmente. No se dijeron cosas importantes esperadas en la respuesta.	Sí. Se respondió la pregunta total y coherentemente.

5.6.3. Actividades

- Planificación del proyecto.
- Capacitación de plataforma.
- Definición de equipos de trabajo.
- Seguimiento diario.
- Evaluación de ejecución.
- Presentación de proyecto.
- Evaluación de presentación.

5.6.4. Procedimiento

- El docente de la materia de Lenguajes de Programación realiza la planificación del proyecto de fin de curso. Para ello completa el *ARTEFACTO 01: Definición de proyecto*.
- El docente realiza un taller de capacitación de GitHub, de tres (3) horas de duración. En el taller se realizan las siguientes actividades:

- i. Presentación del ámbito de la herramienta, su objetivo, utilidad, proyectos importantes soportados por la plataforma. El objetivo es motivar al estudiante en el uso de la plataforma.
- ii. Creación de cuentas individuales de los estudiantes en el portal de GitHub.
- iii. Instalación de la herramienta GitHub de escritorio para el sistema operativo de preferencia del estudiante.
- iv. Creación de un proyecto por equipo en la plataforma y vinculación de cada uno de los miembros.
- v. Explicación y práctica del uso de la herramienta GitHub para escritorio.
- vi. Inclusión del docente como miembro de cada uno de los equipos.
- c. El docente entrega del *ARTEFACTO 02: Manual de GitHub* a los estudiantes como complemento de la capacitación y como una fuente de consulta y referencia para su utilización posterior.
- d. El docente envía a un estudiante definido como el líder del equipo el *ARTEFACTO 03: Definición de equipo*, el mismo que es llenado y enviado de regreso al docente.
- e. El docente instala la aplicación GitHub para escritorio en el sistema operativo de su preferencia.
- f. Desde el día acordado como la fecha de inicio del proyecto el docente diariamente, en lo posible siempre en horario similar, realiza las siguientes actividades:
 - i. Descarga la última versión de cada uno de los proyectos.
 - ii. Revisa los cambios realizados por el equipo de estudiantes desde el día anterior hasta este nuevo día de revisión.
 - iii. Llena la información del *ARTEFACTO 04: Retroalimentación*, en la sección de trabajo en equipo.
 - iv. Revisa los cambios realizados de manera individual por cada miembro del equipo.
 - v. Llena la información del *ARTEFACTO 04: Retroalimentación*, en la sección de trabajo individual.
 - vi. Una vez completada la revisión envía las retroalimentaciones completadas a cada uno de los estudiantes, al correo electrónico que cada uno haya dispuesto en el *ARTEFACTO 03: Definición de equipo*. Es deseable realizar el envío por correo electrónico por ser un medio de comunicación personal. GitHub en su esquema gratuito expone de manera pública toda la información, mecanismo inconveniente para el envío de retroalimentaciones. Su exposición pública podría causar perjuicios

por parte de otros miembros del mismo equipo, de otros equipos o de personas externas al proceso de enseñanza.

- vii. El docente completa el *ARTEFACTO 05: Registro de avance* en las secciones individual y grupal para este día de revisión.
- viii. En el caso que el día de revisión exista un aporte grupal o individual de gran impacto o de interés técnico/pedagógico, el docente incluye esta información en un formato de pregunta en el *ARTEFACTO 06: Cuestionario final*.
 - g. El día definido como fecha de fin de ejecución del proyecto, el docente realiza la última revisión de los proyectos y envía las últimas retroalimentaciones.
 - h. En las fechas comprendidas entre la fecha de fin de ejecución del proyecto y la fecha de presentación final, el docente llena el *ARTEFACTO 07: Rúbrica de evaluación* en su sección “Durante la ejecución del proyecto”. El docente utiliza para ello la información recolectada en el *ARTEFACTO 05: Registro de avance*.
 - i. El docente valida que el *ARTEFACTO 06: Cuestionario final* se encuentre completo para cada uno de los equipos y estudiantes, caso contrario lo completa en base al trabajo de seguimiento realizado sobre cada uno de los proyectos.
 - j. Se realiza la presentación final de los proyectos en la fecha definida para dicho evento. Se entrega a los estudiantes el tiempo definido para la presentación y defensa del proyecto.
 - k. Luego de la presentación, el docente realiza al menos una pregunta individual a cada uno de los miembros de los equipos. La pregunta es tomada desde el *ARTEFACTO 06: Cuestionario final*. De ser necesario el docente realiza las preguntas adicionales que crea necesarias.
 - l. El docente llena el *ARTEFACTO 07: Rúbrica de evaluación* en la sección “Presentación de proyecto”. Utiliza para ello la técnica de observación durante la presentación y la ronda de contestación de preguntas.
 - m. Una vez realizadas las presentaciones finales, el docente totaliza los valores del *ARTEFACTO 07: Rúbrica de evaluación*. El valor resultante es el grado de calidad en la ejecución del proyecto de cada equipo en su componente individual y grupal. Este valor que representa un porcentaje de 0% a 100% puede ser correspondido con la calificación final del proyecto. Así, si un estudiante ha obtenido un valor del 70% para un proyecto cuya puntuación haya sido planificada para 10 puntos el puntaje correspondiente del estudiante será de 7.

5.7. Ventajas y Desventajas

La aplicación del método propuesto presenta algunas ventajas y desventajas. La Tabla 1-5 presenta algunas de ellas:

Tabla 1-5: Ventajas y desventajas del método MESEPP

Ventajas	Desventajas
<p>El estudiante puede recibir una retroalimentación personalizada respecto a su avance. Esto tiene un componente motivacional y pedagógico intrínseco. Por un lado el estudiante puede sentir una atención directa sobre el trabajo y por el esfuerzo que está realizando sobre el proyecto. Por otro lado la enseñanza se enmarca en un esquema personalizado de gran valor didáctico para la retención y afianzamiento de conocimientos.</p>	<p>El método requiere bastante esfuerzo y tiempo por parte del docente. A pesar de que el método es relativamente sencillo, la mayor cantidad de esfuerzo recae en las revisiones diarias que el docente deba ejecutar durante el tiempo de ejecución de los proyectos. Las revisiones deben ser detalladas y minuciosas para entregar retroalimentaciones no sólo de valor grupal sino individual. Esto requiere bastante del tiempo del docente que debería ser reservado de su ya apretada agenda.</p>
<p>El docente tiene una visión global del avance de todos los equipos. Esto es importante para detectar temáticas o conceptos que el docente deba reforzar con actividades en el aula de clases o en sus prácticas de laboratorio.</p>	<p>Los resultados de la aplicación del método dependen de la percepción del docente. Desde el porcentaje de avance del proyecto, pasando por las retroalimentaciones y extendiéndose hacia la valoración en la rúbrica, todas son actividades que dependen del punto de vista del docente. Esto no es diferente a como se realizan actualmente las evaluaciones de proyectos de programación en la academia, sin embargo visibiliza el hecho de que el método no controla los resultados que son de exclusiva responsabilidad del docente.</p>

Realizado por: Javier Salazar, 2015

Tabla 1-5: Ventajas y desventajas del método MESEPP (Continuación)

Ventajas	Desventajas
<p>La valoración del esfuerzo grupal e individual es objetiva. Gracias a la utilización de la plataforma informática que permite visualizar el aporte individual de cada uno de los miembros de los equipos, el docente tiene la información necesaria para realizar una valoración fundamentada, objetiva y finalmente justa para sus estudiantes.</p>	<p>El uso de la plataforma GitHub puede resultar compleja para los estudiantes. A pesar de que en el esquema básico para modificar un documento y subir los cambios a la plataforma, GitHub es relativamente sencillo, el proceso empieza a complicarse cuando el usuario confunde los cambios locales con los remotos, cuando modifica en paralelo las mismas líneas de código que alguno de los miembros de su equipo, entre otras.</p>
<p>La valoración es progresiva y no acumulativa. El método permite realizar una valoración basada en el avance progresivo de las actividades de los equipos e individuos. El método concentra el 80% de la valoración en el esfuerzo realizado en el proceso y sólo un 20% en la presentación final.</p>	<p>El método no automatiza el procedimiento completo. A pesar de que GitHub como herramienta informática cubre la necesidad de visibilizar los aportes individuales de código por parte de los estudiantes, el resto de tareas del método deben ejecutarse de manera manual. El mantener carpetas ordenadas para cada equipo, abrir cada uno de los artefactos para escribir avances, retroalimentaciones y observaciones, el envío de los emails a los estudiantes, todas estas son tareas que deben ser realizadas de manera manual por el docente y que claramente, como una extensión al método, podrían ser automatizadas.</p>

Realizado por: Javier Salazar, 2015

CONCLUSIONES

- La utilización del método MESEPP mejora el proceso de seguimiento de los proyectos de fin de curso, potenciando el proceso de educación, facilitando el trabajo colaborativo, valorando el esfuerzo distribuido, permitiendo una evaluación objetiva, visibilizando los hábitos de estudio de los estudiantes y, transparentando y automatizando las actividades inherentes a la ejecución de proyectos de software.
- Existen varias experiencias en el sector académico con el uso de GitHub, principalmente en áreas de ingeniería de software. Los resultados obtenidos son prometedores. GitHub mejora la interacción entre docente/estudiante y estudiante/estudiante, potencia las capacidades ofrecidas por los LMSs en tareas como la colaboración para la creación de contenidos, facilita la distribución de asignaciones y ordena el proceso de recolección de tareas. La desventaja común que puede notarse en estas experiencias es la complejidad y curva de aprendizaje que implica la utilización de GitHub para el docente pero principalmente para los estudiantes.
- La plataforma GitHub tiene varias funcionalidades que son de gran utilidad para el seguimiento de los proyectos de fin de curso de materias de lenguajes de programación. Para el método MESEPP propuesto por esta investigación, y como una forma de no involucrar demasiada complejidad a los estudiantes en el experimento, las tareas básicas de COMMIT y SYNC disponibles en la herramienta GitHub para windows han sido utilizadas. Las mismas permiten guardar localmente los cambios que se han realizado en los archivos del proyecto y sincronizar estos cambios de manera remota con el servidor central de versionamiento.
- GitHub provee versionamiento para cualquier tipo de archivo. A pesar de que el presente trabajo de investigación enmarcó la utilización de GitHub en proyectos de desarrollo de software, sus capacidades de versionamiento son de utilidad para cualquier tipo de archivo. Imágenes, diagramas, audios, videos, documentos de ofimática, archivos PDFs todos ellos pueden ser versionados. Las posibilidades de su utilización rebasan el ámbito del software, el ámbito de la ingeniería y evidentemente el ámbito de la educación.
- La aplicación del método MESEPP ha permitido visibilizar los hábitos de estudio y el verdadero trabajo que realizan los estudiantes para la ejecución de los proyectos. Se evidencia un trabajo sin una distribución equitativa entre los miembros del equipo y

cuyo mayor esfuerzo se concentra en los días próximos a la finalización del tiempo fijado para su ejecución. Esta información progresiva y objetiva recolectada con el uso de MESEPP permite al docente realizar una valoración más cercana a lo real y disminuir la sobre estimación que involucra la utilización del método tradicional. Debido a esta objetividad, las valoraciones de los proyectos con el nuevo método han disminuido comparado con las valoraciones obtenidas por la subjetividad del método tradicional.

- El método MESEPP está integrado por siete (7) artefactos, que constituyen plantillas documentales que el docente puede utilizar para llevar a cabo el procedimiento sugerido por el método. Para el seguimiento continuo MESSEP utiliza a GitHub como plataforma tecnológica. Algunos de los artefactos complementan la información no reportada por GitHub, tal como el porcentaje de avance del proyecto respecto al 100% esperado por el docente.
- A pesar de que el método MESEPP utilizó las características básicas y suficientes de GitHub para soportar la colaboración sobre el proyecto, se pudo evidenciar que su uso involucra una gran complejidad para los estudiantes. En aula y en horas de consulta, el docente ha respondido a los estudiantes tantas preguntas acerca del funcionamiento de GitHub como preguntas propias del dominio del proyecto. En base a la experiencia de este estudio se comparte entonces el criterio de otros investigadores que afirman que el uso de GitHub involucra una alta curva de aprendizaje para los estudiantes.
- El método propuesto por este estudio involucra un tiempo significativo por parte del docente para completar las revisiones diarias. El docente ha tomado de una (1) a tres (3) horas por día en dependencia del grado de participación de los estudiantes.
- Desde el punto de vista del docente la plataforma GitHub es de gran utilidad para la revisión del código fuente de los estudiantes. En su esquema más sencillo la plataforma ordena cronológicamente los aportes de los estudiantes, visibiliza que líneas de código han sido añadidas, modificadas, eliminadas, quién lo hizo y cuando lo hizo. Estos datos se los puede visualizar en una herramienta gráfica sencilla que a través de simbologías de colores permite discriminar la información.
- La rúbrica de evaluación incluida como parte del método da lugar a que, en equipos en los que un estudiante no haya tenido participación, aún acumule un puntaje, que es el acumulado por su equipo. Por otro lado, en equipos en los que un estudiante ha sido el único que ha realizado un esfuerzo sobre el proyecto, su calificación se verá afectada

por su ponderación como equipo. Partiendo de la premisa de que el proyecto es un trabajo grupal y que es de esfuerzo y recompensa compartida, es justo que el sobre esfuerzo de un estudiante o la ausencia de trabajo de otro penalicen el sentido de equipo. Así la ponderación ofrecida por la rúbrica es adecuada.

- El componente de retroalimentación incluido en el método ha tenido una respuesta positiva en algunos estudiantes, ellos ha considerado las observaciones para mejorar su proyecto o para corregir su camino cuando la alternativa escogida no era la más adecuada.
- La evidencia encontrada es insuficiente para afirmar con alto grado de certeza que la utilización de un método que incorpore a GitHub como herramienta de soporte tecnológico en el seguimiento de proyectos de fin de curso de materias de Lenguajes de Programación en carreras de Ingeniería Electrónica mejora la calidad de los proyectos, sin embargo, el estudio ha permitido determinar que su utilización evidentemente mejora el proceso de seguimiento de los proyectos.

RECOMENDACIONES

- Aprovechar la información que entrega el método propuesto. Para los docentes que estén interesados en utilizar el método propuesto, la información que se genera durante el proceso no es sólo de valor para la evaluación del proyecto, puede representar una guía para definir los conceptos que deban ser tratados en el aula de clases, las temáticas que deban ser cubiertas con las prácticas de laboratorio, los contenidos que deban ser reforzados y los futuros contenidos que deban ser planificados.
- Automatizar el método propuesto. Para extensiones del presente estudio se sugiere crear una aplicación que mantenga toda la información recogida de los proyectos en una base de datos. Envíos automáticos de mensajes de retroalimentación, ordenamiento cronológico de actividades, marcación de porcentajes de avance, agrupación de información por equipos e individuos, totalización automática de resultados son algunas de las características que pueden ser automatizadas.
- Realizar un nuevo estudio del método propuesto mitigando la complejidad de la plataforma GitHub. Para extensiones del presente estudio se sugiere que el docente involucre a la plataforma GitHub desde el inicio del curso o semestre. Las tareas, laboratorios y prácticas pueden ser trabajados en pares utilizando la plataforma. Su uso temprano permitirá involucrar a los estudiantes con las particularidades de la plataforma y lo que es más importante capacitarlos para tratar con ellas. Nótese que el uso de una herramienta de versionamiento de código en la materia de Lenguajes de Programación no la aparta de sus objetivos, siendo que esta temática es parte del syllabus vigente.
- Aplicar un método como el propuesto en los proyectos de fin de curso. A pesar de que los indicadores del presente estudio demuestran que el método propuesto no consiguió los resultados deseados, el docente u otros estudios pueden encontrar un método efectivo que permita realizar un seguimiento continuo, sostenido y retroalimentado de los proyectos de fin de curso. Encontrar un método que permita explorar al máximo las bondades del proyecto de fin curso, sería un impacto académico de gran valor para los estudiantes.
- Fortalecer las capacidades de lectura comprensiva y consulta en internet. Muchos de los problemas que los estudiantes mantuvieron con la plataforma GitHub, estaban

descritos en la guía de uso. Algunas de las funcionalidades programadas en su aplicación fueron descritas y solicitadas de manera diferente en el documento de definición del proyecto. Algunas secciones de código que podrían resultar complejas para los estudiantes presentan soluciones sencillas en varios sitios de internet. Habilidades de consulta en internet complementadas con lectura comprensiva son capacidades que pueden ayudar a evitar este tipo de omisiones. Se sugiere a los estudiantes potenciar estas capacidades.

- Considerar a este proyecto como una línea base de nuevos proyectos. Por los resultados encontrados en esta investigación se sugiere realizar un nuevo estudio que permita validar que “La utilización de un método que incorpore GitHub como herramienta de soporte tecnológico mejora el proceso de seguimiento de proyectos de fin de curso en materias de Lenguajes de Programación”. Así mismo, se sugiere realizar estudios adicionales y complementarios fuera del ámbito tecnológico, que incluyan la pregunta de investigación: ¿Qué procedimientos o técnicas deberían utilizarse para mejorar el comportamiento de los estudiantes y su motivación de cara a la ejecución de sus proyectos de fin de curso?.

BIBLIOGRAFÍA

1. **Bunge, M.** (2005). *Buscar la filosofía en las ciencias sociales*. México: Siglo XXI Editores.
2. **Chacon S, S. B.** (2014). *Pro Git Second Edition*. Apress.
3. **Coria, A., Pastor, I., & Torres, Z.** (2013). Propuesta de metodología para elaborar una investigación científica en el área de Administración de Negocios. México: Revista científica Pensamiento y Gestión.
4. **Dabbish, L., Stuart, C., Tsay, J., & Herbsleb, J.** (2012). Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository. *ACM 2012 conference on Computer Supported Cooperative Work*. Seattle.
5. **Del Cid, A., Méndez, R., & Sandoval, F.** (2007). *Investigación Fundamentos y Metodología*. México: Pearson.
6. **Git.** (2015). *Git - Fast version control*. Recuperado el 2 de agosto de 2015, de Git: <https://git-scm.com/>
7. **GitHub.** (2015). *Permission levels for a user account repository*. Recuperado el 3 de octubre de 2015, de GitHub Help: <https://help.github.com/articles/permission-levels-for-a-user-account-repository/>
8. **GitHub, Inc.** (2015). *Features*. Recuperado el 13 de octubre de 2015, de GitHub: <https://github.com/features>
9. **GitHub, Inc.** (2015). *Twbs Bootstrap*. Recuperado el 10 de octubre de 2015, de <https://github.com/twbs/bootstrap>
10. **Josh, D.** (2014). *GitHub + University: How College Coding Assignments Should Work*. Recuperado el 13 de octubre de 2015, de joshldavis.com/2014/01/19/github-university-how-college-assignments-should-work/

11. **Lawrance, J., Jung, S., & Wiseman, C.** (2013). *Git on the Cloud in the Classroom*. 44th ACM technical symposium on Computer science education. Denver: ACM.
12. **Ray, B., Posnett, D., Filkov, V., & Devanbu, P.** (2014). *Large Scale Study of Programming Languages and Code Quality in Github*. 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. Hong Kong: ACM.
13. **Real Academia Española.** (2001). *Diccionario de la lengua española*. Recuperado el 25 de Julio de 2015, de Real Academia Española: <http://lema.rae.es/drae/?val=método>
14. **Samaja, J.** (2004). *Epistemología y metodología: elementos para una teoría de la investigación científica*. Buenos Aires: EUDEBA.
15. **Sawers, P.** (11 de Febrero de 2014). *GitHub wants schools to collaborate on code*. (The next web, Inc.) Recuperado el 3 de octubre de 2015, de The Next Web: <http://thenextweb.com/insider/2014/02/11/github-wants-schools-collaborate-code/>
16. **Secretaria Nacional de Planificación y Desarrollo - Ecuador.** (2009). *Plan Nacional Para el Buen Vivir*. Recuperado el 13 de octubre de 2015, de <http://plan.senplades.gob.ec/>
17. **Shaffer, K.** (26 de Mayo de 2013). *Push, Pull, Fork: GitHub for Academics*. Recuperado el 23 de abril de 2015, de HybridPedagogy: <http://www.hybridpedagogy.com/journal/push-pull-fork-github-for-academics/>
18. **Sierra, R.** (2001). *Técnicas de investigación social. Teoría y ejercicios*. Madrid: S.A. EDICIONES PARANINFO.
19. **SitePoint Pty. Ltda.** (2014). *The difference between Git and GitHub*. Recuperado el 2 de agosto de 2015, de sitepoint: <http://www.sitepoint.com/screencast-difference-git-github/>

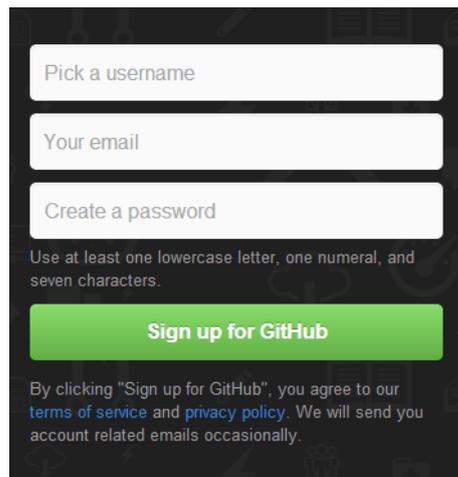
20. **Xu, Z.** (2012). Using Git to Manage Capstone Software Projects. *Seventh International Multi-Conference on Computing in the Global Information Technology*. Venice, Italy: IARIA.
21. **Yu, Y., Yin, G., Wang, H., & Wang, T.** (2014). Exploring the Patterns of Social Behavior in GitHub. *1st International Workshop on Crowd-based Software Development Methods and Technologies*. Hong Kong: ACM Press.
22. **Zagalsky, A., Feliciano, J., Storey, M.-A., Zhao, Y., & Wang, W.** (2015). The Emergence of GitHub as a Collaborative Platform for Education. *18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. Vancouver, BC, Canada.

ANEXOS

Anexo A. ARTEFACTO 02: MANUAL DE GITHUB PARA WINDOWS

COMO EMPIEZO A UTILIZAR GITHUB

1. Ingresa al sitio web <https://github.com>.
2. En la página principal del sitio registra un nuevo usuario:



The screenshot shows the GitHub sign-up form with the following fields and elements:

- Input field: "Pick a username"
- Input field: "Your email"
- Input field: "Create a password"
- Text: "Use at least one lowercase letter, one numeral, and seven characters."
- Button: "Sign up for GitHub" (green)
- Text: "By clicking 'Sign up for GitHub', you agree to our [terms of service](#) and [privacy policy](#). We will send you account related emails occasionally."

3. Selecciona el plan gratuito (free):

Choose your personal plan

Plan	Cost	Private repos	
Large	\$50/month	50	<button>Choose</button>
Medium	\$22/month	20	<button>Choose</button>
Small	\$12/month	10	<button>Choose</button>
Micro	\$7/month	5	<button>Choose</button>
Free	\$0/month	0	<button>Chosen</button>

4. Presiona el botón para finalizar:

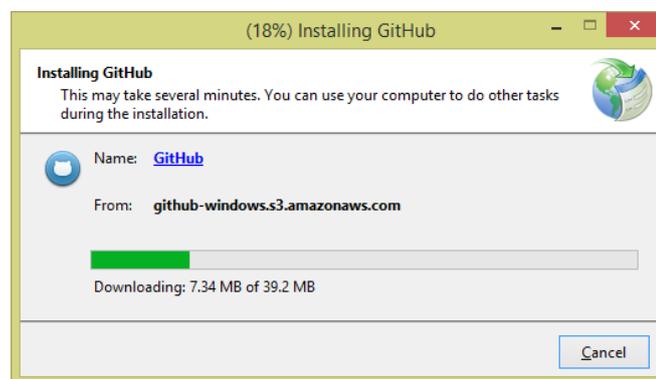


5. En la página principal del sitio busca el enlace de descarga de GitHub para Windows:

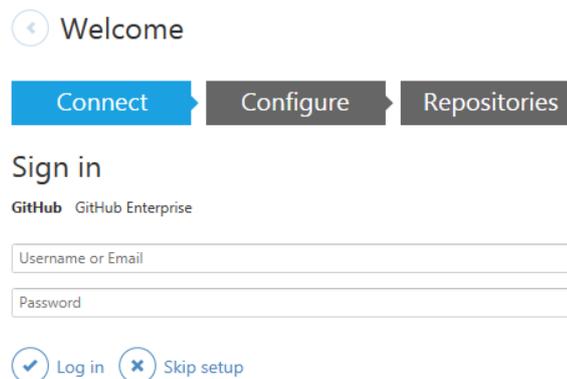
The easiest way to use GitHub on Windows



6. Ejecuta el archivo descargado para realizar la instalación del programa:



7. Ingresa a la nueva aplicación instalada, y en la pantalla principal ingrese tu nuevo nombre de usuario y contraseña:



8. Ingresa tus datos en la pantalla de configuración de tu cuenta:



Configure git

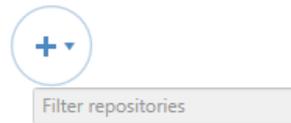
This will be used in the commits you create. Keep in mind that if you publish commits, anyone will have access to this email.

This will change your global gitconfig.

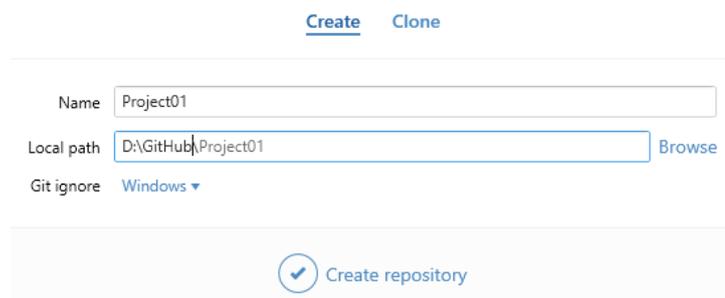
9. En la pantalla de repositorios (repositories) salta el paso.

SOY EL RESPONSABLE DEL EQUIPO, COMO CREO EL PROYECTO LA PRIMERA VEZ

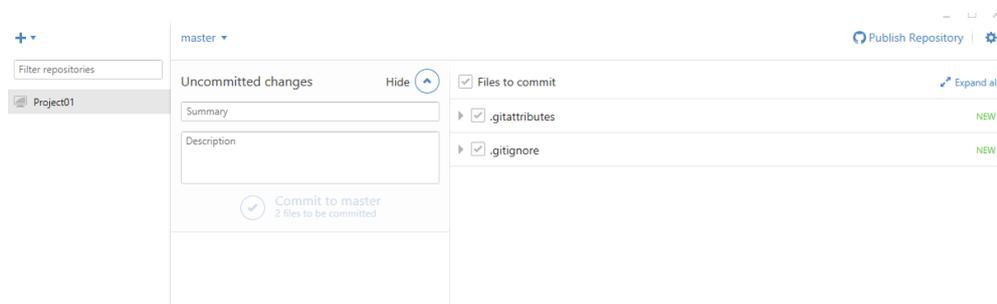
1. En la pantalla principal de GitHub para Windows, presiona en la opción para crear un nuevo repositorio:



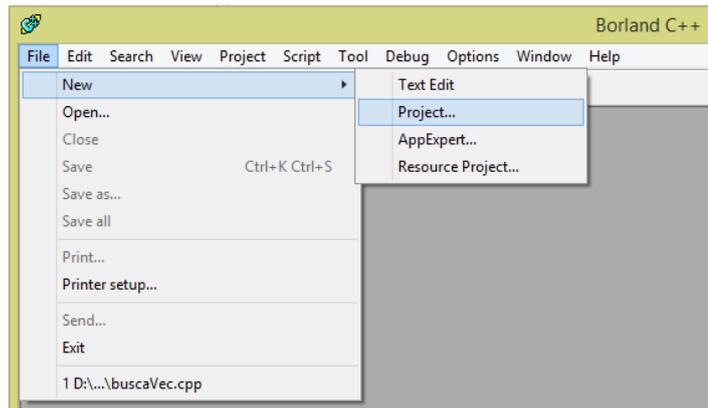
2. Registra el nombre del repositorio y la ubicación en su computador donde será almacenado. Un repositorio es el lugar donde serán colocados los archivos c++ en tu computador. Los archivos de esta ubicación serán sincronizados luego con una ubicación remota en internet en el sitio de GitHub. Recuerda utilizar el nombre del proyecto sugerido por tu profesor moderador (ProyLPEquipo1, donde 1 es el número que se haya asignado a tu grupo de trabajo.).



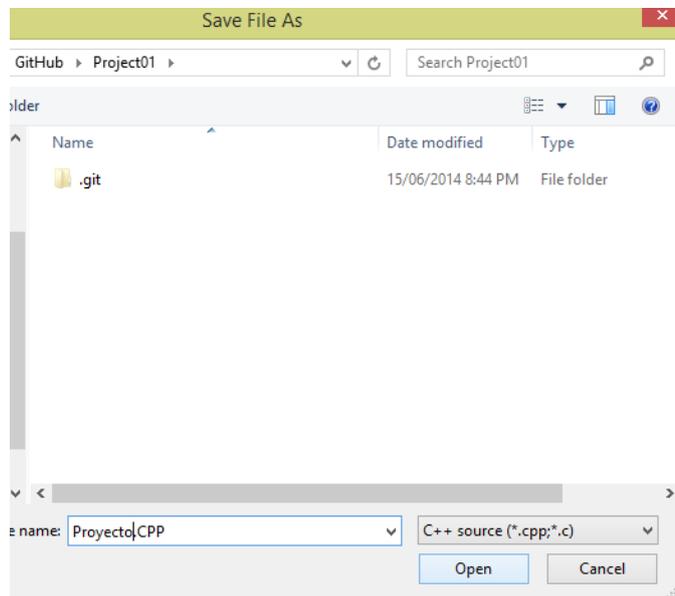
3. Una vez finalizada esta configuración se visualizará la pantalla principal de la aplicación:



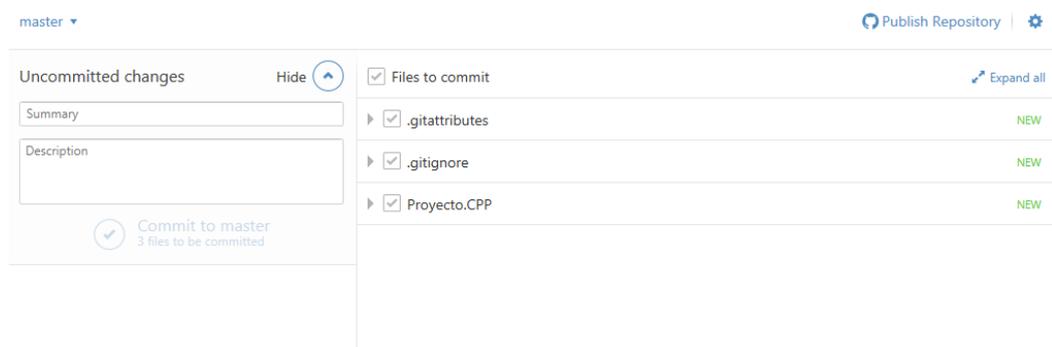
4. Ingresa a la aplicación de Borland C++ y crea el primer o primero archivos de tu proyecto:



5. Guarda este primer o primeros archivos en la ubicación que antes (desde GitHub para windows) hayas registrado como tu repositorio:



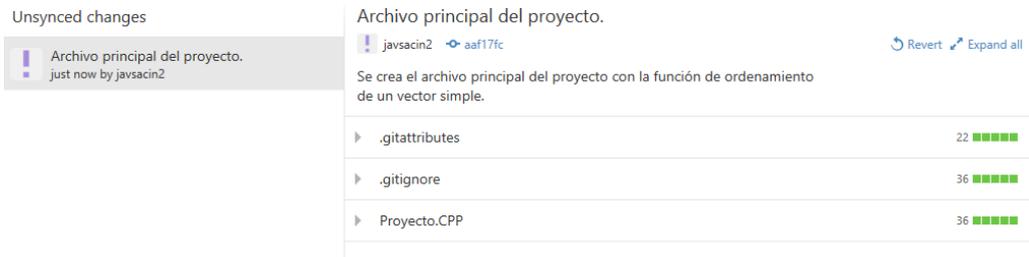
6. Cualquier archivo creado en Borland C++ será automáticamente reconocido por la aplicación GitHub de Windows:



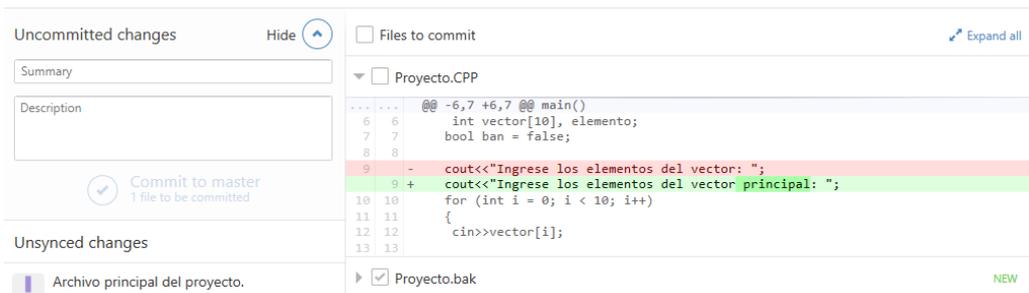
- Una vez realizada la creación o modificación de tus primeros archivos de proyecto, publícalos a tu repositorio local. Para ello digita el resumen de lo que estás publicando y una descripción (en el caso de que sea necesario). Como estos son los primeros y nuevos archivos, visualizarán una etiqueta nuevo (New) en la parte derecha de cada uno de ellos:



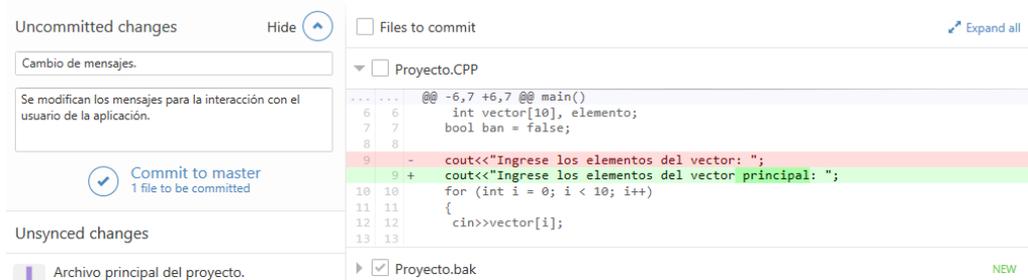
- Presiona el botón de publicación (Commit to master).
- Los archivos serán publicados a tu repositorio local (un lugar todavía en tu computador local). Aún no se han publicado archivos al servidor remoto de GitHub donde tendrán acceso los miembros de tu equipo:



- Realiza todos los cambios que creas necesarios en los archivos desde Borland C++ para dejar una versión hábil para que empiecen a trabajar el resto de los miembros de tu equipo. En la pestaña de Cambios no publicados (Uncommitted changes) de la aplicación GitHub para Windows se verán reflejados los cambios que realices:



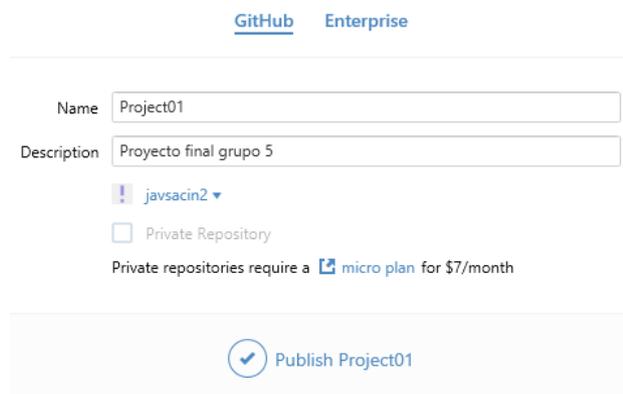
- Para publicar los nuevos cambios a tu repositorio local, digita el resumen y la descripción de los cambios que hayas realizado. En la descripción no es necesario incluir un detalle muy minucioso de tus cambios, una descripción general que indique el objetivo principal será suficiente:



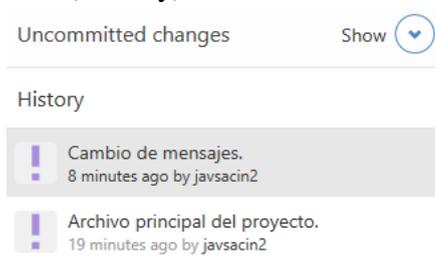
- Presiona el botón para publicar “Commit to master”.
- Todos los cambios que se encuentren registrados en tu repositorio local (tu propio computador) pero que aún no hayan sido enviados al servidor remoto los podrás visualizar en la pestaña cambios no sincronizados (Unsynced changes).



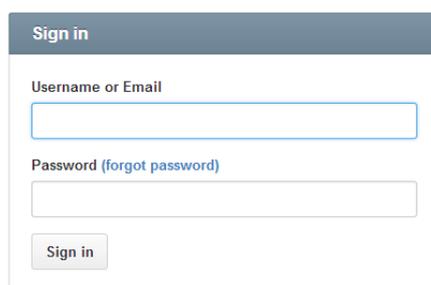
- Para sincronizar todos los cambios de tu repositorio local con el servidor remoto GitHub, presione el botón Publish Repository de la parte superior derecha de la aplicación.
- Coloca el nombre y la descripción del proyecto y presione el botón Publicar (Publish <Nombre Proyecto>). Recuerda colocar como nombre del proyecto el sugerido por tu profesor moderador (ProyLPEquipo1, donde 1 es el número que se haya asignado a tu grupo de trabajo.)



16. Los cambios serán publicados a tu repositorio remoto y ahora serán parte de la pestaña historia de cambios (History):



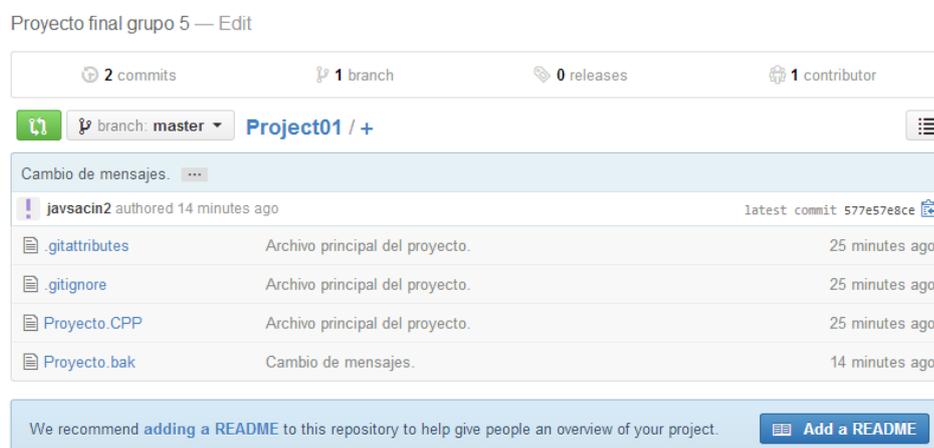
17. Ingrese a la página <https://github.com> con tu nombre de usuario y contraseña:



18. En la sección de tus repositorios (Your Repositories) seleccione el proyecto antes creado:



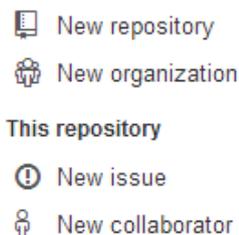
19. Visualizarás una pantalla con los archivos de tu proyecto:



20. En la barra de opciones de la sección superior derecha, selecciona el icono (+):



21. Selecciona la opción Nuevo Colaborador (New collaborator) de las opciones disponibles:



22. Digita el nombre de usuario de la persona a la cual desees añadir como colaborador y presiona el botón Añadir Colaborador (Add collaborator):

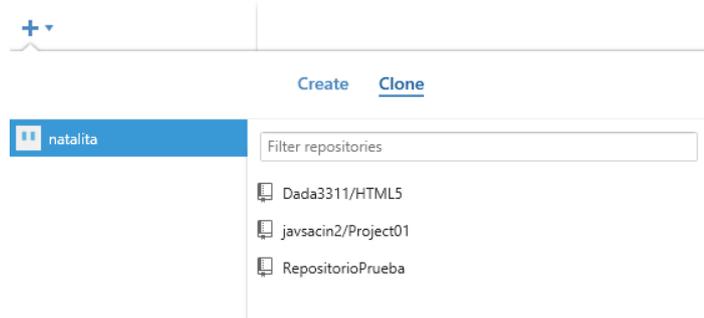
Collaborators		Full access to the repository
<input type="text" value="Type a username"/>	<input type="button" value="Add collaborator"/>	

23. Realiza esta misma operación para el resto de los miembros de tu equipo.

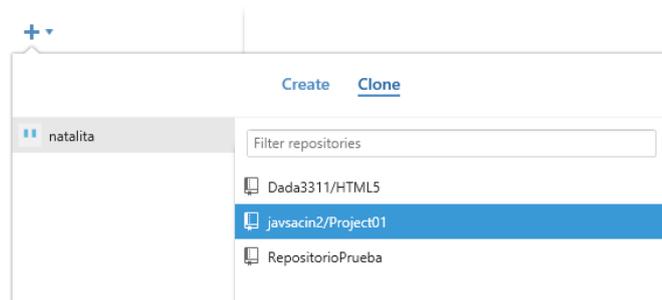
24. No olvides incluir como colaborador a tu profesor moderador “natalita”.

COMO ME CONECTO LA PRIMERA VEZ AL PROYECTO CREADO POR EL RESPONSABLE DE MI EQUIPO

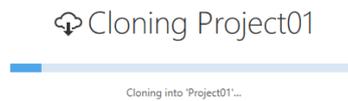
1. Ingresa a la aplicación GitHub para Windows, ingrese tu nombre de usuario y contraseña para conectarte a la aplicación.
2. En la pantalla principal, selecciona el icono (+) en la parte superior izquierda de la pantalla y seleccione la pestaña clonar (Clone).



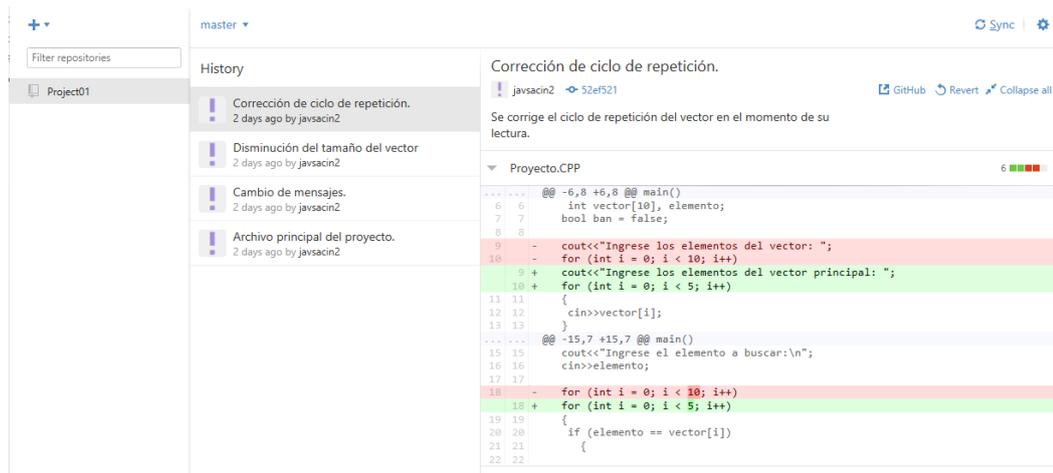
3. Se visualizará el proyecto para el cuál te hayan añadido como colaborador. Selecciónalo.



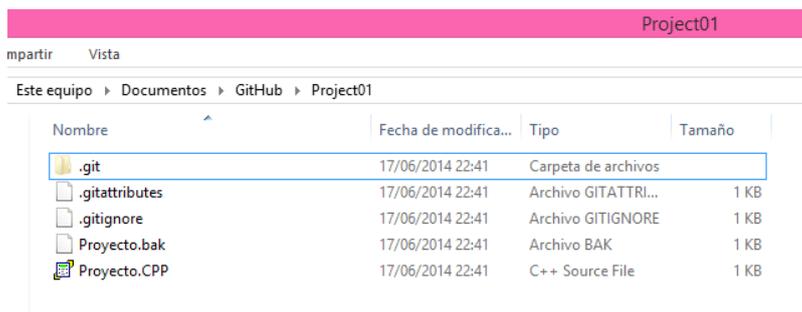
4. Se visualizará una pantalla de progreso de la clonación del proyecto en su computador:



5. Una vez terminada la clonación podrás visualizar el proyecto y la historia de cada uno de los cambios que se hayan ejecutado hasta el momento:



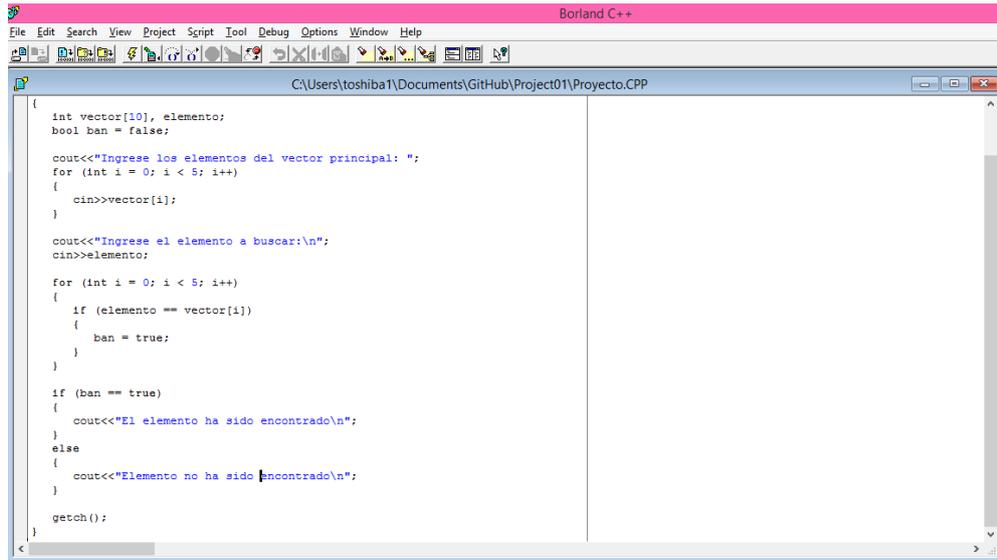
6. Los archivos del proyecto los podrás encontrar en el path que hayas designado para la instalación de GitHub para Windows. Por defecto podrás encontrar los archivos en la carpeta Documentos/GitHub del usuario actual:



7. Una vez localizados los archivos clonados en tu computador podrás abrirlos con Borland C++ y realizar los cambios necesarios para colaborar con el proyecto.

YA ME HE CONECTADO LA PRIMERA VEZ, COMO COLABORO DURANTE EL RESTO DEL PROYECTO

1. Realiza los cambios en el o los archivos en Borland C++ en la ubicación de tu repositorio local. Puedes crear nuevos archivos o modificar archivos existentes.



```
{
int vector[10], elemento;
bool ban = false;

cout<<"Ingrese los elementos del vector principal: ";
for (int i = 0; i < 5; i++)
{
cin>>vector[i];
}

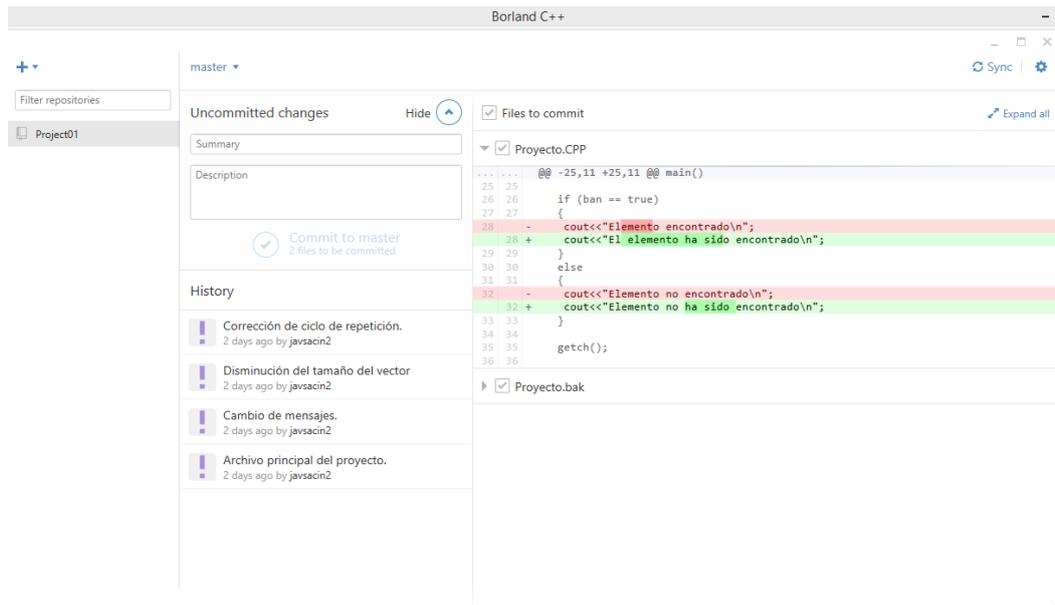
cout<<"Ingrese el elemento a buscar:\n";
cin>>elemento;

for (int i = 0; i < 5; i++)
{
if (elemento == vector[i])
{
ban = true;
}
}

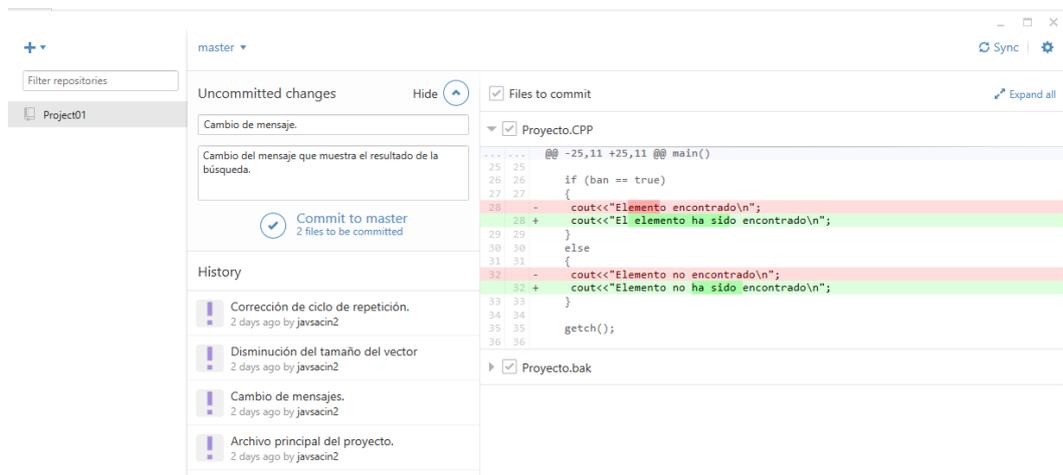
if (ban == true)
{
cout<<"El elemento ha sido encontrado\n";
}
else
{
cout<<"Elemento no ha sido encontrado\n";
}

getch();
}
```

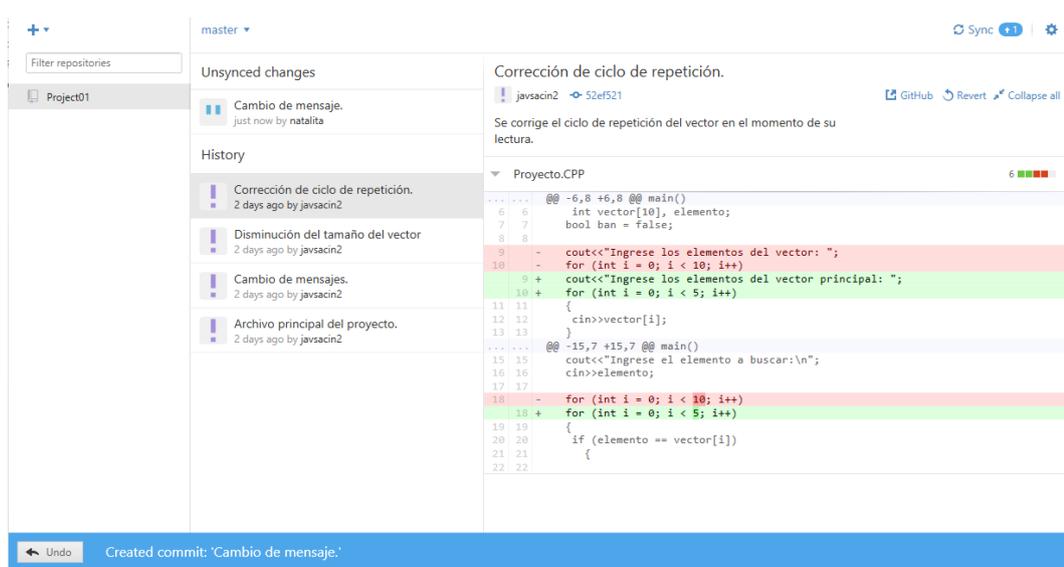
2. Una vez que tengas un estado de cambios que desees publicar, abre o dirígete (si lo tienes abierto) a GitHub para Windows. Podrás observar los cambios realizados listados como cambios no publicados en tu repositorio local (Uncommitted changes)



3. Escribe el resumen y la descripción de los cambios realizados.



4. Presione el botón para publicar en tu repositorio local (Commit to master).
5. Ahora se visualizarán los cambios en la pestaña de cambios no sincronizados con el servidor remoto (Unsynced changes)

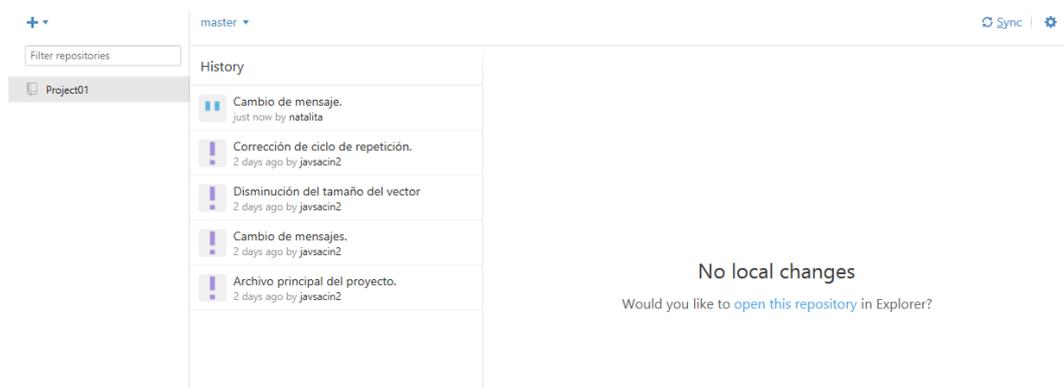


- Hasta allí tendrás publicados los cambios en tu repositorio local. Para sincronizarlo con el servidor remoto presiona el botón de sincronización de la parte superior derecha de la pantalla (Sync).



Nota: En el caso de se produzca algún problema en el momento de la sincronización con el repositorio remoto, verifica que todos los cambios que hayas realizado desde Borland C++ estén publicados en tu repositorio local primero.

- Una vez sincronizados los cambios con el servidor remoto, un mensaje te indicará que no tienes cambios locales (No local changes). Todos los cambios realizados serán visualizados en la pestaña de historial de cambios (History).



8. Repite estos pasos tantas veces como creas necesario por día para colaborar con tu equipo en el proyecto.

Anexo B. PROYECTO FINAL LENGUAJES DE PROGRAMACIÓN I PERIODO

2014

OBJETIVOS

- Afianzar los conocimientos adquiridos en la materia.
- Practicar de manera integral los conceptos aprendidos en clase.
- Potenciar el trabajo en equipo para la ejecución de proyectos.
- Solucionar problemas de la vida real.

DESCRIPCIÓN

Como trabajo final de la materia los estudiantes deberán presentar un proyecto de programación en lenguaje c++. Los detalles del proyecto se presentan a continuación:

Título del Proyecto: Calendario de eventos

Objetivo General: Desarrollar una aplicación que permita el ingreso y visualización de entradas en un calendario de eventos.

Objetivos Específicos:

- Realizar una programación modular.
- Utilizar mecanismos de ordenamiento de arreglos de datos.
- Utilizar archivos como un mecanismo de almacenamiento de información.

Descripción General:

Carlos requiere tener un control de sus actividades diarias. Para ello está buscando una aplicación informática que le permita realizar ese control. Carlos necesita que la aplicación le permita ingresar nuevos eventos en su calendario y que luego le permita visualizar cronológicamente los eventos planificados para el día de hoy y para toda la semana.

Requerimientos Específicos:

- Visualiza una pantalla, tipo menú principal, desde la cual Carlos pueda seleccionar una de las siguientes actividades:

- Ingresar nuevo evento.
- Ver mis eventos para hoy.
- Ver mis eventos para toda la semana.
- Cuando Carlos ingrese un nuevo evento permítele registrar los siguientes datos:
 - Nombre corto del evento.
 - Descripción del evento.
 - Fecha/Hora de Inicio
 - Utiliza un formato de ingreso aaaa/mm/dd hh:mm o utiliza dos campos uno para ingresar la fecha y otro para ingresar la hora.
 - Fecha/Hora de Fin
 - Utiliza un formato de ingreso aaaa/mm/dd hh:mm o utiliza dos campos uno para ingresar la fecha y otro para ingresar la hora.
- Permite que Carlos deje el campo “descripción del evento” en blanco si así lo desea, exige que ingrese el resto de campos.
- En la opción “Ver mis eventos para hoy”:
 - Determina automáticamente cuál es la fecha de hoy basada en la fecha del computador desde la cual Carlos ingresa a la aplicación.
 - Visualiza una lista de eventos ordenados por la Fecha/Hora de inicio de los eventos, esto es, los eventos que Carlos deba ejecutar primero lístalos primero.
 - Para cada evento presenta la información como el siguiente evento de ejemplo:

Desde 2014/06/23 20H00 Hasta 2014/06/23 23H00

Cumpleaños Majito

Llamar a Robert para confirmar el lugar.

- Si Carlos no escribió una descripción no la presentes, quedando el evento de ejemplo así:

Desde 2014/06/23 20H00 Hasta 2014/06/23 23H00

Cumpleaños Majito

- Una vez que Carlos visualice todos los eventos del día de hoy presenta nuevamente el menú principal.
- En la opción “Ver mis eventos para toda la semana”
 - Determina automáticamente la semana actual basada en la fecha de hoy. Considera el Lunes como el primer día de la semana y el domingo como el último día de la semana. Así, si hoy es miércoles 2 de Julio del 2014, el primer día de la semana será el lunes 30 de Junio del 2014.
 - Visualiza una lista de eventos ordenados por la Fecha/Hora de inicio de los eventos, esto es, los eventos que Carlos deba ejecutar primero lístalos primero.

- Para cada evento presenta la información como el siguiente evento de ejemplo:

Desde 2014/06/23 20H00 Hasta 2014/06/23 23H00

Cumpleaños Majito

Llamar a Robert para confirmar el lugar.

- Si Carlos no escribió una descripción no la presentes, quedando el evento de ejemplo así:

Desde 2014/06/23 20H00 Hasta 2014/06/23 23H00

Cumpleaños Majito

- Una vez que Carlos visualice todos los eventos de la semana presenta nuevamente el menú principal.

Anexo C. PROYECTO FINAL LENGUAJES DE PROGRAMACIÓN I PERIODO 2015

OBJETIVOS DEL TRABAJO FINAL

- Afianzar los conocimientos adquiridos en la materia.
- Practicar de manera integral los conceptos aprendidos en clase.
- Potenciar el trabajo en equipo para la ejecución de proyectos.
- Solucionar problemas de la vida real.

DESCRIPCIÓN DEL TRABAJO FINAL

Como trabajo final de la materia los estudiantes deberán presentar un proyecto de programación en lenguaje C++. Los detalles del proyecto se presentan a continuación:

Título del Proyecto: Horario de clases

Objetivo de la aplicación: Desarrollar una aplicación que permita el ingreso y visualización de un horario de clases.

Objetivos en la materia:

- Realizar una programación modular.
- Utilizar mecanismos de ordenamiento de arreglos de datos.
- Utilizar archivos como un mecanismo de almacenamiento de información.

Descripción General:

Pedro es docente de una universidad de la ciudad. El dicta clases de lunes a viernes en diferentes materias y semestres en una de las escuelas. Pedro necesita un programa que le permita ingresar su horario de clases y visualizarlo cada vez que él lo necesite.

Requerimientos Específicos:

- Visualizar una pantalla, tipo “**Menú principal**”, desde la cual Pedro pueda seleccionar una de las siguientes actividades:
 - Ingresar una nueva hora de clases.
 - Ver el horario de hoy.

- Ver el horario de la semana.
- Salir.
- Cuando Pedro seleccione la opción “**Ingresar una nueva hora de clases**” del menú, presenta una pantalla para el ingreso de la siguiente información:
 - Nombre de la materia: Ej. Matemáticas II
 - Nombre del semestre: Ej. Segundo semestre
 - Día de la semana: Ej. LUN
 - Hora de inicio: Ej. 07H00
 - Hora de fin: Ej. 09H00
- Todos los campos citados arriba son requeridos. No permitas que deje uno de estos campos en blanco por error.
- En el campo “Día de la semana”, permite ingresar los siguientes valores: LUN, MAR, MIE, JUE, VIE.
- Guarda esta información en el computador de Pedro. Cualquier otro día que Pedro ingrese nuevamente a la aplicación su información estará disponible.
- Cuando Pedro seleccione la opción “**Ver el horario de hoy**” del menú, realiza las siguientes actividades:
 - Determina automáticamente cual es el día de hoy (Ej. Lunes) basado en la fecha del computador desde el cual Pedro ingresa a la aplicación.
 - Visualiza una lista de las horas de clase que el día de hoy debe dictar Pedro. Ordena las horas desde las más tempranas del día hasta las últimas del día.
 - Utiliza el formato de presentación del siguiente ejemplo. El ejemplo presenta un horario en el que Pedro ingresó dos horas de clase para el día lunes:


```
Lunes: 07H00 - 09H00
Segundo Semestre
Matemáticas II

Lunes: 14H00 - 16H00
Primer Semestre
Matemáticas I
```
 - Cuando Pedro haya terminado de revisar su horario para el día hoy y presione una tecla, presenta nuevamente el menú principal.
- Cuando Pedro seleccione la opción “**Ver el horario de la semana**” del menú, realiza las siguientes actividades:
 - Visualiza todas las horas de clase que Pedro haya ingresado en la aplicación.
 - Ordena las horas por día (de lunes a viernes) y por hora (más tempranas del día a últimas del día).
 - Utiliza el formato de presentación del siguiente ejemplo. El ejemplo presenta 4 horas de clase, 2 para el lunes y 2 para el martes:


```
Lunes: 07H00 - 09H00
Segundo Semestre
Matemáticas II
```

Lunes: 14H00 - 16H00
Primer Semestre
Matemáticas I

Martes: 09H00 - 11H00
Tercer Semestre
Algebra I

Martes: 16H00 - 18H00
Cuarto Semestre
Cálculo II

- Cuando Pedro haya terminado de revisar su horario de toda la semana y presione una tecla, presenta nuevamente el menú principal.
- Cuando Pedro seleccione la opción “**Salir**” del menú, cierra la aplicación.

Fecha de Inicio:

Martes 7 de julio 2015. Este será el primer día que el docente moderador comenzará a revisar tus avances en la plataforma GitHub. Las revisiones se realizarán en la noche. Tienes todo el día para realizar algún aporte sobre tu proyecto.

Fecha de Fin:

Miércoles 22 de julio 2015. Este será el último día que el docente moderador realice una revisión y te entregue una retroalimentación en el avance de tu proyecto. Hasta aquí habrás alcanzado el 80% de la calificación de tu proyecto.

Presentación Final:

Viernes 24 de julio 2015, en los laboratorios de la FIE. Aquí podrás alcanzar el 20% restante de tu calificación.

Método de ejecución:

Para cada uno de los siguientes pasos, puedes utilizar el documento “GitHub Guía Paso a Paso” entregada por tu profesor moderador.

Como Iniciar

- Como equipo seleccionen a un líder.
- Cada uno de los miembros del equipo registre una cuenta personal en <https://github.com>.

- Como líder de equipo, crea el proyecto de nombre ProyLPEquipo1, donde 1 es el número que se haya asignado a tu grupo de trabajo.
- Como líder incluye a cada uno de los miembros de tu equipo como colaborador de tu proyecto.
- Como líder a más de añadir a tus compañeros como colaboradores, añade la cuenta de tu profesor moderador “natalita”.

Como ejecutar el proyecto

- Instala la aplicación GitHub para tu sistema operativo de preferencia y colabora en la ejecución del proyecto.
- Como equipo dividan el proyecto en varios archivos y funciones para que su aplicación sea modular.
- Como equipo repartan de manera equitativa el esfuerzo en el proyecto.
- Como equipo conversen y compartan para la construcción de la aplicación.
- Su profesor moderador revisará diariamente los avances de su proyecto en la plataforma GitHub, en base a las revisiones enviará una retroalimentación a los emails personales de cada uno de los miembros del equipo.
- Revisa continuamente en tu email personal la retroalimentación entregada por tu profesor moderador.
- En el caso de que la retroalimentación sugiera algún cambio, realiza los ajustes necesarios a tu aplicación. Conversa con el resto de los miembros del equipo acerca de la retroalimentación y cambios que deban realizar en el proyecto.
- En el caso de que tengas dudas o comentarios relacionados con el proyecto, o con la retroalimentación que te sea enviada, siéntete libre de enviar un email. El mismo será respondido por tu profesor moderador el siguiente día de revisión.
- En el caso de que tengas dudas que deban ser resueltas personalmente, pregunta a tu profesor moderador en las horas de consulta.
- Revisa la rúbrica de calificación para que consideres temas importantes para una buena valoración/calificación de tu proyecto.
- La revisión y retroalimentación de tu moderador se realizará hasta 2 días antes de la presentación final de tu proyecto.
- Hasta aquí habrás alcanzado el 80% de la calificación de tu proyecto.

Como finalizar

- Prepárate para conseguir un 20% en la presentación final.
- Preparen una presentación práctica de su aplicación con tu equipo.
- Demuestra tu conocimiento en las preguntas que tu profesor moderador te realizará en el momento de la presentación. Las preguntas estarán centradas en la materia y en el código fuente que hayas aportado en GitHub. Tu moderador no preguntará acerca de código aportado por otros miembros de tu equipo.
- El tiempo de la presentación no deberá superar los 15 minutos.

Rúbrica de calificación: Tu proyecto será calificado sobre 4 puntos. A continuación tienes los detalles para tu calificación:

<Aquí se ha presentado a los estudiantes la rúbrica de evaluación propuesta por el método>