RECURSOS DEL SOFTWARE LIBRE PARA FAVORECER LA ENSEÑANZA, APRENDIZAJE E INNOVACIÓN EN CURSOS DE PROGRAMACIÓN DE COMPUTADORES EN INGENIERÍAS

MIGUEL ÁNGEL TOVAR CARDOZO

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
CONVENIO CON LA UNIVERSIDAD DE OBERTA DE CATALUÑA ESPAÑA
MAESTRÍA EN SOFTWARE LIBRE
2014

RECURSOS DEL SOFTWARE LIBRE PARA FAVORECER LA ENSEÑANZA, APRENDIZAJE E INNOVACIÓN EN CURSOS DE PROGRAMACIÓN DE COMPUTADORES EN INGENIERÍAS

MIGUEL ÁNGEL TOVAR CARDOZO

Trabajo de grado para optar el título de Magister en Software Libre

Director
Mg. JOSÉ DANIEL CABRERA CRUZ

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
CONVENIO CON LA UNIVERSIDAD DE OBERTA DE CATALUÑA ESPAÑA
MAESTRÍA EN SOFTWARE LIBRE
2014

AGRADECIMIENTOS

El autor ofrece sus agradecimientos a todos los estudiantes de los grupos participantes, al profesor del curso que dentro de la población muestra cumplió las excelentes funciones de un grupo control, Esp. Luis Miguel Arguello, y expresa su especial agradecimiento al Director de la presente tesis, Mg. José Daniel Cabrera Cruz, quien con generosidad y motivación, le brindó la oportunidad de obtener durante la realización del trabajo investigativo, sus apreciados y relevantes aportes.

RESUME

Este documento presenta una investigación llevada a cabo en el curso de Programación de Computadores, ofrecido a las Carreras de Ingeniería de la Universidad Antonio Nariño sede Neiva. El estudio, titulado "Recursos del Software Libre para Favorecer la Enseñanza, Aprendizaje e Innovación en cursos de Programación de Computadores en Ingenierías", partió de la formulación de la pregunta ¿Cómo se pueden aprovechar los recursos que brinda el Software Libre para potenciar los procesos de enseñanza aprendizaje e innovación en el marco de los cursos de la Programación de Computadores en programas de ingeniería? Y se decidió enmarcarse en una investigación de corte cualitativo para abarcar su objetivo de proponer e implementar herramientas y métodos del campo del Software Libre para favorecer la enseñanza y el aprendizaje de la Programación de Computadores con orientación a la innovación en cursos de Ingeniería. Con el trabajo investigativo se logró corroborar que con el uso de software no privativo se pueden fortalecer los procesos de aprendizaje garantizando la abierta participación del estudiante. Aquí se parte de un diagnóstico local, hasta llegar a una propuesta de intervención parcial que incorpora la herramienta Blockly y un sistema de gestión de aprendizaje (LMS), el cual permite a los estudiantes acceder a contenidos teóricos de la materia y también evaluar sus avances por medio de cuestionarios. Se hace referencia, entonces, a Chamilo LMS, como la más indicado para esta tarea.

Palabras clave: Software Libre, programación de Computadores, herramientas innovadoras, Ingenierías.

CONTENIDO

pág.

INTRODUCCIÓN 14

1. PLANTEAMIENTO Y PREGUNTA DE INVESTIGACIÓN 15

2. OBJETIVOS 20

2.1 OBJETIVO GENERAL 20

2.2 OBJETIVOS ESPECÍFICOS 20

3. MARCO REFERENCIAL 21

3.1 MARCO CONCEPTUAL 21

3.1.1 Software Libre 21

3.1.2 Programación de Computadores y Educación en Programación de Computadores 23

3.1.3 Innovación y Educación para la innovación 24

3.1.4 Educación en Ingeniería 26

3.1.5 Formación en Tecnología 27

3.1.6 Programación informática 28

3.1.7 Innovación Tecnológica 28

3.1.8 Educación o enseñanza técnica 29

3.1.9 Educación en innovación 29

3.2 MARCO TEÓRICO 31

3.2.1 Software Libre en innovación 32

3.2.2 Teorías Referidas al Software Libre en Educación 32

3.2.3 Teorías Referidas al Software Libre en Programación de Ingenierías 33

3.3 ESTADO DEL ARTE 35

3.3.1 Enseñanza de la programación de computadores en Colombia y el mundo 36

3.3.2 Uso del Software Libre como estrategia educativa 38

3.3.3 Innovación en la enseñanza del Software Libre 41

3.3.4 Software Libre, Innovación y Programación de Computadores 43
3.3.5 Programación de Computadores haciendo referencia a Software Libre 45
3.3.6 Innovación en educación en ingenierías 45
3.3.7 Innovación en Programación de Computadores 46
3.3.8 Software Libre haciendo Referencia a la Programación de Computadores 47
3.4 MARCO NORMATIVO 48
3.4.1 Enseñanza de la Programación según ACM 48
3.4.2 Enseñanza de la Programación según ABET 48
3.4.3 Estándar para Software Libre 49
3.5 MARCO INSTITUCIONAL 50
4. DESCRIPCIÓN PROCESO INVESTIGATIVO 51
4.1 ENFOQUE DE LA INVESTIGACIÓN 51
4.2 TIPO DE INVESTIGACIÓN 51
4.3 ASPECTOS METODOLÓGICOS 52
4.3.1 Población, muestra y grupo control 52
4.3.2 Instrumento de Recolección de Información 52
4.4 DESCRIPCIÓN DE LAS ACTIVIDADES 53
4.4.1 Diagnóstico de la enseñanza, aprendizaje e innovación en cursos de Programación de Computadores 53
4.4.1.1 Revisión de literatura selecta y relevante sobre problemas o dificultades para favorecer la enseñanza, aprendizaje e innovación en cursos de Programación de Computadores 54
4.4.1.2 Síntesis de la revisión realizada 54
4.4.1.3 Auto-reflexión del investigador como docente de programación de computadores para identificar problemas 54
4.4.1.4 Entrevista a otros docentes de Programación de Computadores en busca de problemas 56
4.4.1.5 Entrevistas a estudiantes de semestres que ya hayan tomado el curso de Programación de Computadores 56
4.4.1.6 Síntesis de los problemas identificados en entrevistas, auto-reflexión y revisión bibliográfica 56

4.4.1.7 Diligenciamiento de un cuadro diagnóstico que resuma, describa los problemas e identifica oportunidades de uso del software libre 60

4.4.1.8 Identificación de herramientas o recursos del software libre y maneras o métodos en que se han usado en la enseñanza y aprendizaje de Programación de Computadores 60

4.4.2 Comparación de métodos o herramientas identificados considerando el diagnóstico realizado 64

4.4.2.1 Revisión de la literatura científica o en línea sobre herramientas o recursos del software libre y maneras o métodos como se han usado para la educación en Programación de Computadores 64

4.4.2.2 Síntesis de la revisión bibliográfica realizada sobre herramientas y métodos 66

4.4.2.3 Elaboración de un cuadro que resuma las herramientas y recursos del software libre y la manera en que han sido usados para la educación en programación de computadores 67

4.4.3 Revisión bibliográfica sobre criterios de calidad del software libre, ingeniería de software y software educativo 67

4.4.3.1 Definición de criterios de comparación de herramientas y recursos del software libre para su uso en educación para Programación de Computadores 72

4.4.3.2 Realización de la comparación entre herramientas y métodos utilizando los criterios definidos 73

4.4.3.3 Diligenciamiento de una matriz que sintetice la comparación de herramientas y métodos y su confrontación con los problemas identificados en el diagnóstico 73

4.4.4 Modificación del plan de curso de la asignatura de Programación de Computadores ofrecida a programas de ingeniería de la Universidad Antonio Nariño 73

4.4.4.1 Selección del curso de Programación de Computadores que se intervendría 77

4.4.4.2 Revisión del plan del curso seleccionado 77

4.4.4.3 Identificación de las posibles modificaciones al plan de curso para la incorporación de recursos y herramientas del software libre 77

4.4.4.4 Definición de las reformas al plan de curso para incorporar herramientas y métodos del software libre 78

4.4.4.5 Realización de las reformas definidas al plan de curso 78

4.4.4.6 Selección de los grupos de Programación de Computadores de ingeniería en los cuales se haría la implementación o prueba parcial 79

4.4.4.7 Definición de las actividades del plan de curso reformado que se realizarían en los cursos que se van a intervenir 79

4.4.4.8 Diseño de un instrumento de recolección de información de los grupos a los que se haría seguimiento 80

4.4.4.9 Programación y organización de las actividades que se realizarían en los cursos a los que se haría seguimiento (Grupo Experimental) 80

4.4.4.10 Ejecución de las actividades definidas de incorporación del software libre en los grupos seleccionados 83

4.4.4.11 Aplicación del instrumento de recolección de información en los grupos seleccionados 87

4.4.4.12 Análisis y reflexión a partir de la información recolectada en los grupos a los que se hizo seguimiento 88

5. RESULTADOS 94

5.1 Cuadro de diagnóstico de problemas en los cursos de Programación de Computadores en ingeniería y oportunidades del uso del software libre 94

5.2 Matriz que confronta recursos del SL y el diagnóstico realizado a los cursos de Programación de Computadores 96

5.3 Plan de curso reformado de la materia Programación de Computadores en programas de ingeniería 97

5.4 Resultados de la implementación parcial del plan de curso reformado de Programación de Computadores 100

6. CONCLUSIONES 101

6.1 Estado en que se deja el Problema de implementación 101

6.2 Estado en que se deja la pregunta e hipótesis de investigación 102

6.3 Relevancia de los resultados del proyecto 102

6.4 Aporte del proyecto al estado del arte 103

7. RECOMENDACIONES 105

7.1 Aspectos del problema de investigación que quedan pendientes 105

7.2 Aspectos de la pregunta de investigación que quedan pendientes 106

7.3 Dificultades que se presentaron en el proyecto y modos en que se podrían resolver en futuros trabajos 106

7.4 Cómo se podría continuar la investigación en futuros trabajos 107
REFERENCIAS BIBLIOGRÁFICAS 108
ANEXOS 117

LISTA DE TABLAS

pág.

Tabla 1. Síntesis revisión -problemas en aprendizaje e innovación en cursos de Programación de Computadores 55

Tabla 2. Herramientas y casos de éxito 68

Tabla 3. Programación y organización de las actividades 81

LISTA DE CUADROS

pág.

Cuadro 1. Problemas y oportunidades de uso del software libre en Programación 94

Cuadro 2. Métodos o herramientas considerados 96

Cuadro 3. Planeador de la Asignatura por Contenidos 98

LISTA DE FIGURAS

pág.

Figura 1. Razones para que las ingenierías se formen en la innovación 31

Figura 2. Relación de licencias contempladas por el Software Libre 50

Figura 3. Herramientas innovadoras 74

Figura 4. Herramientas para acompañar proceso enseñanza-aprendizaje 75

Figura 5. Matriz herramientas/Problemas diagnosticados 76

Figura 6. Captura de pantalla Página Principal propuesta del docente investigador 83

Figura 7. Interfaz para el Docente 83

Figura 8. Creación de un documento 84

Figura 9. Trabajo en los Foros 84

Figura 10. Vista desde el perfil de un estudiante 85

Figura 11. Introducción a Blockly 85

Figura 12. Ejercicio de cálculo 86

Figura 13. Ejercicio Libre 86

Figura 14. Creando Aplicaciones 87

LISTA DE ANEXOS

pág.

Anexo A. Instrumento Guía de Preguntas Entrevista a docentes 56

Anexo B. Instrumento Guía de Preguntas Entrevista a estudiantes 56

Anexo C. Cursos de Programación de Computadores (Población muestra) 77

Anexo D. Selección estudiantes de Programación de Computadores 79

Anexo E. Grupo Control 79

Anexo F. Instrumento Guía de preguntas Entrevista a Población Muestra 80

INTRODUCCIÓN

Acudir a las TIC con la firme idea de fortalecer el proceso de enseñanza aprendizaje, se viene convirtiendo en una oportunidad para que el curso de Programación de Computadores de la facultad de Ingeniería en la Universidad Antonio Nariño de Neiva, replantee su prácticas; por supuesto, esto sin alejarse de los saberes, los aprendizajes, los valores y los compromisos que impone el Plan de Estudios. Por ello, este documento se concibió pensando en ahondar en los niveles de acercamiento a las herramientas que ofrece el Software Libre, y el aprovechamiento de estas tanto en clase, como fuera de ella.

Propiciar un horizonte de mirada a los actuales estudiantes y a otros docentes que participan en la materia Programación de Computadores, en medio de los vertiginosos cambios a los que nos vemos abocados en la últimas dos décadas, amplía el ángulo de visión y agudiza los sentidos, de tal forma que se encuentra para ellos también, un lugar propio dentro del mundo globalizado que toca a las puertas de la Universidad indudablemente.

Se aclara, entonces, que este estudio se preocupa por escudriñar el tema de la enseñanza aprendizaje del curso Programación de Computadores, enfocándolo en la innovación, asumiendo que el docente de esta asignatura comprende que la creatividad, el pensamiento propositivo y el razonamiento lógico como habilidades esenciales en las diversas ingenierías, pueden desarrollarse con mayor amplitud si su curso lo apoya en recursos del software libre. Eso es precisamente parte de lo que se puede descubrir con proyectos como el aquí abordado.

1. PLANTEAMIENTO Y PREGUNTA DE INVESTIGACIÓN

Desde hace algunos años, los programas de Ingeniería ofrecidos por la Universidad Antonio Nariño con sede en Neiva, y en especial su asignatura de Programación de Computadores, vienen reconociendo con ahínco la importancia de las TIC en los procesos de aprendizaje. Sin embargo, a la par de la búsqueda de su vinculación, se destaca la necesidad de incorporar herramientas digitales que apoyen realmente su aprendizaje, pues esta materia es de las que reporta mayor dificultad en las carreras de ingeniería que la acogen dentro de su *pensum*. La realidad es que el docente de la asignatura de Programación de Computadores, enfrenta un cúmulo de problemas que, después de una concienzuda reflexión, se pueden organizar alrededor de tres ejes básicos: El docente de Programación de Computadores, el estudiante de dicho curso, y los problemas relacionados con los recursos innovadores y el acogimiento de software libres para el proceso enseñanza aprendizaje. En esos términos, se puede describir la actual problemática vivida en la asignatura, de la siguiente manera:

A. En cuanto al docente.

- Algunos docentes de programación de computadores tienen recursos y estrategias didácticas exitosas para enseñar temas específicos de sus cursos. Sin embargo, estas ventajas están disponibles apenas para los estudiantes que asisten a sus clases; es difícil que los estudiantes de otros cursos se beneficien también. No existen facilidades para que todos los docentes puedan compartir y extender sus recursos y mejores prácticas a los demás. No es sencillo compartir videos, grabaciones, materiales didácticos entre estudiantes de los diferentes cursos de programación y que permanezcan para que puedan ser utilizados en los próximos semestres por los nuevos estudiantes de programación de computadores.

- Los estudiantes llegan con deficiencias generales que afectan su aprendizaje de la programación de computadores. Por ejemplo: dificultades para identificar claramente un problema, para definir u ordenar y luego seguir un procedimiento para la solución del mismo. Si bien estas dificultades afectan el aprendizaje de otros cursos, es más apremiante en el caso de la asignatura en cuestión, en la cual estas habilidades son esenciales.

B. En cuanto al estudiante.

- Los estudiantes de carreras diferentes a ingeniería de sistemas, no entienden o no le hallan el sentido que tiene la Programación de Computadores en su ingeniería. Esto hace que no se sientan motivados o interesados en esos cursos.

- Se tiende a limitar el ofrecimiento de materiales, recursos y actividades a lo que se hace en el aula de clase con acompañamiento del docente. No se ofrece mayor atención a lo que el estudiante hace por fuera del aula (es decir, su trabajo independiente). El docente considera que el estudiante no siempre cuenta con recursos dispuestos para sacar mejor provecho de su tiempo de trabajo independiente. Adicionalmente, no resulta sencillo para el docente percibir la posibilidad de que el estudiante reciba realimentación oportuna y seguimiento de lo que está haciendo por fuera de la clase.

- Algunos docentes tratan de enseñar la programación de computadores sin utilizar los computadores. A los estudiantes de hoy en día que tienen interés y expectativas de utilizar elementos tecnológicos e informáticos, eso les resulta desmotivante, especialmente cuando se trata de un curso vinculado esencialmente con la tecnología.

- Los estudiantes de los cursos de programación de computadores a veces no ven la aplicación o uso práctico de lo que están aprendiendo. Esto puede deberse a que las actividades que se realizan dentro de los cursos se reducen a pruebas de escritorio que no se vinculan con la vida cotidiana del estudiante, como por ejemplo: el hecho de que los programas que se construyen no resuelven problemas o necesidades cotidianas que puedan ejecutarse desde sus móviles o portátiles para resolver algún problema o necesidad clara para ellos.

- Se desaprovechan los recursos y contenidos que se brindan en la clase. Los métodos que usan los estudiantes para registrar u organizar los recursos y contenidos que se brindan en clase no son muy eficientes; de modo que, con frecuencia, no pueden disponer de éstos cuando los necesitan para su estudio en futuras clases y en su tiempo de trabajo independiente.

C. En cuanto al aprovechamiento de recursos relacionados con la innovación y el acogimiento de software libre para el proceso enseñanza aprendizaje del curso:

- En los cursos de programación de computadores se desaprovechan recursos e innovaciones tecnológicas educativas existentes, tanto aquellas de carácter general (LMS, CMS, por ejemplo), como aquellas específicamente orientadas a la educación en programación de computadores. Respecto de estas últimas, se utilizan herramientas como DFD, PSINT y Zinjal, que si bien resultan un apoyo importante, no facilitan su uso desde Web con los dispositivos móviles de los estudiantes y a veces no facilitan el aprendizaje y seguimiento de los estudiantes.

Estas herramientas no ofrecen muchas alternativas diferentes y quizás más llamativas para programar, por ejemplo: compilación en línea, formas gráficas o visuales y variadas de presentación de los programas, la posibilidad de centralizar y disponer los recursos que van encontrando o generando ellos mismos. Existen hoy en día herramientas más novedosas que pueden motivar y favorecer la enseñanza y aprendizaje de la programación de computadores.

- Cuando a los estudiantes de Programación de Computadores no se les brindan opciones prácticas de participar en el aprendizaje, suelen mostrarse insatisfechos, pueden hallarse más confundidos en su proceso de aprendizaje, y no le encuentran sentido a lo que el docente les plantea en el currículo: no descubren la utilidad para su vida profesional. Es ahí donde aparece el Software Libre como una opción para descartar problemas como estos.

- Tradicionalmente en el aula se enseña a programar con lenguajes y herramientas privativas (no libres) que se limitan a lo que ofrece la casa desarrolladora de estas herramientas.

- Por desconocimiento o cultura de los docentes dentro del curso de Programación de Computadores, no se proponen herramientas de software libre y si se hace, no enfatizan en los beneficios y en el trabajo en comunidad que es uno de las características fundamentales del software libre.

De esta manera, no se puede seguir pretendiendo que con un plan de estudios que ya cuenta con relativa trayectoria, se logre llegar a un grupo de estudiantes que cursan su primer año de pregrado (estamos hablando de jóvenes con edades que oscilan entre los 18 y 20 años) y a quienes -desde luego- si algo se les facilita, es precisamente la utilización de dispositivos electrónicos, sin antes como profesor proponerse enfocar el conocimiento curricular, a un campo donde tenga real aplicabilidad y sentido para ellos: el tecnológico.

Para los nacidos en esta era digital, y que se les considera “usuarios permanentes de las tecnologías con una habilidad consumada” (Benito, García, Portillo, & Romo, 2007), el currículo debe ser otro: Uno que de alguna manera tenga claro sus necesidades de comunicación, información y, desde luego, de formación. Los expertos aseguran que estos nuevos usuarios enfocan el aprendizaje de nuevas formas: absorben rápidamente la información multimedia de imágenes y videos, igual o mejor que si fuera texto; consumen datos simultáneamente de múltiples fuentes; esperan respuestas instantáneas; permanecen comunicados permanentemente y crean también sus propios contenidos. Funcionan mejor trabajando en red, puesto que

> el nativo digital en su niñez ha construido sus conceptos de espacio, tiempo, número, causalidad, identidad, memoria y mente a partir, precisamente, de los objetos

> digitales que le rodean, pertenecientes a un entorno altamente tecnificado. Hay quien sostiene que el crecimiento en este entorno tecnológico puede haber influido en la evolución del cerebro de aquellos individuos. (Benito, García, Portillo, & Romo, 2007)

Esta necesidad le demanda al docente de la materia, saber cómo diseñar y manipular la realidad a través de la tecnología, considerando importante su rol para el éxito académico, la disminución de las tasas significativas de deserción, cancelación y reticencia del programa, y la consecuente efectividad de los futuros profesionales. Se requiere determinar qué dirección debería tomar la innovación en las ingenierías, así como en todos los niveles formativos, para adecuarse a las características de los nativos digitales. No en vano se ha planteado que los efectos que producen las nuevas tecnologías demandan estudios con una perspectiva pedagógica, debido a que la enseñanza no se origina en la renovación del equipo, sino más bien por la reestructuración de los enfoques pedagógicos. (Maggio, 2002)

Además, reconociendo que existe una necesidad real de transmitir conocimientos mediante las redes de comunicación e Internet, puesto que forman parte del mundo académico, se acepta que aun teniendo una plataforma web muchos usos, y que incluso admite herramientas para comunicarse con los alumnos de manera bidireccional, presenta un inconveniente: demanda de la existencia de un profesional que instale o configure una aplicación específica. Ante esta situación, se vislumbran las herramientas llamadas gestores de contenidos, las cuales resultan sin duda de vital importancia en el aula donde la tecnología de la transmisión de la información ocupa hoy un lugar privilegiado.

En esos términos, se puede considerar que el software libre, puede convertirse en un ingrediente valioso en las Ingenierías. El conjunto de las propiedades que brinda el software libre, puede llegar a ser la alternativa de solución del problema, y con todo su ambiente creador, puede contrarrestar las dificultades que los estudiantes manifiestan dentro de la asignatura para aprender a programar, a tal punto que los estudiantes puedan abrir y estudiar un programa, puedan hacerle cambios y aprender viendo lo que pasa con cada cambio, sin impedimentos o restricciones ellos lograrán mezclarse con otras rutinas o programas libres.

En Colombia, en todos los sectores, hay una preocupación fundamental: es preciso atraer y formar talento que hagan parte de la era de la información y el conocimiento, es innegable que actualmente no se logren llenar todas las vacantes mundiales exigiendo personal con conocimientos en lenguaje de programación; se hace vital, por lo tanto, en estos momentos, incentivar este aprendizaje en las actuales generaciones. Sin embargo, se recuerda que, aunque para algunos ingenieros la programación es una gran ayuda a

la hora de resolver algunos problemas tecnológicos, no necesariamente ellos la estudian para ser programadores.

Planteada la necesidad de la asignatura Programación de Computadores en los pregrados de Ingeniería ofrecidos por la Universidad Antonio Nariño sede Neiva, y precisando el respectivo problema, surge el siguiente interrogante:

¿Cómo se pueden aprovechar los recursos que brinda el Software Libre para potenciar los procesos de enseñanza aprendizaje e innovación en el marco de los cursos de la Programación de Computadores en programas de ingeniería?

2. OBJETIVOS

2.1 OBJETIVO GENERAL

Proponer e implementar herramientas y métodos del campo del software libre para favorecer la enseñanza y el aprendizaje de la programación de computadores con orientación a la innovación en cursos de ingeniería.

2.2 OBJETIVOS ESPECÍFICOS

- Diagnosticar problemas que se presentan en la enseñanza y aprendizaje de la Programación de Computadores con orientación a la innovación en cursos de ingeniería y que brinden oportunidades de uso de software libre, con base en la observación de algún curso existente y la revisión bibliográfica.

- Identificar herramientas o recursos que brinda el software libre y maneras o métodos en que se han usado para favorecer la enseñanza, el aprendizaje y la innovación en cursos de Programación de Computadores en ingeniería.

- Comparar herramientas o recursos y métodos identificados que ofrece el software libre para la enseñanza, aprendizaje e innovación en la Programación de Computadores, considerando el diagnóstico realizado.

- Plantear e implementar una propuesta de uso de herramientas y métodos de software libre para favorecer la enseñanza, aprendizaje e innovación en algún curso de Programación de Computadores en ingeniería.

3. MARCO REFERENCIAL

3.1 MARCO CONCEPTUAL

En este capítulo, se pretende definir y explicar los referentes que rodean al tema del software libre, cuando la pretensión es favorecer la enseñanza y el aprendizaje en cursos de programación de computadores en ingenierías. Para poder hacerlo,
es necesario primero definir varios conceptos, entre los cuales se encuentra el de software libre, de tal manera que se logre comprender qué es y en qué se diferencia de otro tipo de software; en qué consiste la innovación del proceso enseñanza aprendizaje, y desde luego, la forma en la que sebe comprender la Programación de computadores.

3.1.1 Software Libre. El software libre se define como un tipo de licenciamiento (Rosa & Heinz, 2007, pág. 29), para decirlo con el documento de la UNESCO, por lo que estos expertos prefieren llamarlo "software licenciado bajo condiciones libres".

Por su parte, la *Free Software Fundation* (FSF) aclara entre un sencillo lenguaje que software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software (Stallman, 2004)[1].

- **Libertad 0:** La libertad de usar el programa, con cualquier propósito, en cualquier lugar y para siempre.
- **Libertad 1:** La libertad de estudiar cómo funciona el programa, y adaptarlo a las propias necesidades del usuario. El acceso al código fuente es una condición previa para esto.
- **Libertad 2:** La libertad de distribuir copias, con lo que se pueda ayudar a un vecino.
- **Libertad 3:** La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. El acceso al código fuente es un requisito previo para esto.

[1] Richard Stallman es el fundador del movimiento de *software libre* y creador del concepto *copyleft,* pues utiliza los aspectos legales del derecho de autor para proteger las 4 libertades del *software*. También es el presidente y fundador de la Fundación de *software* Libre, así como el desarrollador principal del proyecto GNU (que significa, GNU *is not Unix*). Con el *kernel* o núcleo del sistema operativo, desarrollado por *Linus Torvalds,* Stallman y un equipo de hackers crearon GNU*Linux* (el conocido sistema *Linux*).

Sin embargo, estas libertades, para decirlo con González (2003), se pueden garantizar de acuerdo con la legalidad vigente por medio de una licencia. En esta se plasman las libertades, pero también restricciones compatibles con ellas, como dar crédito a los autores originales si se da la redistribución. Incluso puede obligar a que los programas ajenos mejorados por el propio usuario también sean libres, promoviendo así la creación de más software libre.

Al pretender que las libertades 1 y 3 (la libertad para hacer cambios y para publicar las versiones mejoradas) adquieran significado, se debe disponer del código fuente del programa. Es decir que gozar de accesibilidad al código fuente, es requisito para el software libre.

Software protegido con copyleft: El copyleft es una idea impulsada por Stallman. Desde su proyecto GNU él explica que si los desarrolladores de software privativo usan copyright es para quitar libertad a los usuarios, ya que ellos usan los derechos reservados para garantizar su libertad. Por eso transforman la palabra copyright en copyleft, y se identifica con una C invertida (Stallman, 2004). En otras palabras, copyleft es la "norma que establece que, al redistribuir el programa, no pueden añadirse restricciones que nieguen a los demás sus libertades centrales" (Alfonso R. & Segnini R., 2009, pág. 107)

Sin embargo, hay que indicar que el software libre sin copyleft también existe.

> El software protegido con copyleft es software libre cuyos términos de distribución no permiten a los redistribuidores agregar ninguna restricción adicional cuando estos redistribuyen o modifican el software. Esto significa que cada copia del software, aun si ha sido modificada, debe ser software libre. (Prendes Espinosa, 2009).

En el proyecto GNU, tal como lo propone Stallman, se protege casi todo el software que se desarrolla, con el ánimo de dar a cada usuario las libertades que el término software libre abarca, y encarna una fuente de posibilidades y prerrogativas "para reducir, la brecha tecnológica existente en los países en vía de desarrollo. El uso de tecnología de bajo costo y de estándares abiertos constituye una oportunidad para el crecimiento económico, social y educativo" (Bitocchi, 2013).

Software educativo: El término se emplea para designar genéricamente los programas para ordenador, creados con la finalidad específica de servir como medio didáctico, es decir, para facilitar los procesos de enseñanza aprendizaje. Cisneros advierte aclarando el término, que el concepto de software educativo se usa para referirse a "todo software que fue diseñado con una finalidad didáctica, para apoyar, reforzar y, evaluar el

aprendizaje en cualquier de sus modalidades y momentos" (Cisneros Arocha & Cisneros Arocha, 2008).

De acuerdo con Ceja, Software Educativo "es aquel programa creado con la finalidad específica de ser utilizados como medio didáctico; es decir, para facilitar los procesos de enseñanza y aprendizaje, tanto en su modalidad tradicional presencial, como en la flexible y a distancia" (Naula Daquilema, 2011).

El software educativo se caracteriza por llevar a cabo diversas funciones, entre las cuales se mencionan (Marquès Graells, 1995):

- Una función informativa estructuradora de la realidad.
- Una función instructiva, que promueve actividades de los alumnos orientadas al logro de los objetivos educativos.
- Una función motivadora, al considerar elementos para motivar a los alumnos y dirigirlos hacia los aspectos más importantes de las actividades.
- Una función lúdica se incorpora a menudo para aumentar la motivación.
- Además, por lo general, se considera también una función evaluadora del trabajo realizado por los alumnos.

Marqués también enriquece aún más este referente, cuando entre las características que deben poseer este tipo de software, asegura que se destacan las siguientes (Marquès Graells, 1995):

- Debe permitir una interacción fácil entre el software y el estudiante
- Debe guiar al estudiante hacia el aprendizaje.
- Debe propiciar la participación del estudiante.
- Debe poseer un componente evaluativo.

3.1.2 Programación de Computadores y educación en programación de computadores. Cuando Adell y Bernabé (2007, pág. 34) recuerdan que un programa no es más que un conjunto de instrucciones que le dicen al ordenador qué tiene que hacer, y que los programas los escriben los seres humanos utilizando lenguajes de programación, necesariamente se está refiriendo a la parte central en la temática del curso Programación de Computadores. Pero también las autoras mencionan en su documento que

> antes de que el ordenador pueda ejecutar un programa, es necesario traducir dichas instrucciones a su lenguaje, esto es a "código máquina": largas series de ceros y unos. (…) El proceso de convertir un programa escrito en un lenguaje de programación a instrucciones inteligibles para el ordenador, se denomina "compilación" y lo hacen otros programas de ordenador especializados: los compiladores. Una vez el programa está compilado ya es posible ejecutarlo, a cambio, una vez "traducido" a código máquina, es casi imposible que un ser humano entienda algo de la larga serie de unos y ceros en que se ha convertido. Y este es el meollo del software libre. (Adell & Bernabé, 2007)

Cuando se parte de reconocer que el término Software educativo se emplea para designar genéricamente los programas para ordenador, creados con la finalidad específica de servir como medio didáctico, es decir, para facilitar los procesos de enseñanza aprendizaje, se puede hablar de Programación de Computadores y educación en programación de computadores. Además, Cisneros advierte que el concepto de software educativo se usa para referirse a "todo software que fue diseñado con una finalidad didáctica, para apoyar, reforzar y, evaluar el aprendizaje en cualquier de sus modalidades y momentos" (2008, pág. 11).

3.1.3 Innovación y Educación para la innovación. Frente al tema, se debe partir de aclarar el vocablo *innovar*; y ante este, acudir al concepto que brinda la Real Academia de la Lengua. Allí, es fácil detectar la connotación que concierte a este estudio, pues asegura que innovar no es simplemente hacer cosas novedosas; es desarrollar ideas que tienen un efecto concreto en la productividad. Creación o modificación de un producto, y su introducción en un mercado (Real Academia Española, DRAE). Son ideas aplicadas exitosamente. Pero para aclarar aún más el término, Duque (2009, pág. 108) propone romper con la idea común que existe de que innovar es simplemente tener una idea novedosa. Innovar, según él, va un poco más allá; la innovación requiere efectivamente que esa idea llegue a ser utilizada por la sociedad exitosamente.

En otra conferencia, Duque Escobar expone que

> no basta la actividad de la ingeniería para innovar. La innovación se presenta a menudo en el lugar de actividad cuando se observan críticamente los procesos y productos. La innovación involucra a la organización completa, a la sociedad completa: productores, consumidores, organismos de regulación, de investigación. La innovación es una actitud contraria a contentarse con lo que se tiene y se hace. (2009, pág. 50)

Ahora bien, con el avance de las tecnologías y las necesidades de la sociedad actual en cuanto a la apropiación de las mismas, las instituciones de educación deben desarrollar metodologías innovadoras para el aprovechamiento de las TIC.

Para Salinas (2004), la innovación va asociada a planificación y mejora:

> Si consideramos la innovación como la selección, organización y utilización creativa de recursos humanos y materiales de formas novedosas y apropiadas que den como resultado el logro de objetivos previamente marcados, estamos hablando de cambios que producen mejora, cambios que responden a un proceso planeado, deliberativo, sistematizado e intencional, no de simples novedades, de cambios momentáneos ni de propuestas visionarias. (pág. 36)

Elementos que permiten distinguir 'innovación' de 'cambio':

a) Innovación supone una transformación significativa e implica un cambio en nuestra concepción de enseñanza, que obviamente repercutirá en nuestra práctica educativa, en nuestros hábitos... con el fin de mejorar la calidad del aprendizaje. Y este proceso comienza en los centros TIC con una reflexión previa (que se concreta en sus proyectos) que pretende dar respuesta a unas necesidades detectadas en su entorno. Este es el punto de partida para una transformación que debe ser gradual, que abre el camino a ese proceso de innovación.

b) La innovación *no* es un fin, es un medio para mejorar la calidad y conseguir con mayores garantías los fines que se persigue en los centros educativos. Si consideramos innovación el hecho de la llegada de equipos informáticos sin que se produzca otro tipo de cambios, lo que habrá ocurrido realmente es que se ha cambiado algunos recursos, pero no un cambio significativo en la enseñanza.

c) Innovación *no* implica necesariamente una invención, aunque sí un cambio que propicia una mayor calidad. En este proceso debemos apoyarnos en lo existente y complementarlo con lo "nuevo". Los centros TIC no son "revolucionarios" en el sentido de desterrar lo anterior, sino innovadores al incorporar elementos nuevos que se suman a los que se disponía, para lograr una enseñanza de calidad.

d) Innovación *sí* implica una intencionalidad o intervención deliberada. La aprobación de los proyectos a los centros TIC supone un reconocimiento a dicha labor de planificación intencionada que se hace desde los centros solicitantes. Y esa planificación, obviamente, debe ser controlada, revisada... periódicamente y reflexionar, desde la práctica, sobre los cambios que se produzcan en los centros. Como con cualquier previsión, si a lo largo del desarrollo del proyecto es necesario modificarlo, para que se convierta en un instrumento útil, y debido a que se requiera su adecuación a las circunstancias que se produzcan en la práctica diaria, debe hacerse.

En muchas ocasiones, como especifica Cebrián (Cebrián de la Serna, 2004) la innovación educativa suele venir asociada a la adaptación, revisión y/o producción de materiales educativos. Esto no tiene por qué ser necesariamente así, pues el profesorado debe saber cómo explotar didácticamente los recursos, no tiene por qué elaborarlos, aunque a veces requiera adaptarlos y modificarlos a una situación personal

3.1.4 Educación en ingeniería. La pedagogía y la didáctica proveen elementos para asumir las relaciones complejas entre el instructor y el aprendiz, respecto al quehacer cognitivo alrededor de un objeto de estudio, y el papel que juegan los diferentes instrumentos incorporados al proceso. Se toma como referencia la propuesta pedagógica de la teoría constructivista del aprendizaje, que describe el proceso de aprender, los distintos modos de representación y las características de una teoría de la instrucción. Partiendo de los postulados pedagógicos, se define los principios aplicados al área de conocimiento del interés particular, y se formulan las estrategias que deben ser tenidas en cuenta dentro del proceso de enseñanza de la ingeniería de software.

Postulado Pedagógico: *El aprendizaje como proceso de categorización.* El aprendizaje consiste esencialmente en la categorización de las cosas con el propósito de simplificar la interacción con la realidad y facilitar la acción (Bruner, 2001, citado por Anaya). La categorización está estrechamente relacionada con procesos como la selección de información, generación de proposiciones, simplificación, toma de decisiones y construcción y verificación de hipótesis. El aprendiz interactúa con la realidad organizando las entradas según sus propias categorías, posiblemente creando nuevas o modificando las preexistentes. El aprendizaje es por lo tanto un proceso activo, de asociación y construcción (Anaya, 2006).

Sin embargo, en medio del concepto que rodea a la ‘educación en ingenierías’, este estudio se permite ahondar en un panorama que expone con argumentos soportados, el déficit de ingenieros tanto en Colombia como en el mundo occidental, se precisa tener en cuenta la formación en ingeniería, puesto que las diversas ingenierías son carreras técnicas que, para decirlo con Ulloa, “no aparecen como prioritarias dentro de las opciones de vida de los bachilleres y en las universidades tenemos pocos profesionales para entregar al mercado laboral” (2010, pág. 21). El mismo Ulloa, cita el estudio "Programas de Ingeniería en Colombia, Cuarta Versión", expuesto por la Asociación Colombiana de Facultades de Ingeniería – ACOFI, presentando con verdaderos argumentos cuantitativos, que la educación en Ingeniería en este país se está convulsionando desde el 2005, aproximadamente (2010, pág. 22).

Interesante seguir citando a Ulloa, cuando afirma que “en nuestra sociedad, tan permeada por el dinero fácil y los resultados inmediatos, una profesión como la Ingeniería que requiere mayor esfuerzo no es necesariamente popular entre la juventud” (2010, pág.

2), y ante tan preocupante estado de la educación en ingeniería, la ACOFI manifiesta a través de la Dra. Lueny Morell, de Hewlett Packard, que “si los ingenieros son la clave del desarrollo económico, necesitamos innovar, reformar la educación en ingeniería, para responder mejor a los desafíos globales” (2010, pág. 4)

Ahora bien, dentro de los programas de ingenierías y por aquella urgencia de innovar (recordando a la ACOFI), se sabe que se ha convenido como estrategia pedagógica el uso de software, libre o comercial, y estos se han venido empleando como herramienta de apoyo en el proceso de enseñanza de diversas asignaturas en donde se requiere desarrollar competencias en modelación matemática, por ejemplo. Sin embargo, diversos estudios en torno a cuál tipo de software acoger, han definido que ante el software libre encuentran una “alternativa apropiada para el proceso de enseñanza y solución de diversos modelos en el entorno académico” (Araoz Cajiao , Caballero Villalobos, González Neira, & García Cáceres, 2013).

En los últimos años, ya se comienzan a percibir posturas frente al hecho de utilizar Software Propietario con los alumnos de pregrado dentro de las diversas ingenierías; ya hay quienes, dentro de la academia asegura que con ello “se está aislando a los estudiantes y los están convirtiendo en ignorantes” (Sotomayor, 2013), expresa un docente peruano. El docente y propulsor en plantear modelos de estudio utilizando herramientas de software libre para carreras técnicas y programas de diplomados, postgrado en Software Libre, sigue manifestando en su espacio web con respecto a los estudiantes de las ingenierías, que “es posible que solo sean consumidores y no desarrolladores en un mundo donde si tenemos acceso al software libre” (Sotomayor, 2013).

3.1.5 Formación en Tecnología. Bastante cercano al concepto de educación en ingeniería, es el referente de la formación en tecnología: Para nadie es un secreto que hasta finales del siglo XX las universidades no se preocupaban por la formación tecnológica, pues su eje era la formación de profesionales. Tal vez por eso afirma Rama que

> el peso desproporcionado de la matrícula en profesiones tradicionales y la baja presencia en las áreas tecnológicas, contribuyó a una mayor distancia, en términos de pertinencia, entre las universidades y un aparato productivo que buscaba diversificarse en el marco de la crisis de los modelos económicos. (2006, pág. 4)

Desde luego que a medida que se avanza en el tiempo, al lado de la investigación y de la innovación, surge la generación de tecnologías como ese “motor central del crecimiento y de la dinámica económica de las sociedades modernas” (Rama, 2006, pág. 14). Es por ello, que la formación tecnológica debe ser motivo de preocupación en las

instituciones de educación superior de cualquier país que como Colombia, se esfuerza por ganar espacios en plena era del conocimiento; las universidades ya no pueden ser maniobradas dentro de los esquemas de gestión surgidos a comienzos del siglo pasado y que aún se conservan en muchas instituciones de educación superior.

3.1.6 Programación Informática. Dentro de las carreras tecnológicas y de nivel profesional, se comienza a hacer referencia a la idea de que los programas informáticos sean desarrollados con un estilo formal, con una rigurosa pero creativa metodología, que los lleve a ganar en legibilidad y eficiencia. La Universidad de Viña del Mar, en su programa académico lo percibe cuando dice: "hay que entender y asumir la diferencia entre alguien que consigue que sus programas funcionen y alguien que no sólo elabora sus programas de forma coherente, sino que consigue mejorar la ejecución de dichos programas, ya sea en velocidad o en el consumo de recursos" (2012, pág. 2).

3.1.7 Innovación Tecnológica. Rama (2006) insiste a lo largo de su documento, que el centro de gravedad de la economía se ha ido desplazando de la producción de bienes a las actividades ligadas a los servicios, a la información, a la innovación tecnológica y a la utilización acelerada de los nuevos conocimientos. El experto asegura que bajo esas condiciones, la generación de nuevas tecnologías, al lado de la investigación, toma relevante importancia; a tal punto se debe dar lugar preponderante a la innovación tecnológica, que se hace innegable aceptar que este mundo cruza una era en la cual "el crecimiento económico es más un proceso de acumulación de conocimiento que de acumulación de capital" (2006, pág. 15).

En este contexto, autores como Ruiz y Mandado aseguran que la innovación tecnológica comprende todas "las etapas científicas, técnicas, comerciales y financieras" (1989, pág. 14) empleadas para aspectos como el desarrollo y comercialización con éxito de productos novedosos o mejorados, al punto de convertirla en un ingrediente esencial para el futuro, siendo un aspecto vital en la conservación de la prosperidad de una empresa o de una región.

3.1.8 Educación o enseñanza técnica. Dentro del sistema de Educación se aprecia la formación profesional, la enseñanza tecnológica y la formación técnica. La enseñanza técnica es una de las diversas modalidades que en el campo educativo se ofrece en diversos países. En Colombia, por ejemplo, se reconoce como una opción de formación para el trabajo junto a la educación técnica secundaria profesional, la formación técnica profesional y la formación profesional extraescolar del SENA.

La Formación Técnica es fundamental, no sólo porque ser una opción vocacional para muchos jóvenes, sino porque constituye una base relevante para apoyar la

competitividad del país. Algunos autores la aprecian como “un coadyuvante indispensable en el proceso de incorporación” (Camargo, 2006) de lo local, en lo nacional; por lo menos esa es la tónica de la gran mayoría de países en vía de desarrollo, y así lo considera el Ministerio de Educación de un vecino país como lo es Chile cuando comparte que

> actualmente, el 40% de los titulados de Educación Superior son técnicos de nivel superior y la meta es que para el año 2020 sean 60%. Por eso es tan importante aumentar la cobertura de la Formación Técnica, así como la calidad y la pertinencia de los programas para contar a corto y mediano plazo con el capital humano que Chile requiere para su desarrollo. (Ministerio de Educación de Chile, 2012)

3.1.9 Educación en innovación. Esta expresión alude al hecho de enfrentar procesos de innovación en el sistema educativo. Educar en la innovación implica asumir la urgente necesidad de renovar prácticas y políticas educativas, tal como lo manifiesta Ezpeleta (2004). Ella afirma que la investigación y la experiencia recogida en los sistemas educativos indican que las innovaciones son inseparables de los contextos y procesos institucionales, y que por lo tanto adquieren un carácter político, y con este telón de fondo, se viene también comprendiendo porqué los países desarrollados desde hace décadas vienen preocupándose por la innovación en la educación.

Al hacer referencia al proceso de educar en innovación, y reconociendo su vínculo imborrable con prácticas educativas, obligatoriamente se piensa en el rol del docente, quien en su desempeño las innovaciones sean fomentadas en el aula, de manera que afecten al desarrollo de su práctica formativa. Los docentes innovadores son profesores capaces de analizar aplicaciones e investigaciones referentes a los distintos contenidos de la enseñanza y el aprendizaje de su área de desempeño; son docentes que logran determinar situaciones problemáticas relacionadas con el proceso de enseñanza-aprendizaje, seleccionan un tipo de diseño adecuado al problema a atender, aplican estrategias, y reflexionan sobre sus resultados.

Y en ese camino, aparecen las TIC brindando opciones a los profesores y resultando ser bastante llamativas a los estudiantes. Castañeda (2011) asegura que los educadores tienen que concebir las TIC no sólo como medios para la enseñanza sino también como medios que sirven para comunicarse e influir en el aprendizaje, y como tales, requieren para la enseñanza y para el aprendizaje metodologías innovadoras y colaborativas. A diario las instituciones educativas y sus docentes, especialmente, se proponen proyectos innovadores para mejorar la práctica educativa.

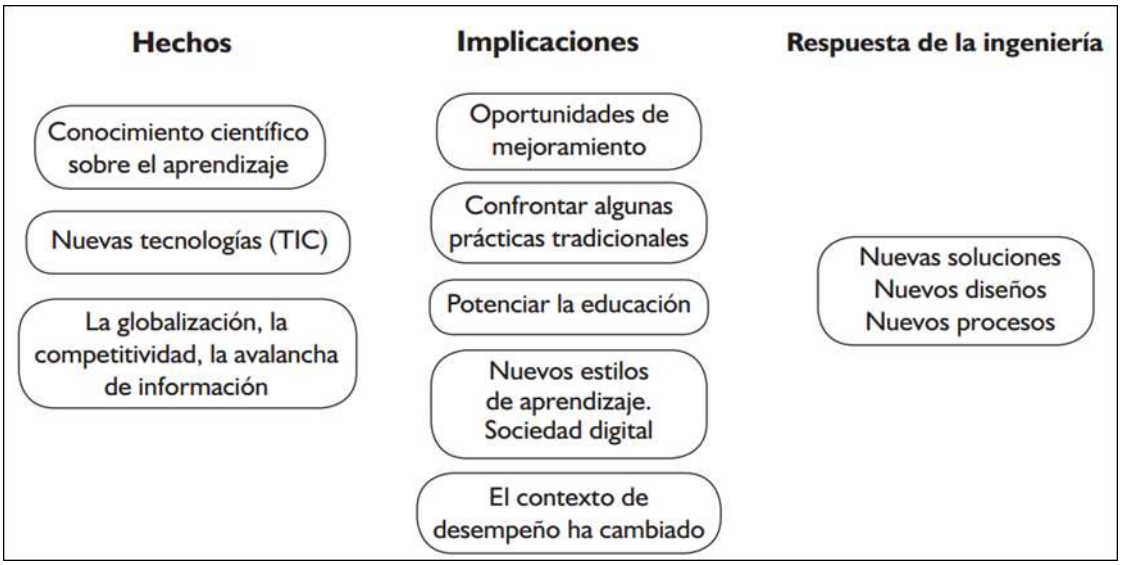
Un valor agregado a la innovación en el aula, es la formación integral. Carrasco (2000) manifiesta que en el momento actual de la educación, todos los proyectos que utilizan

métodos o técnicas de enseñanza y aprendizaje innovadoras, e incorporan esta forma de trabajo como experiencia, descubren que el sujeto que aprende se forma como persona.

Un interesante estudio, recuerda cómo en la actualidad las instituciones de educación superior desempeñan un papel muy importante en la formación de profesionales preparados para responder adecuadamente a los requerimientos de la sociedad moderna como un factor clave para incrementar la competitividad y calidad de vida (Chumba, 2009), dando cabida importante a la práctica de estrategias innovadoras; concretamente en el ámbito universitario, "la sociedad de la información y la comunicación en la que vivimos está generando nuevos replanteamientos" (Renés, s/f); a tal punto que se requiere planteamientos reflexivos, exploratorios, experimentados, colaborativos y consensuados, por parte de los miembros que constituyen los procesos de enseñanza y aprendizaje. La web 2.0 educativa en la que nos encontramos; permite la interacción, el trabajo colaborativo y el autoaprendizaje (Castaño, Maiz Olazabalaga, & Palacio, 2008), pero no por sí mismas, sino en consonancia con planteamientos pedagógicos sustanciales y realistas. Educastur ha dicho al respecto, y lo recuerda Sánchez: "Se concibe como el escenario en que convergen los usuarios, los servicios, los medios y las herramientas" (2011, pág. 2).
Es decir, la educación en la innovación propende por lograr la amalgama entre los contenidos específicos propios de la materia y el conjunto de herramientas, ya sean tecnológicas u otras, que puedan favorecer el enriquecimiento del aprendizaje de los mismos durante los procesos formativos (Renés, s/f).

Frente al tema de la necesidad de educar en la innovación, y más precisamente en la educación de las ingenierías, Duque (2009, pág. 51) acude a exponer su posición con la gráfica que se aprecia a continuación (Figura 1).

Figura 1. Razones para que las ingenierías se formen en la innovación



Fuente: Duque, En conferencia "Cómo formar ingenieros para la innovación"; 2009.

3.2 MARCO TEÓRICO

Para el presente trabajo de investigación se realizó una exhaustiva revisión bibliográfica con la cual asegurar el respectivo soporte teórico en torno al problema de cómo aprovechar los recursos que brinda el Software Libre para potenciar los procesos de enseñanza de la programación de computadores en programas de ingeniería, y mediante el cual sea posible establecer los hilos conductores del estudio. Tal visión de conjunto se desenmaraña a continuación:

3.2.1 Software libre en innovación. Con base a recordar que se entiende como software libre al software que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente, y que suele estar disponible sin costo económico en Internet, haciéndose propiedad de todos, en el sentido de que "cada persona en el mundo tiene derecho a usar el software, modificarlo y copiarlo de la misma manera que los autores de este mismo" (Hernández, 2005), pues implica ubicarlo en términos de innovación.

Hablar de software libre es hablar de innovación, en el sentido que este permite ser modificado y adaptado en función de intereses del usuario y de los objetivos que persiga (Romero, 2006) con un trasfondo creativo. Y emplearlo constituye un mayor esfuerzo en los actores sociales del proceso educativo, sobretodo en el docente. Es el profesor a quien se invita a transformar su rol, quien acogerá la innovación en primera instancia, quien conservará la validez de lo adquirido previamente, y sabiamente lo combinará con el toque participativo y activo del estudiante de hoy.

3.2.2 Teorías Referidas al Software Libre en Educación. Dentro de este marco teórico, se pueden mencionar cinco teorías cognitivas que no solamente sostienen un vínculo con el aprendizaje, sino que también se relacionan con el hecho de acoger el software libre; estas son: aprender descubriendo, aprendizaje significativo, cognitivismo, constructivismo y socio-constructivismo. Se describen, con base a los apuntes que sobre ellas teje Osorio Alvarado (2012), de la siguiente manera:

Aprender descubriendo: Esta teoría fue desarrollada por Jerome Bruner, y sostiene que el aprendizaje se realiza por la persona sobre el estudio de la realidad, un ejemplo de esta teoría podría ser la manzana que cae del árbol por Newton.

Aprendizaje significativo: La teoría del aprendizaje significativo fue desarrollada por David Paul Ausubel y J. Novak. Esta teoría se basa en que el aprendizaje no se debe memorizar sino que debe entenderse por medio de conocimientos anteriores.

Cognitivismo: Esta teoría se basa en las teorías de procesamiento de información, teoría conductista y el aprendizaje significativo; y por lo tanto, se aprecia como una mezcla de estas tres teorías.

Constructivismo: Teoría de aprendizaje desarrollada por Jean William Fritz Piaget; promulgan que debe existir flexibilidad entre los conocimientos que se poseen para que no exista resistencia al adquirir nuevo conocimiento. La flexibilidad debe ser exacta ya que si existe demasiada todo el conocimiento adquirido cambia por cualquier influencia; y si no existe flexibilidad, se produce el riesgo de no aceptar nuevos conocimientos. El constructivismo es una teoría que entiende la educación como un proceso en el cual el estudiante, de manera activa, construye nuevas ideas o conceptos basados en conocimientos presentes y pasados. Hablar de software libre en educación, es recordar al constructivismo cuando aprecia al aprendizaje como una oportunidad de construir nuestros propios conocimientos desde nuestras propias experiencias, recordando a Ormrod. (Tur Viñes, Orbea Mira, Fernández Poyatos, & Poveda Salvá, 2008).

Socio-constructivismo: Teoría basada en los pensamientos de Lev Semiónovich Vygotsky donde el aprendizaje se basa en conocimientos anteriores (Osorio Alvarado B. , 2012), pero dicho conocimiento se relaciona con la sociedad (culturas). Moodle es un proyecto inspirado en la pedagogía del constructivismo social (González Mariño, s/f), por ejemplo. “Utilizando Moodle como Entorno Virtual de Aprendizaje obtendremos un sistema flexible donde, además de aprender los alumnos pueden compartir experiencias de aprendizaje y conocimientos con otras comunidades virtuales, compuestas por otros

usuarios de la plataforma en todo el mundo" (González Mariño J. , 2006).

3.2.3 Teorías Referidas al software Libre en Programación de Ingenierías. En esta parte del marco teórico, se hace indispensable recurrir como un principal referente de la investigación, a los postulados de la *Free Software Foundation* (FSF) cuando tiene como lema: "Las escuelas deben enseñar a sus alumnos a ser ciudadanos de una sociedad fuerte, capaz, independiente y libre". Stallman y la fundación, predican como principales razones por las cuales las universidades (y con justa razón se incluyen las ingenierías y el tema de programación de computadores) deben utilizar exclusivamente software libre (GNU Operating Sistems, s/f). En estos términos, a continuación se retoman los planteamientos teóricos de Stallman y su equipo (1999):

> ***Compartir:*** Las escuelas deberían enseñar el valor de compartir dando el ejemplo. El software de código abierto, por ser asequible beneficia a la educación, en términos de apoyar la transmisión de información, y ser motivante indiscutible de la comunicación:
>
> ***Conocimiento.*** Es creciente el número de alumnos que poseen tienen talento para programar, pero sucede que con el software privativo la información deseada es cubierta a sus ojos, al punto que a sus docentes se les impide facilitarles su conocimiento a sus discípulos. Cosa contraria sucede frente al software libre, donde el profesor logra exponer el tema básico y luego conceder el código fuente a los estudiantes quienes logran leerlo y con ello, aprender (Stallman R. , Sas Libre, s/f).
>
> *Herramientas.* Es destacable la característica concedida al software libre, al ser tomado como herramienta. Pues aquella condición en la que este no solo permite copiar sino que también se incentiva a los profesores para que pueden entregar a los alumnos copias de los programas que se usan en clase para que los utilicen también en casa, es supremamente beneficioso como herramienta de aprendizaje. (GNU Operating Sistems, s/f)
>
> ***Responsabilidad social:*** Las TIC se han convertido en un elemento de la cotidianidad, y en plena era digital se descubren acelerados cambios en la sociedad y, desde luego, en las escuelas. Es el en aula que se despliega la marca vital de una sociedad. Por ello, se habla de que el papel de las instituciones educativas es formar en los estudiantes capacidades para que logren ser más participativos en medio de una sociedad digital libre, a través del fortalecimiento precisamente de las aptitudes. (GNU Operating Sistems, s/f)
>
> ***Independencia:*** Cada institución encargada de la educación de una comunidad, debe asumir de manera ética su responsabilidad de mostrar las bondades, no la subordinación de un solo producto o de una poderosa empresa exclusiva. Al elegir un software no privativo, la misma entidad escolar gana libertad frente al interés comercial y evade eternizarse ante un único proveedor. (GNU Operating Sistems, s/f)

- Las compañías de software propietario utilizan las escuelas y universidades como trampolines para llegar a los usuarios y así poder imponer su software a toda la sociedad. Ofrecen descuentos y hasta copias gratuitas de sus programas propietarios a las instituciones educativas y de esa manera los alumnos aprenden a utilizarlos y terminan por depender de ellos. Una vez que esos estudiantes se hayan graduado, a ninguno de ellos ni a sus futuros empleadores se les ofrecerán copias con descuentos. En otras palabras, lo que hacen estas empresas es reclutar escuelas y universidades para transformarlas en agentes comerciales cuya función es dirigir a las personas hacia una larga dependencia de por vida.

- Las licencias de software libre no expiran, lo cual significa que una vez que ha sido adoptado, las instituciones conservan su independencia del vendedor (Stallman R. , Sas Libre, s/f). Más aún, las licencias de software libre garantizan a los usuarios el derecho no sólo de utilizar el software como deseen, copiarlo y distribuirlo, sino también de modificarlo con el fin de adaptarlo a sus necesidades personales. Por lo tanto, en caso de que las instituciones necesiten implementar una función en particular en el software que han elegido, pueden contratar a cualquier desarrollador para que realice la tarea, independientemente del vendedor original.

Aprendizaje: En la búsqueda de una institución para estudiar, son cada vez más los estudiantes que toman en cuenta si la universidad enseña ciencias de la computación y desarrollo de software utilizando software libre. El motivo es que con el software libre los estudiantes tienen la libertad de examinar cómo funcionan los programas y de aprender cómo adaptarlos si fuera necesario. Con el software libre se aprende también la ética del desarrollo de software y la práctica profesional.

Ahorro: Esta es una ventaja obvia que percibirán inmediatamente muchos administradores de instituciones educativas, pero se trata de un beneficio marginal. El punto principal de este aspecto es que, por estar autorizadas a distribuir copias de los programas de manera económica o simplemente gratuita, las entidades escolares tienen la opción de ayudar a las familias que sostienen dificultades financieras; con ello, se está suscitando el tema de la equidad y la correspondencia de oportunidades de aprendizaje entre los educandos.

Calidad: Este planteamiento asegura que el software libre brinda estabilidad, seguridad y se caracteriza además, por ser fácilmente instalable, permitiéndose así, ser de bastante utilidad en la educación. La GNU y Stallman han promulgado ventajas de este tipo, junto al excelente desempeño como un punto a favor, sin dejar de reconocerle que su principal importancia radica en la libertad para los usuarios de computadores.

3.3 ESTADO DEL ARTE

Investigar qué herramientas innovadoras en el campo del software libre, pueden favorecer el proceso de enseñanza aprendizaje de la asignatura Programación de Computadores en los pregrados de Ingeniería, llevó a auscultar los estudios que al

respecto existan en el universo académico y científico, para así tener un punto de referencia con el que se abordara el tema en el contexto internacional y nacional. Los expertos recomiendan (Buendía, Colás, & Hernández, 1998) que antes de planificar y ejecutar un trabajo, se debe revisar lo que otros autores han escrito o investigado sobre el mismo asunto.

Diversos autores con investigaciones especializadas, ofrecen una perspectiva de interpretación y estudios de casos específicos que permiten tener un referente de base teórica importante para la investigación, encontrando que sí hay propuestas y acciones que brindan una voz de aliento al presente estudio, y corroborándole que aún hay mucho por hacer en el entorno del software libre.

3.3.1 Enseñanza de la programación de computadores en Colombia y el mundo. Para este punto, se destaca el caso de la enseñanza de programación de computadores en la Universidad Politécnico Grancolombiano (Malaver, Rey, & Rodríguez, 2011), la cual registra que el aprendizaje de la programación es un área de la enseñanza en la que no se aprecia un consenso real. Existen diferentes enfoques y posibilidades al momento de enfrentar el proceso de transmisión del conocimiento, y dichos enfoques generan, por definición, diferentes resultados.

En el escenario de formación del Politécnico Grancolombiano pueden encontrarse dos asignaturas que buscan ofrecer al estudiante las herramientas básicas para enfrentarse a problemas de programación de computadores: Pensamiento Algorítmico y Programación de Computadores. La asignatura Pensamiento Algorítmico es una adición más o menos reciente al pensum que busca introducir de manera informal conceptos asociados con estrategias de resolución de problemas y con la utilización de conceptos matemáticos, geométricos, algebraicos y lógicos en dichas estrategias. La metodología utilizada en las clases se apoya muy fuertemente en la resolución de problemas de tipo lógico – matemático, en primera instancia, mediante una aproximación intuitiva por parte de los estudiantes, para luego mostrarles patrones básicos de solución de problemas que siembren las semillas de métodos más formales como “dividir y vencer”, o que utilicen conceptos más rigurosos como la inducción.

Luego de aprobar esta asignatura, los estudiantes pasan a cursar Programación de Computadores, donde tienen su primer encuentro con los elementos formales del proceso de construcción de algoritmos y su posterior implementación en un lenguaje de programación. En esa asignatura se ofrecen las bases conceptuales necesarias para la programación de computadores, y se dispone de los espacios prácticos para poner a prueba dichas bases. En términos de paradigma y alcance, la asignatura promueve el estudio de la programación estructurada, por cuanto este enfoque muestra los

componentes fundamentales para la implementación de un algoritmo sin complicar el proceso más allá de lo necesario para esta etapa básica de formación.

Como lenguaje de programación, actualmente se utiliza Java para ofrecer a los estudiantes el contacto con uno de los lenguajes con más penetración en el mercado laboral en la actualidad, al mismo tiempo que tienen contacto con un lenguaje abierto, con una gran comunidad de usuarios y un vasto conjunto de recursos de aprendizaje fuera del aula. Como metodología, se cuenta con tres sesiones semanales, una de las cuales está dedicada a la introducción teórica de conceptos nuevos ilustrados por ejemplos y talleres elaborados con apoyo del docente, mientras que las otras dos están enfocadas en trabajo práctico autónomo por parte de los estudiantes con asistencia del docente cuando el estudiante encuentra dificultades.

A continuación se relacionan las dificultades más relevantes que se han encontrado al momento de articular el proceso de enseñanza – aprendizaje con los estudiantes:

• A pesar de la inclusión de Pensamiento Algorítmico, los estudiantes llegan a la asignatura Programación de Computadores con deficiencias conceptuales en términos de la aplicación de conceptos matemáticos adquiridos previamente en su educación media. Se aprecian deficiencias para identificar dichos conocimientos como herramientas válidas de solución a problemas prácticos, así como para aplicarlos de manera correcta una vez son identificados como una opción teórica de solución.

• La decisión de utilizar Java como lenguaje de programación para la asignatura ha tenido resultados mixtos. Mientras por un lado es evidente la ventaja que presenta el tener contacto temprano con un lenguaje que está siendo utilizado actualmente en la mayor parte de los escenarios productivos, y que es fruto de un proceso de maduración riguroso, también resulta evidente que las herramientas de desarrollo e incluso los conceptos mismos inherentes al lenguaje resultan en ocasiones ser contraproducentes al tratarse de una asignatura básica con estudiantes de primeros semestres. Concretamente, el IDE2 utilizado en las clases (actualmente Eclipse, aunque se han hecho pruebas utilizando NetBeans, y BlueJ) puede ser intimidante para los alumnos, y el hecho de enseñar programación estructurada utilizando un lenguaje orientado por objetos genera la necesidad de ignorar ciertos elementos y características del código (clases, constructores, etc.), lo que no resulta ideal.

• A pesar de que las herramientas utilizadas ofrecen opciones como la ejecución paso a paso del código, este proceso no resulta intuitivo para los alumnos. Esto resulta infortunado por cuanto una de las dificultades más notorias es la de “visualizar” el funcionamiento de un algoritmo en el tiempo. Para los estudiantes resulta complejo entender cómo una serie estática de instrucciones se comporta en tiempo de ejecución; particularmente las sentencias condicionales y los ciclos resultan confusos en términos

de su interpretación en el tiempo. Las pruebas de escritorio ayudan mucho en este sentido, pero en muchos de los casos no son prácticas en términos de tiempo.

• La sintaxis del lenguaje se convierte en un obstáculo adicional a la complejidad que presenta la conceptualización de la solución. Existen escenarios en los que los alumnos tienen claridad respecto al proceso a seguir para solucionar un problema, pero presentan inconvenientes a la hora de escribir la solución que diseñan en el lenguaje de programación. La apropiación del conjunto de reglas inherentes a un lenguaje de alto nivel añade otro nivel de dificultad al aprendizaje de programación.

A pesar de ser claro que la mayor parte de las dificultades aquí expresadas son naturales y deben ser manejadas y eliminadas a lo largo del proceso de aprendizaje, este proceso resultaría mucho más fluido si fuera posible eliminarlas o reducirlas en un momento temprano de la dinámica enseñanza-aprendizaje.

3.3.2 Uso del Software libre como estrategia educativa. De las experiencias consultadas, se relacionan a continuación, los estudios que atañen al problema descrito, mencionándose las que cumplen con rigor metodológico, lenguaje claro y coherencia investigativa. El autor decide citar inicialmente los documentos internacionales, y luego culmina con aquellos estudios realizados en el territorio colombiano.

Rodríguez Cánovas, Salvador. Universidad de Almería. (Almería, España). Título: "Diseño de un laboratorio virtual para la docencia de la materia de tecnología en educación secundaria" (2011). La investigación se realizó con el objeto de diseñar una serie de prácticas guiadas con simulaciones de mecánica para la asignatura de tecnología; dichas simulaciones se integran con sus correspondientes guiones en una página web, en este caso en la plataforma educativa virtual "Moodle". Las simulaciones han sido desarrolladas con la ayuda de la herramienta de software libre EJS (Easy Java Simulations). Rodríguez midió el grado de facilidad y beneficio que se alcanza con el uso de dichas simulaciones respecto a la enseñanza tradicional. Para este fin realizó una encuesta a un grupo de 28 alumnos que se encontraban cursando la materia de tecnología, concluyendo que los laboratorios virtuales se constituyen como un nuevo recurso didáctico que puede ser integrado junto a otros recursos dentro de plataformas de e-Learning como Moodle.

Otro estudio es el de Cisneros Arocha, Juan Vicente y Cisneros Arocha y Elías Oswaldo, del Instituto Superior Politécnico "José Antonio Echeverría" (Caracas, Venezuela). Ellos adelantaron un estudio en el año 2008 titulado: "Videojuego Educativo para la enseñanza de la Arquitectura del Computador e Introducción a la Programación apoyadas en Software Libre" (Arocha, 2008). Los ponentes se propusieron formular la investigación para el desarrollo de un videojuego educativo en la enseñanza de la Arquitectura del

Computador e Introducción a la Programación apoyada en software libre, luego de observadas las dificultades vivenciadas en la Aldea Universitaria Fray Pedro de Agreda, en la ciudad de Caracas - Venezuela, Municipio Libertador, Parroquia El Valle. Los autores llegaron a la conclusión que el software educativo se ha convertido en una solución viable y asequible como herramienta de apoyo para el aprendizaje. A pesar de la controversia, se afirma que el videojuego es una alternativa para apoyar este proceso y para ampliar el abanico de recursos didácticos. Mediante de la incorporación de los videojuegos se desea que el alumno asuma un rol más activo y participativo, mejorando la motivación y siendo constructor de su aprendizaje.

La Universidad Rey Juan Carlos, Grupo GsyC/LibreSoft, el Equipo CENATIC (Centro Nacional de Referencia de Aplicación de las TIC basadas en Fuentes Abiertas) y el Observatorio Nacional del Software de Fuentes Abiertas (ONSFA) en España, publicaron un informe titulado “Estudio sobre la situación actual del Software de Fuentes Abiertas en las Universidades y Centros I+D españoles” (Cenatic, 2009). Así pues, este documento presenta información estratégica sobre la situación del Software de Fuentes Abiertas en las Universidades españolas, que sin duda permite a cualquier lector interesado en ahondar en investigaciones de este paradigma, apreciar las posibilidades de este tipo de tecnología, así como entender los beneficios que el Software Libre aporta, “no sólo en términos de costes económicos, sino en el desarrollo sostenible del software y la generación de metodologías colaborativas de trabajo, así como conocimientos y capacidades tecnológicas esenciales para la Sociedad de la Información" (Jaque B., 2009, pág. 5). Ellos logran realizar una interesante reflexión analítica a partir de lo observado en aquellas instituciones que desde hace años vienen incorporando con buenos resultados las tecnologías abiertas a los procesos de enseñanza- aprendizaje.

Realizando la respectiva pesquisa en Colombia, se decide mencionar lo logrado por Villalobos Salcedo, Jorge Alberto. Proyecto "CUPI2 – Una solución integral al problema de enseñar y aprender a programar" (Villalobos Salcedo, 2009). Grupo de investigación TICSW en Construcción de Tecnologías de Información de Software, propuesto por investigadores de la Universidad de los Andes. Este proyecto fue el ganador del 10° premio colombiano en informática educativa, categoría investigación, realizado en el año 2009. Las dificultades en la enseñanza/aprendizaje de la programación ha sido un problema que persiste en las casi tres últimas décadas en Colombia, como en el resto del planeta.

A lo largo del tiempo se han propuesto numerosas soluciones sin que ninguna haya resultado realmente efectiva. A los problemas de motivación de los estudiantes se une la falta de un estudio a fondo de las habilidades que deben adquirir, reduciendo muchas veces los cursos a un recorrido de estructuras sintácticas de un lenguaje de programación. El estudio corroboró que es claro que los cursos de programación son fundamentales en la formación de ingenieros.

Con el proyecto Cupi2 se ha logrado que los cursos de programación soporten mejor al resto de cursos del currículo, dando al estudiante una visión global de la problemática de construcción de software y permitiendo obtener resultados que antes se encontraban restringidos a cursos más avanzados. La mortalidad en los cursos ha disminuido hasta en un 50%, a la vez que la motivación de los estudiantes (medida a través de las encuestas de la Universidad) ha aumentado en más del 20%. Los estudiantes terminan los cursos con una actitud mucho más positiva con respecto a lo que quiere decir desarrollar software y con una visión global de todas las oportunidades que allí se pueden encontrar (Villalobos Salcedo, 2009).

La Universidad Tecnológica de Bolívar en la ciudad de Cartagena, viene implementando el Sistema de aprendizaje Virtual Interactivo (SAVIO), como una herramienta base del modelo pedagógico que les ha permitido flexibilizar los procesos de enseñanza-aprendizaje en el sistema presencial de la universidad. SAVIO es una plataforma de apoyo para los cursos presenciales y virtuales, desarrollada en la Universidad Tecnológica de Bolívar (Serrano, 2005). Desde el 2001 inició la investigación y construcción por parte de los estudiantes de la facultad de ingeniería de sistemas. Esto es un ejemplo del aprovechamiento de los recursos que el software libre provee. La idea siempre fue “desarrollar un producto autóctono, que cubriera las necesidades más apremiantes, capaz de ampliarse a gusto propio” (Serrano, 2005).

Ahora bien, deseando continuar con las experiencias en el contexto colombiano, según Jiménez et. al., aunque existan diversos grupos de usuarios de Software Libre en diferentes ciudades del país, estos se han dedicado a estudiar y divulgar la filosofía que promulga este tipo de software, de tal forma que uno de esos grupos se ha enfocado específicamente al área de Software Libre en la Educación, el cual se denomina “Software de Libre Redistribución y Educación en Colombia SLEC (SLEC, 2012)”; sus miembros, han descrito las ventajas que tiene el uso de esta clase de software en los procesos de enseñanza y aprendizaje, pero sus estudios se han limitado a indicar qué instituciones lo utilizan (tanto Sistema Operativo como aplicaciones) y qué tipo de programas usan. La ponencia de Jiménez et al., y la indagación en la red, permite seguir aceptando en parte lo afirmado, en cuanto a que

> el conocimiento del Software Libre en los Establecimientos Educativos es bajo en este momento, en el país solo existen unas pocas instituciones educativas (una de ellas en el municipio de Roldanillo en el departamento del Valle y otra en la ciudad de Bogotá Distrito Capital) que tienen el 100% de sus computadores (tanto en las aulas informáticas como los destinados a tareas administrativas) con el sistema operativo GNU/Linux y con aplicaciones libres para llevar a cabo el control de los procesos de gestión de notas y logros académicos y para actividades pedagógicas de enseñanza y aprendizaje en diversas asignaturas. Por otra parte, algunos pocos

> Establecimientos Educativos conocen de software libre pero lo usan de manera esporádica. En el Gimnasio Fidel Cano y el Gimnasio Norte del Valle se están llevando a cabo estrategias pedagógicas que se relacionen con la filosofía del Software Libre. (Jiménez, 2005)

Se considera tener en cuenta otros apuntes al estado del arte en cuestión y que se recogen en el mundo académico a través de la red de internet, por su intrínseca relación con la acogida del software libre. Son estos:

UNIVATES, en Brasil, es una universidad estatal trabajando exclusivamente sobre plataformas tecnológicas libres (Contreras, 2013). A propósito, se puede recordar cómo desde el gobierno, en especial durante los dos periodos presidenciales Luiz Inácio Lula da Silva, se apoyó el uso del software libre, con el argumento de su lucha contra la 'desigualdad digital'. El software libre, según él, permitió que el país pudiera ahorrarse cientos de miles de dólares en licencias (Mirás, 2012). Otros gobiernos como el de Argentina, imitan el proceder del gobierno brasileño, e incluso del mexicano que le antecedió, dedicando un grupo interinstitucional específico para el estudio del Software Libre en el Estado, existe el Proyecto de Ley Dragan sobre Política de Utilización de Software Libre por el Estado Nacional de Argentina (Zorzoli, 2002). Cuba y Chile también han librado grandes batallas.

Hay legislación pendiente al respecto en países como Uruguay y Costa Rica, donde se han empezado a llevar a cabo proyectos a menor escala. Algunos países, tales como Venezuela, han incluido el Software Libre en su estrategia de gobierno digital, del cual se puede destacar la importancia del Decreto Presidencial Nº 3390 sobre Software Libre, en donde dice: "El Artículo 110 de la Constitución Bolivariana de Venezuela establece como estrategia nacional relativa al conocimiento su reconocimiento como interés público y su importancia fundamental para el desarrollo económico, social y político del país, así como para la seguridad y soberanía nacional." Con base a este artículo surge el Decreto Presidencial nº 3390. Dicho decreto insta a la Administración Pública venezolana a emplear prioritariamente Software Libre desarrollado con Estándares Abiertos, en sus sistemas, proyectos y servicios informáticos (Jiménez, 2005).

3.3.3 Innovación en la enseñanza del software libre. El libro "Software libre para una sociedad libre", escrito por Stallman (2004), se conserva como un tratado de obligatorio antecedente para esta investigación. El fundador de la FSF, defiende la opción de innovar en la enseñanza del software libre, y con él se comprende que cuando se encuentra un estudiante frente a la posibilidad de desarrollar un procesador de texto nuevo, bueno e innovador, de tal forma que pueda incluir algunas ideas nuevas, así deba incluir cientos de viejas ideas, sólo si está en libertad de poder usarlas, estará siendo realmente un procesador de textos innovador. El autor asegura que en vista de que el radio de acción

en términos de creatividad de software es tan grande, el resultado es que no se necesita ningún plan artificial para llegar a la innovación.

Una reciente conferencia de Stallman (Por qué utilizar y enseñar software libre, 2013), recuerda la inminente condición de innovar en la enseñanza, cuando se acepta que hoy son muchos los jóvenes estudiantes que tienen talento para programar, y que están fascinados con las computadoras y ansiosos por aprender cómo funcionan. Con el software privativo esta información es secreta, por lo tanto los profesores no tienen forma de dejarla a disposición de los alumnos. Pero si es software libre, el docente consigue exponer el tema base para luego conceder el código fuente a los estudiantes quienes llegan entonces, a leerlo y aprender de él. En razón de que las licencias de software libre no expiran, en cuanto haya sido acogido, las instituciones sostienen la respectiva independencia frente a su creador. Se recuerda que las licencias de software libre garantizan a los usuarios el derecho no sólo de utilizar el software como deseen, copiarlo y distribuirlo, sino también de modificarlo; en una palabra, garantiza la innovación.

La Universidad de Alicante, en España, expone la investigación de Tur Viñes, et. al (2008), titulada "*Utilización del software libre Moodle en la asignatura creatividad publicitaria I*", corroborando que la vocación innovadora que nutre a la enseñanza fundamentada en el acogimiento del software libre, es la ideal. Los autores reconociendo en el carácter eminentemente práctico de la asignatura Creatividad Publicitaria I, de tercer curso en la Licenciatura de Publicidad y Relaciones Públicas de la Universidad de Alicante, llevaron a cabo un proyecto de investigación docente en red, cuyo resultado fue, en un primer momento, el diseño de la guía docente. Posteriormente, al avanzar en la implementación de dicha guía, se necesitó recurrir a una plataforma virtual que facilitara la gestión de la evaluación continua. Para ello, se recurrió a Moodle, un software libre para la docencia *on line* del cual se han aprovechado sus interesantes prestaciones en el envío de trabajos, evaluación y comunicación no presencial con el alumno.

Por otra parte, una interesante ponencia de Briet (2006), insiste en que el paso definitivo hacia la innovación en la enseñanza lo debe dar el docente; el autor ha insistido en que los cambios tanto sociales, técnicos y científicos, están en las manos de los profesores. En su propuesta convence en cuanto a que es el profesor quien necesariamente tendrá que cambiar su rol, para implementar metodologías innovadoras que aporten a los estudiantes herramientas para integrar los conocimientos nuevos con los previamente adquiridos.

Pasando a nivel latinoamericano, un interesante aporte proviene del artículo que soporta la investigación de González Mariño (2006), llamado "*B-Learning* y Software Libre, una Alternativa para la Innovación de la Educación en el Contexto de las Sociedades del Conocimiento", con el respaldo de la Universidad Autónoma de Tamaulipas, México. El autor en su trabajo describe el concepto de *Blended-learning,* como una alternativa para la innovación de la práctica del profesor universitario, y reitera, tal como los demás

investigadores mencionados aquí, la necesidad de que se implementen paralelamente, programas de formación inicial continua para el profesorado.

La Universidad de Sonora participa en el XXIV Encuentro Nacional de México el pasado año, con el documento de Castillo y González (2012) en el preciso momento en el cual la educación superior ha sido cargada de esperanzas e incertidumbres; lo hacen para describir el contexto en el cual las instituciones han hecho reales esfuerzos por mejorar su infraestructura pero, a la vez, para señalar que no se ha avanzado mucho en el tema de difusión de la innovación dirigida hacia el profesorado. La ponencia de los dos estudiosos en el tema, permite pensar que sus postulados coinciden con lo manifestado por Briet y por González, en cuanto a que al hablar de innovación en la enseñanza del software libre, la tarea y la responsabilidad mayor de reflejar verdaderos resultados en términos de calidad y de creatividad, sigue residiendo en el docente.

3.3.4 Software libre, innovación y programación de computadores. En una institución de educación superior de la ciudad de Mogi Guaçu/SP/Brasil, siendo exactos, en la Facultad Municipal “Professor Franco Montoro”, se propuso la utilización de herramientas computacionales orientadas hacia el desarrollo de contenido 3D, pensando en un marco educativo, por supuesto. Con la intención de utilizar un conjunto de herramientas basadas en software libre, de calidad excelente y con un gran potencial para el desarrollo de las aplicaciones propuestas, seis docentes e investigadores universitarios, de los cuales uno es español, emprenden la investigación. El proyecto permitió que los estudiantes del curso de Graduación en Ciencias Informáticas desarrollaran una serie de animaciones modeladas por ordenador con la utilización de software libre. De allí se previó la creación de nuevas "células" para el desarrollo de animaciones y simulaciones con la participación de los alumnos interesados en este ámbito profesional, así no pertenecieran al área de computación (Vizconde Veraszto, y otros, 2012). Se puede afirmar que este es el antecedente más cercano a la presente investigación, por su interés en interrelacionar software libre, innovación y programación de computadores, aseverando que son diversas las herramientas computacionales que están a nuestra disposición y definen un muy interesante espacio para el desarrollo de aplicaciones educativas.

Por su parte, Graham Atwell (2007) ha puesto de manifiesto un hecho diferencial del software libre en la educación que no se puede dejar de señalar en este estado del arte: su maridaje con la innovación educativa. Las razones son diversas En primer lugar, en los proyectos de software libre el coste inicial es muy bajo: suelen ser personales o de un pequeño grupo de entusiastas. En segundo lugar, se puede “construir” sobre el trabajo de otros proyectos y explorar sus aplicaciones educativas (por ejemplo, integrando herramientas que originalmente no fueron diseñadas con propósito educativo, como blogs y wikis). Si el proyecto cuaja, porque la gente lo encuentra de interés, es fácil abrirlo a la colaboración. Un ejemplo de este proceso es Moodle, una plataforma de enseñanza basada en presupuestos socio-constructivistas del aprendizaje que ha sobrepasado en

funcionalidades e implantación a sus alternativas privativas. Iniciado por una sola persona, Martin Dougiamas, que, descontento por cómo estaba diseñado y funcionaba el software privativo equivalente de su universidad, “se hizo” una plataforma (realmente modesta en sus inicios) para sus clases. Hoy, la comunidad Moodle está formada por decenas de desarrolladores, miles de usuarios, sus instalaciones se cuentan por millares y varios millones de estudiantes y profesores utilizan Moodle en sus clases presenciales, semi-presenciales o a distancia.

Una tercera razón reside en el efecto de unir en una comunidad en pos de un objetivo común a informáticos y especialistas en otros campos. La comunidad Moodle está formada por informáticos profesionales, profesores de informática, educadores de diferentes niveles educativos, especialistas en tecnología educativa y en e-learning, etc. El proceso por el que se proponen, discuten, diseñan, desarrollan, prueban, modifican, vuelven a probar, rediseñan, perfeccionan y adoptan nuevas funcionalidades es un modelo típico de desarrollo de software de código abierto. En el proceso, tanto los programadores como los educadores proponen, argumentan, programan, prueban, critican, etc. y, mientras tanto, aprenden unos de otros.

Según Atwell (2007), la comunidad de usuarios/desarrolladores es, sin duda alguna, lo que ha convertido a Moodle en un sistema puntero desde el punto de vista didáctico y tecnológico, líder mundial en número de instalaciones, que van desde universidades gigantescas (la Open University, por ejemplo, con cerca de 120.000 estudiantes distribuidos por todo el mundo), pasando por numerosas universidades presenciales de tamaño medio o pequeñas (como la del autor), hasta escuelas rurales minúsculas en países de los cinco continentes. Moodle ha sido traducido por voluntarios a más de 70 lenguas, incluyendo algunas sumamente minoritarias, para las que la probabilidad de que una gran empresa de software “localice” y traduzca a su lengua un producto comercial de estas características es exactamente “ninguna.” La razón: no hay dinero a ganar.

Muchos proyectos de código abierto poseen este tipo de comunidades mixtas en las que desarrolladores informáticos y especialistas en el área de aplicación unen sus conocimientos para crear un producto adaptado a las necesidades reales de los usuarios. Estas comunidades sirven como espacios naturales de intercambio de ideas, de debate y reflexión, de formación mutua en el “otro” campo del conocimiento y en el “propio”. Son un lugar excelente para aprender.

3.3.5 Programación de computadores haciendo referencia a software libre. La Agencia Española de Cooperación Internacional, con la participación del Instituto Tecnológico de Costa Rica y de la Universidad de Murcia, en la provincia del mismo nombre, presenta un interesante estudio titulado “Implementación de las TIC en los programas académicos del Instituto Tecnológico de Costa Rica” (2008). La investigación

logra expresar de manera analítica las principales implicaciones del software libre en las Universidades, atendiendo al interés manifestado por el Instituto Tecnológico de Costa Rica por medio del TEC DIGITAL de incorporar herramientas de software libre en la enseñanza. En su informe hacen referencia a las plataformas virtuales para la enseñanza y el aprendizaje, y con ese propósito mencionan a Claroline, describiéndola como un proyecto con licencia Open Source (software libre), basado en un lenguaje de programación PHP, que integra estándares SCORM e IMS y es empleado con fines educativos. Este informe compartido en el mundo académico, se convierte en un innegable antecedente para la presente investigación.

3.3.6 Innovación en educación en ingenierías.

Jordi Adell afirma que

> Una escuela o instituto conectados a la Internet, con el suficiente número de ordenadores para los estudiantes y con un profesorado formado, sería muy pobre, si se limitara a usar los ordenadores como un mero soporte de los materiales de estudio y siguiera con prácticas didácticas tradicionales (Revista Andalucia Educativa .Del software libre al conocimiento libre, 2005).

La capacidad innovadora de las presentaciones con ordenador es bastante baja. En su misma denominación se resume su funcionalidad didáctica: una "pizarra electrónica" no deja de ser una pizarra, un instrumento del profesorado para "mostrar" cosas a sus alumnas y alumnos. Si alguien espera una revolución en cómo aprende el alumnado por sustituir pizarras de tiza por presentaciones "PowerPoint" es fácil que se desilusione". "El software libre puede funcionar como un virus intelectual, que impregne con su filosofía otros ámbitos de interés educativo."

3.3.7 Innovación en programación de Computadores. El caso de la enseñanza de la informática a nivel universitario es especial. En primer lugar, el software libre permite ver y analizar cómo está diseñado y funcionan programas de ordenador de primerísimo nivel. En segundo lugar, algunas de las mejores herramientas software son libres y los estudiantes pueden utilizarlas sin coste alguno. Pero más allá de estudiar y usar software de código abierto, los estudiantes pueden participar activamente en proyectos reales de desarrollo (Shockey, 2005). Los proyectos proporcionan un contexto más amplio que las típicas tareas académicas en pequeño grupo y les permiten comprender las relaciones entre desarrolladores y comunidad de usuarios, practicar habilidades comunicativas, trabajar en equipo con materiales, ideas y líneas de trabajo establecidas, explorar posibilidades y soluciones nuevas, etc. Es decir, los proyectos libres (y la facilidad para contribuir a ellos) proporcionan un contexto real de trabajo y un valioso entorno de programadores profesionales y altamente cualificados.

Otros autores, como Farber (Long, 2009), han sugerido utilizar el proceso de desarrollo de software libre como modelo para diseñar procesos de enseñanza/aprendizaje formales. Es decir, intentar reproducir el modelo de un entorno distribuido de construcción colaborativa de artefactos en el aula presencial. Sin embargo, como principio de dicho modelo, Faber utiliza una serie de consejos de Raymond (1999) a quienes aspiran a desarrollar software de código abierto (Long, 2009), extraídos de su ensayo La catedral y el bazar sobre el desarrollo de Linux y sus propias experiencias como desarrollador.

La correspondencia entre los consejos de Raymond (de los que Faber elije el subconjunto más "aprovechable" pedagógicamente) y los aspectos mínimos necesarios de un modelo educativo es, cuando menos, tenue. Los desarrolladores de software libre, las comunidades que se forman de manera más o menos espontánea alrededor de proyectos de software libre ejemplifican, sin duda alguna, procesos interesantes desde el punto de vista educativo. "Es hora de que las instituciones de educación superior tomen en consideración este importante y nuevo método de producción y aprendizaje seriamente (el código abierto), y adopten muchos de sus métodos" (Titlestad, 2005)

Un caso interesante de Innovación en programación de Computadores, es el de Labra, Fernández, Calvo y Cernuda (2006), expuesto en el artículo titulado "Una Experiencia de aprendizaje basado en proyectos utilizando herramientas colaborativas de desarrollo de software libre". Los investigadores hacen referencia a una experiencia desarrollada en el ámbito de una asignatura optativa sobre programación declarativa, utilizando herramientas colaborativas habituales en el desarrollo de software libre. Allí se creó un proyecto común entre todos los estudiantes, con el objetivo de facilitar un aprendizaje basado en proyectos.

La utilización de técnicas de aprendizaje basadas en proyectos ha supuesto una novedad considerable para los profesores de la asignatura, dicen los autores. Los docentes carecían de formación en este tipo de técnicas y han tenido que improvisar en numerosas ocasiones. Sin embargo, la experiencia puede considerarse positiva; posiblemente los estudiantes hayan aprendido menos cosas de programación declarativa, pero sí parece claro que han aprendido más de otras habilidades como trabajo en equipo, búsqueda de información, uso de herramientas colaborativas, gestión del tiempo, etc. La duda que se plantea es si el equilibrio alcanzado es suficiente. Sería interesante desarrollar un estudio comparativo entre alumnos sometidos a distintos tipos de aprendizaje. (Labra Gayo, Fernández, Calvo Salvador, & Cernuda del Río , 2006)

Respecto al profesor, los investigadores creyeron que era importante plantear si merece la pena el esfuerzo antes de embarcarse en esta aventura. Puede ser reconfortante y en numerosas ocasiones el profesor también aprende cosas, pero puede haber momentos de desánimo en los que apetece volver a un método más tradicional. Incluso la

popularidad del profesor puede verse mermada en este tipo de experiencias, ya que los alumnos comprueban los límites del conocimiento del profesor y pueden llegar a defraudarse cuando ven que el profesor no tiene respuestas para todo. (Labra Gayo, Fernández, Calvo Salvador, & Cernuda del Río , 2006)

3.3.8 Software libre haciendo referencia a la programación de computadores. Adell y Bernabé (2007) escriben un artículo sobre la educación usando software libre para personas involucradas en el proceso, dirigido a docentes en activo de todos los niveles, a gestores educativos y, especialmente, a estudiantes que se están preparando para una profesión relacionada con la educación, para personas o usuario informático normal.

Las autoras se han marcado cuatro objetivos esenciales. El primero es introducir al lector en los conceptos clave del software libre, su definición, su origen y algunas de sus implicaciones. El segundo objetivo es incitar al lector a probar el software libre y a comprobar sus ventajas prácticas sobre el software privativo. El último apartado, las actividades, está dedicado a ello. El tercer objetivo es animar a reflexionar sobre la relación entre los valores que encarna el software libre y los fines de la educación pública. Pretendemos que los estudiantes desarrollen los conocimientos y las capacidades necesarias para integrarse adecuadamente en esta compleja y contradictoria sociedad de la información del siglo XXI, para ser ciudadanos libres, participativos y solidarios, para ser profesionales competentes.

3.4 MARCO NORMATIVO

3.4.1 Enseñanza de la Programación según ACM. La ACM, reconocida como una sociedad científica y educativa pionera en temas de la Computación, en su informe del Grupo de Revisión Provisional, hace las siguientes consideraciones pedagógicas que vienen al tema de la enseñanza de la Programación en Computadores:

> En todas las disciplinas existen enfoques de la enseñanza que inspiran y alientan a los estudiantes. Tal es la inspiración particularmente relevante en los primeros años de un curso en un momento en que los estudiantes a menudo llegan a un acuerdo con las opciones que se enfrentan. En los últimos años, sin embargo, es necesario encontrar un equilibrio entre las dificultades de los alumnos y la construcción de los niveles de confianza y habilidad. Para lograr el éxito de los cursos, se debe hacer hincapié en la importancia de la enseñanza en su carrera profesional, destacando oportunidades y proporcionando ejercicios emocionantes y creativos, así es que se debe retroalimentar constantemente el currículo, mientras se acepta que el estímulo es crucial para influir en los estudiantes.
>
> Incluso en el contexto de la enseñanza del plan de estudios, hay margen para la inclusión de observaciones sobre la aplicabilidad más amplia y la relevancia de las ideas de la computación. El término " pensamiento computacional " se ha acuñado

> para ilustrar este concepto. Tenemos que desarrollar y refinar el concepto de pensamiento computacional, dice la ACM. En algunos casos, esto puede ser un pequeño paso, sin embargo, puede servir en el tiempo para cambiar las percepciones sobre la disciplina y colocarlo en un centro mucho más posicionado, en términos de mayor relevancia, en la educación para la sociedad de hoy. (ACM, 2008)

3.4.2 Enseñanza de la Programación según ABET. Destacando la ABET como esa organización reconocida en los EEUU, que acredita los programas de ingeniería, tecnología, computación y ciencia aplicada de los institutos de educación superior y de las universidades, y que actualmente acredita aproximadamente 2800 programas en más de 550 institutos de educación superior y universidades dentro de los Estados Unidos (Proyecto ABET- ESPOL, 2013), define criterios para ser aplicados a los programas de ingeniería. Con respecto al plan de estudios, la organización manifiesta que se debe preparar a los graduados para aplicar los conocimientos de las matemáticas a través de diferencial y cálculo integral , probabilidad y estadística , química general y cálculo basado en la física, así como la aplicación de conocimientos en cuanto a métodos, materiales, equipo, planificación y programación, entre otros. El informe de la ABET (2012) comparte criterios para explicar los conceptos básicos de la gestión de temas como la economía, los negocios, la contabilidad, la comunicación, el liderazgo, la toma de decisiones y la optimización de los métodos de ingeniería.

3.4.3 Estándar para Software Libre. Para llegar a comprender el entorno del Software libre, se hace indispensable recorrer su marco normativo con respecto a las licencias. Con el marco legal actual la licencia bajo la que se distribuye un programa delimita exactamente los derechos que tienen sobre él sus usuarios, tal como lo manifiesta Zorzoli (2003). Por ejemplo, en la mayoría de los programas propietarios la licencia priva al usuario de los derechos de copia, modificación, préstamo, alquiler, uso en varias máquinas, etc. De hecho, las licencias suelen especificar que la propietaria del programa es la empresa creadora del mismo, la cual simplemente vende derechos restringidos para el uso del programa. En el mundo del software libre, la licencia bajo la que se distribuye un programa también es de gran importancia. Normalmente, las condiciones de las licencias de software libre son el resultado de un compromiso entre varios objetivos hasta cierto punto contrapuestos. Entre ellos, pueden citarse los siguientes:

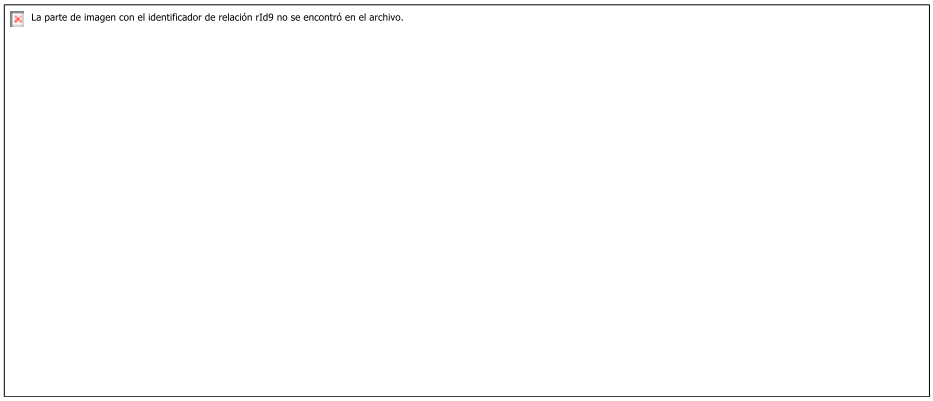
- Garantizar algunas libertades básicas (de redistribución, de modificación, de uso) a los usuarios.
- Asegurar algunas condiciones impuestas por los autores (cita de su nombre en trabajos derivados, etc.).
- Procurar que los trabajos derivados sean también software libre.

Los autores pueden elegir proteger su software con distintas licencias según el grado con que quieran cumplir cada uno de estos objetivos, y los detalles que quieran asegurar. De

hecho, el autor de un programa suele elegir con mucho cuidado la licencia bajo la que lo distribuye. Por otro lado, los usuarios y especialmente quienes redistribuyen o modifiquen el software, deben estudiar con cuidado la licencia del mismo (Zorzoli P. , 2003).

En realidad, casi todo el software libre usa alguna de las licencias más habituales (GPL, LGPL, estilo BSD, estilo Netscape). Pero, sin ahondar en ellas, solo se apreciarán en la figura 2 a manera de información, es necesario enfatizar en el concepto de dominio público; pues muchas veces se comete el error conceptual de suponer que el software libre es de dominio público. Esto sucede simplemente porque la idea de software libre o Código Fuente Abierto es confusa para mucha gente. Tanto el software libre como el de Código Fuente Abierto poseen los derechos de autor reservados, y están protegidos por una licencia. Solo que éstas licencias dan a la gente más derechos de los que están acostumbrados a tener.
Un programa de dominio público es aquel al cual el autor ha renunciado sus derechos, lo recuerda Zorzoli (2003) en su documento. No puede decirse que vengan con una licencia; el programa no tiene propietario y existe la posibilidad de usarse como se desee. Es aquél que no está protegido con copyright, es decir, su autor ha renunciado sus derechos o los derechos de autor han expirado, en condiciones legales, los derechos de autor vencen a los 70 años del fallecimiento del mismo, pasando así a ser de dominio público (Ortega, 2013). Cualquiera puede volver a licenciar un programa de dominio público, o remover el nombre del autor y tratarlo como un trabajo propio. Este es el concepto de dominio público.

Figura 2. Relación de licencias contempladas por el Software Libre



Fuente: Zorzoli, 2003.

3.5 MARCO INSTITUCIONAL

La Universidad Antonio Nariño (de aquí en adelante UAN), consiente del compromiso de impulsar la educación, implementa políticas de desarrollo acordes con los sueños, ambiciones y necesidades de formación de la juventud de nuestro país (UAN, 2013). En Neiva funcionan dos sedes: Sede Buganviles, Calle 19 # 42 – 98 Teléfono: (8) 775968 / 770853; Sede Altico, Calle 7 # 13 – 27 Teléfono: (8) 716028 / 719541.

La UAN se posiciona como una institución de educación superior influyente, con altos niveles de calidad, comprometida con el desarrollo y el impulso de la región. En el momento se cuenta con un portafolio de servicios de programas a nivel de pregrado y postgrado en las modalidades presencial, distancia y virtual y un cuerpo de docentes altamente calificados, contribuyendo en la formación del conocimiento de los futuros profesionales. La facultad de Ingenierías, ofrece: Ingeniería Mecánica, Ingeniería Civil, Ingeniería Electrónica y Biomédica, Ingeniería Industrial, Ingeniería Ambiental, Ingeniería de Sistemas, Ingeniería de Telecomunicaciones. (UAN, 2013)

4. DESCRIPCIÓN PROCESO INVESTIGATIVO

4.1 ENFOQUE DE LA INVESTIGACIÓN

Esta investigación tiene la finalidad de proponer e implementar herramientas y métodos del campo del software libre para favorecer la enseñanza y el aprendizaje de la programación de computadores, con orientación a la innovación en cursos de ingeniería; pero para ello, se requiere iniciar por diagnosticar problemas que se presentan en la enseñanza y aprendizaje de la Programación de Computadores con orientación a la innovación en cursos de ingeniería y que brinden oportunidades de uso de software libre, con base en la observación de algún curso existente y la revisión bibliográfica. Por lo tanto, se decide acoger el enfoque cualitativo para el proceso investigativo.

Según Sabino (2002), durante la investigación cualitativa, la naturaleza del caso o fenómeno es el que orienta al investigador para que indague con mayor detenimiento ciertos aspectos del mismo, que valdría la pena aclarar. La idea es sostener un estudio exhaustivo del fenómeno que rodea el tema del Software Libre en el proceso de enseñanza aprendizaje del curso Programación de Computadores ofrecido en los diversos pregrados de la facultad de ingeniería de la Universidad Antonio Nariño, sede Neiva, lo cual le permite al investigador ampliar su mirada sobre los procesos implícitos en el caso.

4.2 TIPO DE INVESTIGACIÓN

Según Mertens (2005), el constructivismo es quizá el paradigma que ha tenido mayor influencia en el enfoque cualitativo, aunque este planteamiento no deja de tener opositores (Salgado Liévano, 2007). Pensar en este tipo de investigación dentro del enfoque definido, es plenamente probable, puesto que el conocimiento es construido socialmente por las personas que participan en la investigación, y además el investigador y los individuos estudiados se involucran en un proceso interactivo (Mertens, 2005, citado por Salgado 2007).

El constructivismo le otorga a la investigación cualitativa los énfasis principales que lo caracterizan: (a) El reconocimiento de que el investigador necesita encuadrar en los estudios, los puntos de vista de los participantes; (b) La necesidad de inquirir cuestiones abiertas; (c) Dado que el contexto cultural es fundamental, los datos deben recolectarse en los lugares donde las personas realizan sus actividades cotidianas; (d) La investigación debe ser útil para mejorar la forma en que viven los individuos; y (e) Más que variables "exactas" lo que se estudia son conceptos, cuya esencia no solamente se captura a través de mediciones. (Hernández R. F., 2006)

4.3 ASPECTOS METODOLÓGICOS

4.3.1 Población, muestra y grupo control. Se realiza un estudio del tipo de caso-control, dentro del curso de Programación de Computadores de la facultad de Ingeniería de la Universidad Antonio Nariño de la ciudad de Neiva, con el fin de proponer e implementar herramientas y métodos del campo del software libre para favorecer el proceso de enseñanza y aprendizaje.

La muestra está compuesta por estudiantes de Ingeniería que cursan la materia Programación de Computadores durante el periodo 2013-2. Esta muestra, se ha decidido clasificarla en dos grupos, para una mejor comprensión. Éstos son:

- Los sujetos pertenecientes a la condición experimental: Aquí se ubican dos grupos de educandos que cursan diversas ingenierías, y son los grupos en los cuales se haría la prueba, considerándose testigos de la implementación o prueba parcial. Lo conforman 22 estudiantes de un grupo o aula [H= 22, M=0], y 21 del otro curso [H= 19, M=2].

- Se ubica dentro de la investigación, a un tercer grupo (grupo control) al que no se le haría ninguna intervención [H= 13, M=14]. La edad promedio de estos estudiantes participantes, es de 19 años.

4.3.2 Instrumento de recolección de información. En la parte inicial de la investigación, mientras la meta consistiera en hacer un acertado diagnóstico frente a la situación problema, se utilizó el cuestionario aplicado en forma personal como método de recolección de información. Este, junto al análisis documental y a la auto-reflexión del docente investigador a la que hacen referencia Rodríguez y otros como técnica "observación participante" (1996, pág. 65), llegan a permitir sintetizar los problemas de enseñanza aprendizaje de la programación de computadores. Los cuestionarios de preguntas abiertas, fueron dirigidos por un lado, a los educandos de diversas ingenierías cuyo único criterio de selección fue que hubieran tomado en semestres anteriores el curso de programación de Computadores, y por otra parte, se realizaron similares guías a otros docentes de la materia.

Sin embargo, al final de la implementación, también se aplica un sencillo cuestionario con el que se obtiene las impresiones de los estudiantes que fueron sometidos a las herramientas de software libre, para ser confrontadas con las respuestas de los estudiantes del grupo control. De esto se continúa hablando en el numeral 4.4.8, 4.4.11 y 4.4.12.

4.4 DESCRIPCIÓN DE LAS ACTIVIDADES

Como actividades sustanciales del proceso investigativo, y con las cuales se logra alcanzar los objetivos y materializar los resultados, y que además responden al cronograma planteado en el anteproyecto, se encuentran las siguientes. Se describen desde las respectivas subsecciones.

4.4.1 Diagnóstico de la enseñanza, aprendizaje e innovación en cursos de Programación de Computadores. Se decide responder al primer objetivo específico, con el levantamiento de un diagnóstico. La actividad tiene como propósito conocer las necesidades reales del proceso educativo frente al curso Programación de Computadores brindado en las diferentes carreras de ingenierías. El diagnóstico es el primer paso para descubrir con claridad cuáles son las principales dificultades e intereses de los estudiantes, así como los requerimientos de los docentes para desarrollar un proceso de enseñanza/aprendizaje apropiado, en un marco de innovación. Las oportunidades que brindan dichas dificultades e intereses, muestran seguramente las posibilidades de uso del software libre.

En estos términos, se inicia por la revisión y posterior síntesis de literatura existente y en línea, sobre los problemas más comunes que rodean a la enseñanza, aprendizaje e innovación de cursos de Programación en Computadores. Luego, se encamina el verdadero diagnóstico; pues el investigador y docente de dicho curso, se propone elaborar su propio juicio personal sobre el desarrollo del curso que imparte, en cuanto a redescubrir la problemática vivenciada en las aulas. Casi simultáneamente, se decide observar más de cerca su propio entorno, y para eso, aplica entrevistas a estudiantes que en semestres anteriores llegaron a tomar la materia, con él o con cualquier otro docente; para apreciar las dos caras de la moneda, entrevista a colegas que tienen o han tenido recientemente en su carga académica, esta asignatura. Dado por sentado la necesidad de esta tarea, se exponen a continuación las subactividades que la conforman:

4.4.1.1 Revisión de literatura selecta y relevante sobre problemas o dificultades para favorecer la enseñanza, aprendizaje e innovación en cursos de Programación de Computadores. La primera sub-actividad a tratar, se cumple con la búsqueda sistemática de los mejores artículos de investigación publicados en la literatura, y que rodean al tema en cuestión. La gran mayoría de ellos tuvieron espacio en el estado del arte de este documento.

4.4.1.2 Síntesis de la revisión realizada. Como una segunda sub-actividad, se propuso la síntesis "orientada a facilitar la mirada reflexiva del investigador", como diría Salgado (2007, pág. 33) en su documento sobre investigación cualitativa. El resultado de este ejercicio, se plantea más adelante, con la síntesis que recoge a toda la actividad de diagnóstico. Sin embargo, se decide plantear la Tabla 1 para sintetizar con exclusividad, lo más reciente y significativo de la pesquisa realizada.

4.4.1.3 Auto-reflexión del investigador como docente de programación de computadores para identificar problemas. La reflexividad es una habilidad humana presente en las interacciones sociales y precisamente por esto se hace presente en la investigación cualitativa (Cuesta Benjumea, 2003). Esta es precisamente la sub-actividad que aunque hace parte de las acciones que llevan al alcance del objetivo específico ubicado como inicial, se sostiene a lo largo de la investigación, y marcó la pauta de la propuesta, por estar inmerso indiscutiblemente en el planteamiento del problema motivador de la misma.

Para iniciar, es indispensable anotar que en este ejercicio de reflexiones hechas como investigador, se plantean ideas concebidas gracias a la práctica profesional docente, entorno a los problemas que descubre en las clases. Y a la par comienza, por supuesto, a visualizar las posibles soluciones. Se plantea, entre otras dificultades, que a los

estudiantes les cuesta identificar fácilmente un problema, definirlo u ordenarlo y luego seguir un procedimiento buscando la solución del mismo. Esto es lamentable, en vista de que esto hace parte de las habilidades con las que se debe contar en el curso, y a la larga un futuro ingeniero. Para no descargar todas las dificultades en una parte de los actores del proceso educativo, se acepta que en el Curso de Programación de Computadores, los programas que propone el docente para ser construidos en clase, no resuelven problemas o necesidades cotidianas que puedan ejecutarse desde sus móviles o portátiles para resolver algún problema o necesidad clara para ellos, desaprovechándose toda la motivación que tienen los jóvenes en estos equipos tecnológicos que, sin lugar a dudas, poseen relación con la materia.

Tabla 1. Síntesis revisión -problemas en aprendizaje e innovación en cursos de Programación de Computadores

Autor	Año	Caso representativo (Institución)	Síntesis
Malaver, Rey, & Rodríguez	2011 (Colombia)	Enseñanza de la programación de computadores en Colombia Universidad Politécnico Grancolombiano	• A pesar de la inclusión de Pensamiento Algorítmico, los estudiantes llegan a la asignatura Programación de Computadores con deficiencias conceptuales en términos de la aplicación de conceptos matemáticos adquiridos previamente en su educación media. • Para los estudiantes resulta complejo entender cómo una serie estática de instrucciones se comporta en tiempo de ejecución; particularmente las sentencias condicionales y los ciclos resultan confusos en términos de su interpretación en el tiempo. Las pruebas de escritorio ayudan mucho en este sentido, pero en muchos de los casos no son prácticas en términos de tiempo. • La apropiación del conjunto de reglas inherentes a un lenguaje de alto nivel añade otro nivel de dificultad al aprendizaje de programación.
Villalobos Salcedo, Jorge Alberto	2009 (Colombia)	Universidad de los Andes	A los problemas de motivación de los estudiantes se une la falta de un estudio a fondo de las habilidades que deben adquirir, reduciendo muchas veces los cursos a un recorrido de estructuras sintácticas de un lenguaje de programación. El estudio corroboró que es claro que los cursos de programación son fundamentales en la formación de ingenieros.
Hermanos Arocha Cisneros	2008 (Venezuela)	Instituto Superior Politécnico "José Antonio Echeverría"	El alumno asume un rol poco activo y mínimamente participativo; su motivación es mínima y se demuestra desinteresado en ser un constructor de su aprendizaje.
Stallman, Richard	2013	Organización FSF	Frente a la apremiante condición de innovar en la enseñanza, se halla la necesidad de reconocer que los jóvenes estudiantes de hoy están más prestos a programar, pues las computadoras hacen parte de su cotidianidad. Sin embargo, con el software privativo esta información es secreta, por lo tanto los profesores no tienen forma de dejarla a disposición de los alumnos.

Fuente: Elaboración propia

4.4.1.4 Entrevistas a otros docentes de Programación de Computadores en busca de problemas. El instrumento permite la identificación de la persona responsable (en este caso el investigador), la fecha de la aplicación, el objetivo del cuestionario, la autorización de la publicación de sus respuestas, el agradecimiento y el instructivo. Se les acercó el cuestionario (Anexo A) de manera personal, a cinco docentes de programación de Computadores; uno de ellos no ofrece el curso en este momento, pero lo dirigió hasta hace poco tiempo.

4.4.1.5 Entrevistas a estudiantes de semestres que ya hayan tomado el curso de Programación de Computadores. Este cuestionario (Anexo B), lo respondieron cinco estudiantes a quienes se les respetó el anonimato, en vista de garantizar la libre expresión de las ideas. El único criterio de selección consistió en que siendo estudiantes de diversas ingenierías, en semestres anteriores hubieran tomado la materia Programación de Computadores. El instrumento es bastante similar al empleado con los docentes, permitió la identificación de la persona responsable (en este caso el investigador), la fecha de la aplicación, el objetivo del cuestionario, el criterio de selección, las preguntas y el agradecimiento.

4.4.1.6 Síntesis de los problemas identificados en entrevistas, auto- reflexión y revisión bibliográfica. Luego de lograr indagar a los participantes reales del curso que se pretende observar (Programación de Computadores), de observar y de reflexionar como investigador participante, y de profundizar en el tema dirigiendo la mirada a la literatura selecta, se integran las acciones para poder apreciar los problemas identificados, a través de una síntesis. En estos términos, la síntesis es como se aprecia a continuación:

Los estudiantes (según los docentes entrevistados), llegan al curso con dificultades para

- *identificar claramente un problema*
- *definir u ordenar y luego seguir un procedimiento para la solución de un problema*

Los profesores contactados para enriquecer este diagnóstico, expresaban también que

- *se acude aún a una enseñanza por imitación.*
- *algunos docentes resultan teniendo éxito en su curso, pero no sistematizan su experiencia para que sea empleada luego por otro profesor, o incluso, por él mismo tiempo después.*

La auto-reflexión plantea, entonces, que

- *Los programas que se construyen no resuelven problemas o necesidades cotidianas que puedan ejecutarse desde sus móviles o portátiles para resolver algún problema o necesidad clara para ellos.*
- *Los estudiantes presentan dificultades para identificar fácilmente un problema, para definir u ordenar y luego seguir un procedimiento buscando la solución del mismo. Si bien estas deficiencias afectan el aprendizaje de otros cursos, es más apremiante en el caso de la Programación de Computadores en la cual estas habilidades son esenciales.*

Los estudiantes afirman en las entrevistas (expresiones textuales) que

- *en ocasiones la explicación no es suficiente*
- *el profesor se dedica a hacer ejercicios en el tablero o pide ver la solución de un problema en un libro*
- *el tema de la programación es bastante amplio, exigiendo que se deba profundizar extra clase*
- *se queda uno con la impresión de que lo que está aprendiendo, poco aplica en la ingeniería que uno cursa.*

La revisión especializada registra que

- Los estudiantes llegan con problemas de motivación. (Villalobos Salcedo, 2009)
- Es común hallar estudiantes insatisfechos con el aprendizaje de la programación por encontrarlo difícil y poco útil para sus estudios y vida profesional. (Bermúdez, 2007)
- Se descuidan muchas veces las habilidades que deben adquirir, reduciendo los cursos a un recorrido de estructuras sintácticas de un lenguaje de programación. (Villalobos Salcedo, 2009).
- El hecho de reescribir los algoritmos hasta ponerlos a punto es operativamente complicada cuando se trabaja con los elementos tradicionales como lápiz y papel, tiza y pizarrón, etc. (Moroni & Señas, 2005)
- Entre un grupo es común apreciar diferencias de conocimientos entre los estudiantes (Moroni & Señas, 2005)
- Generalmente se espera que quienes aprueben el curso lo hagan porque son capaces de realizar programas con un grado de dificultad mayor al que poseen la mayoría de estudiantes. (Martínez, 2005)

Se identifican oportunidades de uso del software libre en

- Proyecto Cupi2. Universidad de los Andes. (Villalobos Salcedo, 2009). Con el proyecto Cupi2 se ha logrado que los cursos de programación soporten

mejor al resto de cursos del currículo, dando al estudiante una visión global de la problemática de construcción de software y permitiendo obtener resultados que antes se encontraban restringidos a cursos más avanzados. La mortalidad en los cursos ha disminuido hasta en un 50%, a la vez que la motivación de los estudiantes (medida a través de las encuestas de la Universidad) ha aumentado en más del 20%. Los estudiantes terminan los cursos con una actitud mucho más positiva con respecto a lo que quiere decir desarrollar software y con una visión global de todas las oportunidades que allí se pueden encontrar. (Villalobos Salcedo, 2009)

- Universidad Politécnico Grancolombiano (Malaver, Rey, & Rodríguez, 2011), donde optaron por dos asignaturas que buscan ofrecer al estudiante las herramientas básicas para enfrentarse a problemas de programación de computadores: Pensamiento Algorítmico y Programación de Computadores. La asignatura Pensamiento Algorítmico busca introducir de manera informal conceptos asociados con estrategias de resolución de problemas y con la utilización de conceptos matemáticos, geométricos, algebraicos y lógicos en dichas estrategias.

- En Carrie Busey, prestigiosa Escuela Primaria ubicada cerca de Kenwood, exactamente en el condado de Champaign, Illinois, Estados Unidos, según se comenta en estos días dentro de un blog, la "aplicación Blockly" (Blockly es el entorno subyacente), se ha usado en *Hour of Code*. " La Hora del Código". La Hora del Código es promovido por algunas personalidades como (Bill Gates y Mark Zuckerberg, sólo por citar algunas), para enseñar a los niños a programar.

  El autor de un comentario asegura que se trata de un espacio "muy tolerante, muy suave, muy 'web 2.0' y muy instructivo. Se comienza con una tarea muy sencilla, con un objetivo muy claro, con sólo unas pocas opciones para completar esa tarea. Y que sólo construye a partir de ahí. Antes de que usted lo sepa, usted está utilizando bucles y ramificaciones lógicas. (Dschultz, 2013)

  Tanto Kenwood como Carrie Busey han estado explorando cómo utilizar un programa llamado *eToys* en su plan de estudios. Allí es apreciable cómo e Toys está construido sobre una plataforma denominada Squeak. MIT salió con un concepto llamado Scratch muy similar, también construido en la plataforma de Squeak. Donde *eToys* es de composición abierta, robusta y muy completa, Scratch está más centrado, más fácil de usar y enseñar. Se considera a *eToys* lo suficientemente poderoso y grande, Scratch, por su parte, es simple. La Hora del Código utiliza algo llamado Blockly, similar a Squeak, y se ve casi igual a Scratch. Está escrito completamente en JavaScript (a diferencia de Squeak y eToys/Scratch), y los autores lo han

puesto a disposición para que pueda descargarlo a una mera memoria USB y ejecutarlo desde un navegador web moderno.

- Existen también testimonios en línea, sobre la experiencia con Blockly. Hacen referencia a las primeras 20 lecciones de la fase de introducción y a sus 19 etapas restantes. Se comenta sobre sus definitivas ventajas, con respecto al estudiante. Fuera de los bucles de control, ramas de lógica, también hay funciones y parámetros. Pero estos están ocultos, o más bien abstraídos detrás de tareas divertidas que son cortas, rápidas y fácil de digerir, por lo tanto, el aprendizaje está inmerso, casi de trasfondo, dando prioridad al juego y a la exploración creativa, dando espacio a quienes solo quieren probar cosas.

  El comentario de un docente recalca que con el acogimiento de Blockly, descubre que estudiantes tienen diferentes puntos fuertes. Su comentario lo aprovecha para resaltar que algunas personas realmente inteligentes han logrado de manera divertida, abrir las puertas y permitir que cualquiera pueda aprender conceptos en los que se ha trabajado antes. Para algunos, esto podría abrir la puerta a una futura carrera, mientras que para otros, esto puede encender una pasión por los ordenadores.

- UMSA (Universidad Mayor de San Andrés) en La Paz – Bolivia, (Asturizaga, 2011) en su comentario titulado "... versátil y dinámico...", se hace alusión a una de las tantas experiencias empleando Chamilo. El médico que participa en la página de experiencias, es Jefe del Departamento Materno-infantil de la Universidad Mayor de San Andrés, en la capital boliviana, y expresa textualmente la manera como Chamilo fue una grata experiencia en sus aulas universitarias. Relata que la herramienta “les sirvió de apoyo a las actividades presenciales y posibilitó la construcción de conocimiento con el aporte de todos”. Chamilo, según el profesor Asturigaza, brinda la posibilidad de una “autoevaluación permanente, lo que supone una gran ayuda en un proceso de enseñanza-aprendizaje. En resumidas cuentas, Chamilo ha resultado ser un software muy versátil y dinámico”.

- Por otra parte, Montserrat Martínez (2011), como responsable de la plataforma de formación en línea dentro del Hospital Universitario *Vall D´Hebron* en Barcelona -España, destaca de Chamilo LMS "... su manejabilidad, sus alcances tecnológicos y su adaptabilidad al entorno ...". El docente asevera que Chamilo principalmente es un software valioso por “su gran facilidad de uso, su manejabilidad, sus prestaciones tecnológicas, y su adaptabilidad a nuestro entorno sanitario”. Insiste en algo supremamente valioso en él, y que es la utilización de código libre.
  Tecnológicamente se adapta perfectamente a nuestras necesidades formativas, queríamos acercar el conocimiento al mayor número de profesionales, nos es de gran utilidad la planificación de itinerarios de

> aprendizaje, las evaluaciones en línea, la facilidad en tratar los contenidos multimedia, enlaces etc. En estos momentos queremos potenciar las herramientas de comunicación que están disponibles como el chat, foros, mensajes personales, Wiki. Un punto importante que nos ha ofrecido viabilidad y seguridad en el proyecto ha sido el soporte técnico y pedagógico que nos ha facilitado Contidos Dixitais (proveedor oficial de Chamilo), con una respuesta inmediata a una institución tan compleja y variable como la nuestra, con más de 7.000 profesionales. (Montserrat Martínez, 2011)

4.4.1.7 Diligenciamiento de un cuadro diagnóstico que resuma, describa los problemas e identifica oportunidades de uso del software libre. El cuadro contiene cuatro columnas: Identificación del problema, Descripción del problema, Fuente (lugar o momento donde fue observado o expresado) y la columna dedicada a las oportunidades de uso del SL (Cuadro 1). En vista de que el cuadro representa el primer resultado de la investigación, se invita al lector a remitirse al capítulo Cinco (numeral 5.1), donde se halla expuesto.

4.4.1.8 Identificación de herramientas o recursos del software libre y maneras o métodos en que se han usado en la enseñanza y aprendizaje de Programación de Computadores. En esta sub-actividad, se logran reconocer diversas herramientas acogidas para enseñar a programar. En este punto sólo se mencionarán, y se describirán brevemente, puesto que es en la actividad siguiente, cuando se responde al objetivo segundo, que se compararán y se sintetizarán en un cuadro que enriquece a su vez, los resultados de la presente investigación. Las herramientas identificadas, son las siguientes:

- **Waterbear** (http://waterbearlang.com/)**.** Waterbear es un conjunto de herramientas para la creación de lenguajes de programación de arrastrar y soltar, con algunos ejemplos de lenguajes con los que se puede jugar y aprender. El objetivo es hacer más fácil todo para envolver otros lenguajes existentes. Tener un lenguaje visual significa que usted no tiene que centrarse en el aprendizaje de una sintaxis para iniciar la programación. Waterbear es bueno para los niños, artistas y cualquier persona que le gustaría que su computadora haga algo nuevo sin tener que convertirse en un "programador" (aunque podría conducir a eso). Sin embargo, no se encuentran casos de aplicación de Waterbear publicados en la web.

- **Scratch** (http://scratch.mit.edu/). Con Scratch se puede programar historias interactivas, juegos y animaciones y compartir creaciones con otros en la comunidad en línea. Scratch ayuda a los jóvenes a aprender a pensar creativamente, razonar sistemáticamente, y trabajar colaborativamente — habilidades esenciales para la vida en el siglo XXI. Scratch es un proyecto del Grupo Lifelong Kindergarten del Laboratorio de Medios del MIT. Se ofrece

de forma gratuita. Está diseñado teniendo en cuenta el aprendizaje y la educación. Un gran número de educadores ha brindado apoyo a los creadores de Scratch desde el 2007, tanto en ambientes formales como informales de aprendizaje; profesores de K-12, investigadores en educación y ciencias de la computación, bibliotecarios, coordinadores académicos de museos, y padres.

- **Blockly (**https://code.google.com/p/blockly/)**.** Blockly es un editor de programación gráfica basada en web. Los usuarios pueden arrastrar y unir bloques para construir una aplicación. No se necesita escribir. Se ejecuta en un navegador web, es decir, sin necesidad de descargar o instalar *plugins.*

- **App Inventor** (http://appinventor.mit.edu/). App Inventor le permite desarrollar aplicaciones para los teléfonos móviles con *Android* mediante un explorador Web y un teléfono o un emulador conectado. Los servidores de App Inventor almacena su trabajo y le ayudan a mantener un seguimiento de sus proyectos.

- **Rebeca a través del espejo** (http://www.gmrv.es/rebeca-es/proyectorebeca.html). Hace 20 años un típico programa que hiciera salir por pantalla una frase como “Hola Mundo” llenaba de emoción a los estudiantes en sus primeros pasos en la programación. Hoy día, los jóvenes están inmersos en la tecnología, no sólo ordenadores, sino todo tipo de aparatos electrónicos programables están a su alcance: móviles, reproductores de música, consolas de videojuegos, etc. Y, curiosamente, se da la paradoja de que en esta era de la información, en la que las habilidades de programación son fundamentales para el ejercicio de la profesión de muchas titulaciones y que se tiene fácil acceso a la tecnología, los jóvenes se muestran cada vez más reacios a explorar las posibilidades de las máquinas y descubrir cómo están hechas y cómo sacarles mejor partido. Nuestro entorno ha cambiado, y los estímulos que activa, sin perder rigor, mediante un lenguaje más visual y cercano a la realidad que viven y desean crear. Se han hecho tímidos intentos en lenguajes como BASIC, pero las limitaciones de estas tecnologías y la frialdad de la línea de comandos y el editor de texto para el código, en lugar de estimular a los alumnos, en muchos casos ha hecho huir de estas herramientas a estudiantes que tenían un potencial extraordinario para resolver problemas abstractos. Alumnos que de haber conocido la informática con otra aproximación, hubieran podido descubrir su vocación en este campo.

Ahora bien, se pueden identificar otras herramientas sugeridas para aprender la teoría brindada por el curso, llegar a pensar en la retroalimentación de lo visto en clase, y proponer la forma de evaluar a los estudiantes. Estas son:

- Chamilo

- Claroline
- Dokeos
- Moodle
- Edx platform

- **Chamilo** (https://campus.chamilo.org/). Chamilo es, tal vez, la más ambiciosa iniciativa del momento, en plataformas de *E-learning,* dentro del mundo del software libre. Funciona con la plataforma LAMP (Linux, Apache, MySQL, PHP), pero también con Windows y Mac. Su instalación es fácil, así como la creación de contenidos. Con una interfaz clara, permite el seguimiento a los resultados de los usuarios, y sostiene la comunicación sincrónica y asincrónica. Ofrece hasta nueve perfiles de usuarios predefinidos, e incluye hasta 20 herramientas pedagógicas entre las que se tienen al chat, al foro, documentos compartidos, wiki, agenda, asistencia, encuestas, entre otras). Chamilo está dirigida por una fundación sin ánimo de lucro que opera en Bélgica desde el 2010 y apoyada por más de cinco compañías privadas, con un objetivo claro de construir una comunidad de docentes, pero también de impulsar la formación de empresa.

- **Claroline** (http://www.claroline.net) es un software de código abierto, disponible en varios idiomas. Se puede descargar gratuitamente e instalar libremente. Se basa en un modelo educativo flexible en donde la información se convierte en conocimiento a través de las actividades y producciones de los alumnos, en un sistema impulsado por la motivación y la interacción. La amplia gama de herramientas a disposición de los usuarios permite a cualquier profesor o estudiante establecer u operar un dispositivo educativo para el aprendizaje. Los instrumentos genéricos (calendario, documentos, foros,…) pueden utilizarse en la plataforma en diferentes contextos.

  Claroline es una plataforma estable, abierto a todos, lo que permite un fácil uso del espacio para la formación y la colaboración. Su funcionamiento no requiere conocimientos técnicos especiales. Fácil de instalar y también fácil de usar: sólo requiere un navegador para gestionar las diferentes áreas, y contar con los usuarios registrados. El docente puede formular sus proyectos y almacenarlos allí, también puedo hacer además sus planes de estudios, y programar sus clases. Esta propuesta se basa en una comunidad global de usuarios y desarrolladores. La plataforma es así en más de 100 países de todo el mundo. Fue iniciada por *UCLouvain* (Bélgica) en 2001, y como proyecto está dirigido por el Consorcio Claroline que reúne a fundaciones de varios países dentro de una organización internacional sin ánimo de lucro.

- **Dokeos.** Con un entorno de *E-learning,* Dokoes se convierte en una llamativa aplicación de administración de contenidos de cursos y también en una destacada herramienta de colaboración. Es software libre y se encuentra

bajo la licencia GNU GPL. También está certificado por la OSI y puede ser usada como un sistema de gestión de contenido (CMS) por docentes e instituciones educativas. Contiene calendario, proceso de entrenamiento, chat en texto, audio y video, y administración de pruebas. Posee en su documentación, el historial de versiones con consejos técnicos de instalación, así como los errores encontrados en el sistema y sus soluciones. La discusiones de este grupo se llevan a cabo a través de un foro y sintetizados en los textos por Wiki. Hasta el 2007, se hallaba en 34 idiomas (y varios están completos) y es empleada (a septiembre de 2010) por 9900 organizaciones, según reporta el mismo sitio web de la empresa.

- **Moodle[2].** Moodle es un sistema de código abierto para la gestión de cursos (CMS), también conocido como Sistema de Gestión de Aprendizaje (LMS) o un Entorno Virtual de Aprendizaje (VLE). Se ha vuelto muy popular entre los educadores de todo el mundo como una herramienta para crear sitios web dinámicos en línea para sus estudiantes. Para que funcione, puede ser instalado en cualquier servidor web, ya sea en uno de sus propios ordenadores o de una en una empresa de alojamiento web.

  El objetivo del proyecto Moodle es siempre en dar a los educadores las mejores herramientas para gestionar y promover el aprendizaje, pero hay muchas formas de usar Moodle: Moodle tiene características que le permiten escalar a grandes implementaciones de cientos de miles de estudiantes; sin embargo, también se puede utilizar para una escuela primaria o educación para aficionados. Muchas instituciones lo utilizan como su plataforma para realizar cursos totalmente en línea, mientras que algunos lo usan simplemente para aumentar cursos cara a cara (conocida como aprendizaje combinado).

  Muchos de sus usuarios les encanta usar los módulos de actividades (tales como foros, bases de datos y wikis) para construir comunidades ricamente colaborativas de aprendizaje en torno a su temática (en la tradición social constructivista), mientras que otros prefieren utilizar Moodle como una forma de entregar el contenido a los estudiantes (como paquetes SCORM estándar) y evaluar el aprendizaje mediante tareas o cuestionarios.

- **Edx platform.** Proyecto de código abierto de Edx - Open Edx – que promete ser una plataforma de aprendizaje en línea de última generación para llevar educación de calidad a estudiantes de todo el mundo, liderado por la empresa EDX sin ánimo de lucro, y compuesta por 29 instituciones con desarrolladores a nivel global. Fue fundada en el 2012. Edx se viene comprometiendo con las oportunidades brindadas por el código abierto para la expansión de la educación tanto en línea como en el campus.

[2] Extraído y traducido de la página oficial https://moodle.org/about/

4.4.2 Comparación de métodos o herramientas identificados considerando el diagnóstico realizado. Se logra destacar, en el transcurso de la investigación, y más precisamente en esta etapa de aprovechamiento del diagnóstico, que las herramientas o métodos identificados en el proceso de enseñanza aprendizaje van desde herramientas libres usadas habitualmente como gestor de asignaturas o cursos (el caso de Moodle), hasta llegar a aplicaciones como Scratch en código abierto. Para llegar a lograr dicha comparación, se realiza previamente una pesquisa de la literatura científica o en línea sobre herramientas o recursos del software libre y las maneras o métodos como se han usado para la educación en Programación de Computadores. Esta pesquisa se logra sintetizar hasta llevarla a un cuadro que la exponga de manera precisa y cercana a lo que en realidad se acoge (Cuadro 2).

4.4.2.1 Revisión de la literatura científica o en línea sobre herramientas o recursos del software libre y maneras o métodos como se han usado para la educación en Programación de Computadores. Zapata Puerta y Recaman Chaux, docentes de la Facultad de Ingenierías - Politécnico Colombiano Jaime Isaza Cadavid, llevaron a cabo un estudio titulado 'Sistema tutorial interactivo para la enseñanza y aprendizaje de algoritmos- EvaProg'. (2011). Los autores promueven una herramienta como apoyo al proceso de enseñanza aprendizaje en cuanto a algoritmos, con metodología activa, acogiendo ejemplos que les permiten a los estudiantes desarrollar habilidades mientras crean ambientes significativos de aprendizaje. Ellos sugieren a EvaProg a la hora de querer promover el aprendizaje autónomo, induciendo al estudiante al trabajo extra-clase. Se da cabida a la autoevaluación de contenidos, y a la solución de problemas y su codificación en Java/C++. EvaProg le permite al profesor dedicar más tiempo al diseño estratégico y metodológico de las clases, y a acompañar el aprendizaje del estudiante, además, es una ideal opción cuando se trata de innovar frente a la metodología tradicional. (Zapata Puerta & Recaman Chaux, 2011)

Lo interesante aquí, es poder apreciar lo acertado del aporte de Zapata y Recaman, en cuanto a la revisión de métodos pedagógicos innovadores para la enseñanza de la Programación que hace al interior de su publicación. Los autores describen a través de una tabla, diversas herramientas que facilitan el aprendizaje de algoritmos, dividiéndolas en Sistemas con entornos de desarrollo incorporado, Sistemas con entornos basados en ejemplos, Sistemas con entornos basados en visualización y animación, y Sistemas con entornos basados en simulación. (2011, págs. 9-11)

Por su parte, Gómez-Albarrán (2005) es posiblemente quien más haya avanzado en el tema. Con su estudio, financiado por el Ministerio de Ciencia y Tecnología de

España, logra una interesante revisión de las tendencias últimas, en cuanto a métodos de enseñanza empleados en cursos de programación. La investigadora referencia diversos trabajos amanera de ejemplo, y a la vez proporciona las direcciones web en las que se obtienen numerosas herramientas. En el trabajo de Gómez-Albarrán se mencionan tutores virtuales como PROUST y Lisp, pasando por The Programmer's Apprentice, hasta llegar a sistemas menos complejos, en razón de que

> los trabajos iniciales en el campo de la enseñanza de la programación fueron intentos altamente sofisticados, con un elevado componente de conocimiento. Sin embargo, los resultados fueron algo decepcionantes. (2005, pág. 363)

En estos términos, aparece la inclusión de sistemas de enseñanza de la programación que "incluyen un reducido entorno de desarrollo en el que los alumnos pueden escribir código" (Gómez-Albarrán, 2005, pág. 364). Se menciona, para ser más precisos, como entorno de enseñanza al sistema BlueJ (www.bluej.org), y herramientas basadas en web que le facilitan al estudiante la interacción al punto de permitirle el aprendizaje con la exploración de "ejemplos de programas autoexplicativos", entre ellos, el WebEx (2005, pág. 364).

Una gran ventaja de WebEx, según Gómez (2005), es que respeta el ritmo propio de aprendizaje en cada alumno, pues la herramienta expone por niveles los detalles de cada ejemplo; además, otra gran ventaja de WebEx consiste en que cada acción del usuario queda registrada, de tal forma que al profesor de programación se le facilita recorrer el historial de la actividad de cada uno de sus pupilos. Entornos basados en la visualización y animación de algoritmos como los apreciados en http: www.cs.hope.edu./alganim/ccaa/index.html, son traídos a ejemplo por la autora, para confirmar la existencia de gran cantidad de sitios dispuestos para que el estudiante de programación, logre familiarizarse con los algoritmos. (Gómez-Albarrán, 2005)

Por otro lado, se alude a sistemas empleados también para la educación en Programación de Computadores como ANIMAL, LEONARDO, JHAVÉ, XTANGO y POLGA. Gómez-Albarrán no pierde oportunidad para afirmar que aunque existan estudios que brindan buenas expectativas frente al empleo de estas herramientas, sí se hacía conveniente que a los estudiantes se les mejoraran las condiciones pedagógicas, en cuanto a darles la posibilidad de un sistema donde puedan retroceder (dar "vuelta atrás", para decirlo con la investigadora), ante momentos de confusión enfrentadas por el estudiante ante la representación esquemática. (2005)

Otra opción que representó novedad en su momento, giró en torno a la simulación: Aparece Karel, The Robot, utilizando un lenguaje tipo Pascal (http://www.sourceforge.net) y sus versiones posteriores hasta incluir a Java y a Lisp. Con estas herramientas los estudiantes son llevados a comprender más fácilmente cómo funcionan los programas (Gómez-Albarrán, 2005). También se debe hacer alusión a Alice como un interesante entorno de animación, pero esta vez en 3-D (http://www.alice.org).

4.4.2.2 Síntesis de la revisión bibliográfica realizada sobre herramientas y métodos. La revisión permite destacar para esta etapa de la investigación, que Blockly es la herramienta que logra atender problemas de tipo académico, problemas detectados en el diagnóstico y que hacen parte del proceso de enseñanza. Esta herramienta permite llevarse a clase de Programación de Computadores para socializarla con los estudiantes, y luego proponer ejercicios, evaluar resultados y con base a los resultados, proponer más ejercicios de tal manera que la dificultad se va haciendo más compleja.

Cuando ya se haya progresado en un 50% aproximadamente de la asignatura, se hace factible mostrar al estudiante el código que puede generar Blockly a otros lenguajes como JavaScript o Python. Mostrar las estructuras similares, y de esta manera incentivar los estudiantes a escribir código directamente. Mientras tanto, Chamilo LMS, se destaca como una excelente opción a la hora de atender problemas de tipo académico y problemas en el proceso de enseñanza aprendizaje.

La revisión bibliográfica la hace factible para crear espacios destinados a entregar la teoría de todo el curso, organizado para hacer entregas semanales conforme avanza la materia. Se contempla la necesidad de entregar material sobre Blockly y otros lenguajes, al inicio del curso (es lo ideal), así como también es ideal crear canales como foros para resolver dudas y recibir retroalimentación por parte de los estudiantes. Puede considerarse la alternativa de crear una wiki donde se registren las experiencias del docente en el proceso de enseñanza con el fin de compartirlo con sus colegas.

4.4.2.3 Elaboración de un cuadro que resuma las herramientas y recursos del software libre y la manera en que han sido usados para la educación en programación de computadores. Teniendo en cuenta lo destacado en este punto hasta aquí, resulta la siguiente tabla, la cual contiene a manera de síntesis, las diversas herramientas más destacadas junto a las respectivas direcciones y los casos de éxito de las mismas que más se acercan al interés investigativo, según la revisión bibliográfica (Tabla 2). Se precisa que en cuanto a LMS como son Chamilo y Claroline, no se especifica en línea casos precisos de aprovechamiento en los procesos de enseñanza de cursos de programación, tan solo opiniones o

comentarios muy breves en foros, donde son ubicadas como potenciales herramientas para aprender teoría y evaluarla. En cuanto a Moodle, Dokeos y Edx Platform, se perciben experiencias procesos formativos alrededor de la programación, pero sobre todo en cursos masivos en línea.

4.4.3 Revisión bibliográfica sobre criterios de calidad del software libre, ingeniería de software y software educativo. Según el informe de la firma consultora *Coverity* y el Departamento de Seguridad Nacional de los Estados Unidos, iniciado en el 2006 y destacado como uno de los más completos, se recuerda que ya eran 300 millones de líneas de código propietario y 37 millones de líneas de código libre analizadas por los investigadores, reportando que en cuestión de calidad, el código abierto posee condiciones iguales o superiores incluso, que la calidad ofrecida por el software propietario. (Jaques, 2012)

En cuanto a casos donde se han definido criterios de calidad del software libre, se tienen ejemplos de investigaciones focalizadas, como la del equipo del proyecto SIGNA (Sistema Integral de Gestión y Notificación de Alarmas) en la Universidad ORT de Uruguay (Fernández, Larraz, & Silvera, 2009), con el objetivo de determinar el proceso más apropiado de selección de un sistema de código abierto. El proyecto se propuso investigar e implementar una solución fiable para el monitoreo de componentes de un centro de cómputos y posterior notificación de problemas identificados.

La meta en ese caso, era contar con mayor calidad y disponibilidad de los mismos, a través de una herramienta que permitiera detectar problemas en los distintos componentes del sistema. Los integrantes de equipo acogieron tecnologías de código libre que fueran capaces de detectar incidentes e informarlos. Estos problemas serían notificados a técnicos para que estos pudieran resolver las fallas en el menor tiempo posible. Fernández et al. Expusieron en su trabajo de investigación, la importancia del proceso de selección de software, debido a que el sistema seleccionado iba a ser un componente central en la solución final (2009).

Tabla 2. Herramientas y casos de éxito

Herramienta	Casos de éxito
Waterbear	En línea se encuentra exclusivamente un texto del 3 de mayo de 2011, día en el cual Dethe Elza, creadora de esta herramienta, la presenta en JSConf en Portland, OR. Allí se asegura que Waterbear trae los conceptos de lenguajes de programación. Ver más en http://readwrite.com/2011/05/03/waterbear-is-like-scratch-but#awesm=~os1TH05D76W15I
Scratch	Entre los diferentes casos de éxito de esta herramienta que en línea se hallan, se destaca la relatada por un docente que asegura que 300 estudiantes mexicanos trabajaron durante cuatro meses con Scratch, en quienes se generó un ambiente de aprendizaje que los motivó a desarrollar habilidades de razonamiento cognitivo y metacognitivo. El docente que relata la experiencia, recuerda que fue en un taller con Telmex años atrás cuando lo llegó a descubrir; llevaba casi dos décadas estaba empleando *LogoWriter* y *Lego Mindstorms*, pero al encontrarse con Scratch, reconoce en ella todas las ventajas para fortalecer el aprendizaje de la programación. http://scratched.media.mit.edu/stories/using-scratch-develop-reasonning-skills-mexico mas historias en http://scratched.media.mit.edu/stories
Blockly	La página 'Niños Construyendo' (http://constructingkids.com/2013/05/15/vpl/), en una entrada del 15 de mayo de 2013, se habla de esta herramienta para hacer referencia a lenguajes de programación visuales.
App Inventor	En línea se puede leer un testimonio con respecto a un proyecto de aula dispuesto con el fin de enseñar a programar en un instituto de secundaria. Se logró allí que Los alumnos crearan *apps* con esta herramienta. Eso exactamente fue en la asignatura Iniciación Profesional a la Electrónica en la Institución Educativa Palomeras Vallecas de la capital española. La propuesta consistió en que cada alumno diseñara en lenguaje de programación, al menos una aplicación para el móvil. La ponente, Ángeles Araguz, demuestra con este proyecto que la metodología es lo que importa, pues la creación de una *app* para el móvil es un pretexto perfecto para aprender a programar, a escribir código. Más en: http://www.educacontic.es/blog/inventores-de-apps
Rebeca a través del espejo	Dentro de las Actas de las I Jornadas en Innovación y TIC Educativas – JITICE 2010, existe un interesante texto de Estefanía Martín Barroso, Manuel Rubio Sánchez y Jaime Urquiza Fuentes, quienes describen el surgimiento de "Rebeca a través del espejo", basado en Alice, como aquella idea de la Universidad Carnegie Mellon, dispuesta para apoyar y motivar la enseñanza de la programación orientada a objetos. Los ponentes de esta jornada describen que durante el desarrollo de Rebeca se llevaron a cabo diversos talleres pilotos en Juvenalia y el fesTICval, al igual que concursos de animación en la Comunidad de Madrid, subrayando lo interesante de la experiencia y sus positivos resultados. Más en: http://www.etsii.urjc.es/investigacion/archivos/BoletinETSII-2010-001-JITICE-2010.pdf

Fuente: Elaboración propia

Tabla 2. (Continuación)

Herramienta LMS	Casos de éxito
Dokeos	La Universidad Santiago de Cali (usc.edu.co) dispone de un portal soportado en Dokeos para todos sus programas incluido el de Ingeniería de Sistemas. La página es http://virtual.usc.edu.co/
Moodle	Apps.co un programa del gobierno colombiano para promover el emprendimiento en TICs, hace uso de Moodle en la etapa llamada *bootcamps* https://apps.co/inscripciones/fase/bootcamps/, en ella enseñan diferentes tecnologías para programar en diferentes entornos. La dirección donde se hace eso es http://bootcamps.apps.co/
Edx platform	Esta plataforma es usada por la organización Edx www.edx.org, en la que se ofrecen cursos masivos en línea con el apoyo de prestigiosas universidades. Se pueden encontrar diferentes cursos de programación como por ejemplo: https://www.edx.org/course/mitx/mitx-6-00x-introduction-computer-science-586

Fuente: Elaboración propia

También se destaca en línea, casos como el de *OpenBRR* (http://www.openbrr.org/). Esta es una metodología de evaluación de software de código abierto anunciada en 2005, y que define un proceso de evaluación abierto y estándar. La metodología señalada trata de integrar empresas restricciones (en particular para las pruebas y fiabilidad), y se centra en el intercambio y la reducción del coste total de propiedad percibida para el software de código abierto.

Se subraya también el caso investigativo titulado 'Calidad de software de código abierto: la honradez Modelo QualiPSo', (Del Bianco, Lavazza, Morasca, & Taibi,

2011). Sus autores aseguran que la confiabilidad es una de las principales cuestiones sobre las que se basa la decisión de adoptar un software de código abierto (OSS) de producto. En su trabajo describen la manera como alcanzan los siguientes objetivos de: 1) la definición de un concepto adecuado de confiabilidad de los productos de software y artefactos, y 2) la identificación de una serie de factores que influyen en ella.

Ahí informan la manera como identificar las "dimensiones" de confianza, es decir, de las cualidades de alto nivel que los productos de software y los artefactos tienen que poseer a fin de ser considerados dignos de confianza. Estas dimensiones las describen gracias a un modelo conceptual de la honradez, con el cual la fiabilidad de los productos de software libre y artefactos es medido por las características de los productos de software libre y el nivel de confianza percibida por los usuarios de este tipo de productos. (2011)

Alejandro Toledo Tovar, Jeimmy Viviana Cuéllar Rivera y José Raúl Romero Mera (2011), realizaron un estudio titulado “Análisis de modelos de evaluación de calidad de Software Libre” publicado dentro del documento anual “Investigación en Ingeniería de Sistemas e Informática” en Tunja, Boyacá, Colombia, con lo que se enriqueció la Memoria del 2011 de Investigación en ingeniería de sistemas e informática, editada por el Grupo de Investigación en Software - GIS, Adscrito a la Escuela de Ingeniería de Sistemas y Computación, Facultad de Ingeniería de la Universidad Pedagógica y Tecnológica de Colombia – UPTC.

La sexta versión nacional y segunda internacional del encuentro de investigación en ingeniería de sistemas e informática llevada a cabo en el año 2011 en Tunja (Boyacá) - Colombia, logró demostrar la importancia que tiene el poder interactuar con comunidades científicas en las diferentes áreas cubiertas por las ciencias de la computación. En esa oportunidad se contó con la intervención de conferencistas y ponentes de diez (10) países del mundo y más de ocho (8) departamentos colombianos que a través de sus instituciones académico-investigativas presentaron temáticas que coadyuvaron al aprendizaje e integración de conocimiento en más de doscientas cincuenta personas que participaron del EIISI 2011 (Universidad Pedagógica y Tecnológica de Colombia, 2011).

Toledo et al. (2011, pág. 103), en su participación de dicho evento, aseguraron que “la mejor opción es emplear la metodología OpenBRR, a la hora de efectuar la evaluación de software libre”, por razones entre las que se cuenta la Adaptabilidad de la metodología y la Facilidad de manejo. Este estudio encontró que “cada vez es más amplio el uso de aplicaciones o proyectos FLOSS tanto en grandes, medianas y pequeñas industrias e incluso, aunque ciertamente en un menor grado, en el

hogar” (pág. 103).

Los autores aseguran que

> estas aplicaciones o proyectos se encuentran disponibles en gran cantidad de repositorios y diferentes sitios de internet de donde pueden ser descargados. Ante la gran oferta presentada, existe la posibilidad de riesgos que dificultan la toma de decisiones favorables sobre cuál herramienta seleccionar, por lo que se debe tener una metodología de evaluación diferentes a los modelos tradicionales de calidad que se emplean en el desarrollo de software privativo (Normas ISO, modelo CMMI, entre otras), estos, tipos de software, están dirigidos a un modelo de desarrollo central presente en grandes empresas de desarrollo de software. (Toledo Tovar , Cuellar Rivera, & Romero Mera, 2011).

En la misma investigación se expone que la información que se poseía en ese momento acerca de los modelos QualOSS y QualiPso OMM, estaba relacionada con su uso a nivel interno de los proyectos que los concibieron y aún se encontraban terminando de desarrollar algunas herramientas que permitirían en gran porcentaje la medición de métricas de evaluación. (Toledo Tovar , Cuellar Rivera, & Romero Mera, 2011)

Reconociendo que la Ingeniería de software es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software (Granollers, Lorés, & Perdrix, 2002), se encuentran diversos trabajos como el de Alain Abran (2002), y James W. Moore, Pierre Bourque, Robert Dupuis (1999), citado por Dolado, (2004), con cuyos aportes se asegura que “la ingeniería del software está recibiendo una atención especial dentro de los grupos relacionados con la enseñanza y los diseños curriculares informáticos” (Dolado, 2004).

En su trabajo se ofrecen algunas ideas relacionadas con el diseño general de un currículo orientado hacia la ingeniería del software. Con su trabajo concluye que “la propuesta más ambiciosa, referida a la ingeniería del software, es la del SWEBOK, aunque no sea una propuesta curricular” (Dolado, 2004).

> El Software Engineering Body of Knowledge (ver http://www.swebok.org, SWEBOK) es un subproducto de un comité conjunto (SWECC) de la ACM y del IEEE, cuyo objetivo es "establecer los conjuntos de criterios apropiados y las normas para la práctica profesional de la ingeniería del software, sobre las que se puedan basar las decisiones industriales, la certificación profesional y los currículos". (Dolado, 2004)

Al tratar de exponer lo hallado en la literatura sobre los criterios de calidad del Software educativo, es necesario recorrer lo hallado en cuanto a programas

educativos y programas didácticos como sinónimos, para designar genéricamente los programas para ordenador creados con la finalidad específica de ser utilizados como medio didáctico, es decir, para facilitar los procesos de enseñanza y de aprendizaje. (Marqués, 1996)

Esta definición engloba todos los programas que han estado elaborados con fin didáctico, desde los tradicionales programas basados en los modelos conductistas de la enseñanza, los programas de Enseñanza Asistida por Ordenador (EAO), hasta los aun programas experimentales de Enseñanza Inteligente Asistida por Ordenador (EIAO), que, utilizando técnicas propias del campo de los Sistemas Expertos y de la Inteligencia Artificial en general, pretenden imitar la labor tutorial personalizada que realizan los profesores y presentan modelos de representación del conocimiento en consonancia con los procesos cognitivos que desarrollan los alumnos. (Marqués, El software educativo, 1996)

No obstante según esta definición, más basada en un criterio de finalidad que de funcionalidad, se excluyen del software educativo todos los programas de uso general en el mundo empresarial que también se utilizan en los centros educativos con funciones didácticas o instrumentales como por ejemplo: procesadores de textos, gestores de bases de datos, hojas de cálculo, editores gráficos... Estos programas, aunque puedan desarrollar una función didáctica, no han estado elaborados específicamente con esta finalidad. (Marqués, El software educativo, 1996)

4.4.3.1 Definición de criterios de comparación de herramientas y recursos del software libre para su uso en educación para Programación de Computadores. Las fuentes de los criterios han girado en torno al diagnóstico realizado, y a criterios de calidad de software libre, ingeniería de software y software educativo. De esta manera, se definen como criterios a tener en cuenta, los siguientes:

- La facilidad de uso (instalación e interfaz de usuario)
- Las características del producto (tipo de licencia, documentación, que se encuentre en español preferiblemente)
- Objetivo (ámbito y posible descripción)
- Características técnicas (tipo, repositorio, número de personas involucradas en su desarrollo).

4.4.3.2 Realización de la comparación entre herramientas y métodos utilizando los criterios definidos. Tal como se aprecian en las figuras 3 y 4, los criterios se plantean en la primera columna como ‘Componente’, y en la segunda, sus

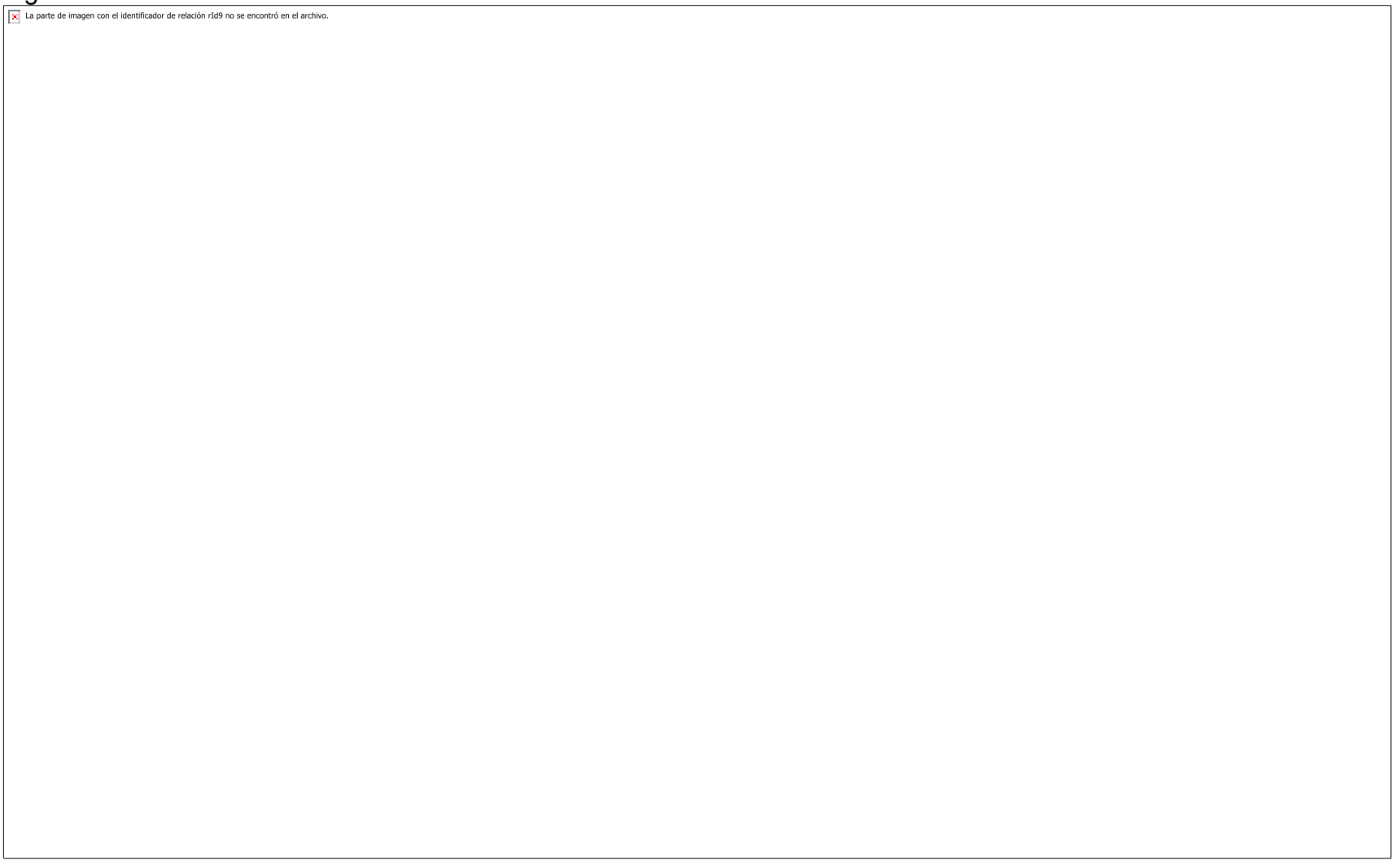
respectivos ‘elementos’. Se decide independizar el análisis de los criterios, cuando se trata de las herramientas para aprender teoría y evaluarla (Figura 4), de las herramientas cuyo fuerte es la visualización y la animación (Figura 3).

Como se puede apreciar, mientras se hallan herramientas que poseen bondades en su instalación, existen otras cuyas ventajas se centran en la interfaz de usuario. De igual manera, en algunas su documentación es más completa que en otras, y se da el caso de excelentes condiciones o características de varias de ellas, pero con el limitante del idioma, así que su selección se ajustará a tener que inclinarse por el método o estrategia que menos impedimentos o trabas presente.

4.4.3.3 Diligenciamiento de una matriz que sintetice la comparación de herramientas y métodos y su confrontación con los problemas identificados en el diagnóstico. A continuación, se presenta una matriz (Figura 5) que pretende de manera concisa, correlacionar los problemas diagnosticados con las principales herramientas descritas hasta este punto. La viñeta () indica la posibilidad intrínseca que posee la herramienta para vencer el problema (P). Es entendible la razón del porqué Blockly termina siendo una herramienta predilecta, y por lo tanto, una buena opción en proyectos educativos más aun cuando se desean vencer los problemas descubiertos con el presente estudio. Se trata de una herramienta con la que se aprende jugando, sirve para opinar, y visualizar diversas iniciativas de uso, y lo mejor de todo: el código es libre, es abierto.

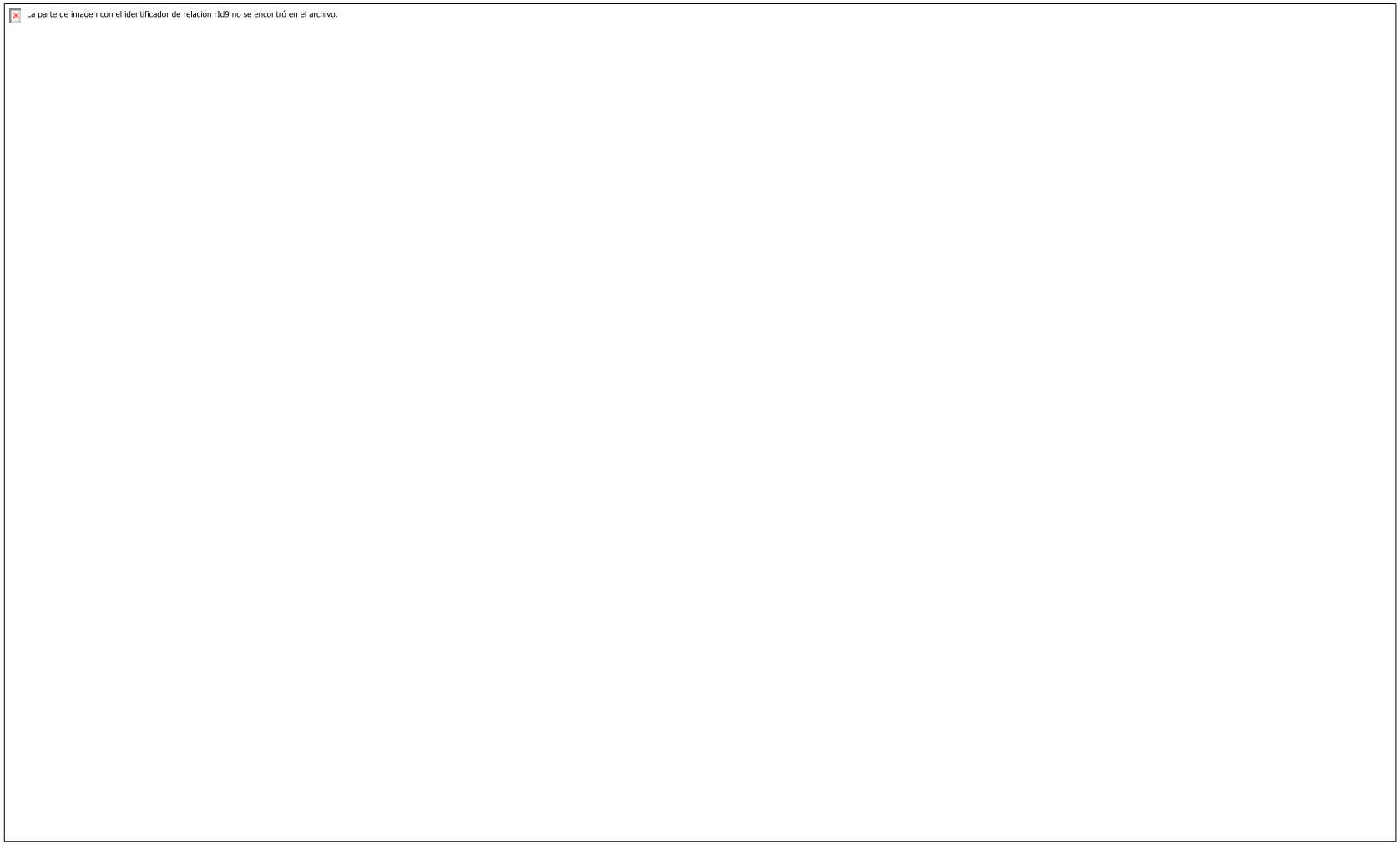
4.4.4 Modificación del plan de curso de la asignatura de Programación de Computadores ofrecida a programas de ingeniería de la Universidad Antonio Nariño. Ante un programa o plan de curso donde la enseñanza se enfoca aprender conceptos sobre Programación Orientada a Objetos contempla un mínimo porcentaje dedicado a la práctica, y donde al estudiante no se le brindan mayores posibilidades de aprender haciendo, o en donde la combinación de la teoría con la práctica es poco probable, se hace ideal crear situaciones que conciban el entusiasmo en el estudiantado por la materia, y su búsqueda por aprender la programación. Es indispensable modificar el plan actual de curso, puesto que los problemas diagnosticados así lo ameritan.

Figura 3. Herramientas innovadoras



Fuente: Elaboración propia

Figura 4. Herramientas para acompañar el proceso de enseñanza-aprendizaje



Fuente: Elaboración propia.

Figura 5. Matriz herramientas/Problemas diagnosticados

		Problemas Diagnosticados				
		P_1	P_2	P_3	P_4	P_5
Herramientas	Waterbear	✓				✓
	Scratch		✓	✓	✓	✓
	Blockly	✓	✓	✓	✓	✓
	App Inventor	✓	✓	✓		✓
	Rebeca a través del espejo	✓				✓
	Chamilo		✓	✓	✓	✓
	Claroline		✓			
	Dokeos		✓	✓	✓	✓
	Moodle		✓			
	Edx Platform		✓	✓	✓	✓
Convenciones: P_1: Elevado nivel de complejidad en el aprendizaje de algoritmos P_2: Problemas relacionados con la enseñanza de teoría y evaluación de ritmos particulares de aprendizaje P_3: Insatisfacción con el proceso enseñanza - aprendizaje P_4: Desmotivación en el estudiantado (encuentra complejo o poco llamativo el curso) P_5: Poca sentido del curso con respecto a la profesión (Ingenierías diversas a la de Sistemas)						

Fuente: Elaboración propia

Como se viene planteando, el Plan de Curso de la materia Programación de Computadores que se conserva vigente en la Universidad Antonio Nariño, no prevé el acogimiento de una herramienta didáctica para el aprendizaje, o un método específico para la revisión de un texto guía o sistema tutorial interactivo, ni mucho menos para la autoevaluación o para el reconocimiento de saberes previos. Se hace, por tanto, necesario proponer un plan donde el maestro juegue un rol diferente al tradicional; sea un facilitador y motivador permanente. Para lograr esto, se ha seleccionado un grupo experimental y un grupo control, y se ha revisado detenidamente el plan del curso de programación de Computadores con el ánimo de formular una posible razón sencilla pero técnica, por la cual surgen los problemas detectados con el diagnóstico. Posteriormente y para lograr este objetivo, se identifican las posibles modificaciones al plan de con el que se hace necesaria la incorporación de recursos y herramientas del software libre.

4.4.4.1 Selección del curso de Programación de Computadores que se intervendría. Se decide sostener la propuesta de población muestra y grupo control expuesta en 4.3.1, de tal forma que los grupos 3 y 4 de la jornada 5 (Anexo C), del periodo 2-2013 Ingeniería Mecánica, en la UAN sede Buganviles-Neiva, conforman la población donde se implementará la propuesta. El curso 2 se dejará como grupo control, y a este no se le dará a conocer ningún tipo de modificación al plan del curso.

4.4.4.2 Revisión del plan del curso seleccionado. El plan del curso se encuentra delineado por el aprendizaje de un lenguaje como C++. Este lenguaje presenta una empinada curva de aprendizaje para el estudiante que nunca ha escrito código, y esta es una hipotética razón por la que ellos temen a la asignatura y se presentan otros factores como la deserción académica. Este Plan de Curso parece ser, se ha venido orientando exclusivamente al aprendizaje de dicho lenguaje, y tal vez se ha dejado a un lado el fomento de las habilidades para que de manera progresiva los estudiantes se apropien de prácticas innovadoras.

4.4.4.3 Identificación de las posibles modificaciones al plan de curso para la incorporación de recursos y herramientas del software libre. El plan del curso no está ligado a algún lenguaje de programación específico, mientras se propone que sea un lenguaje moderno de alto nivel. Se considera dar clases prácticas y teóricas con herramientas innovadoras, de fácil uso y portables, adaptables a cualquier dispositivo y que puedan ser usadas fuera y dentro de la clase. Considerando estas necesidades, las herramientas deben ser en ambiente web.

Se necesita una herramienta o lenguaje gráfico que sea de fácil uso para los estudiantes, que permita explorar todas las estructuras de un lenguaje de programación tradicional. Con esa necesidad diagnosticada, el que más se ajusta a las necesidades es Blockly; un entorno de programación que permite por medio de estructuras gráficas hacer el camino hacia la programación de computadores algo más natural y fácil al estudiante. Como se ha dicho en páginas anteriores, Blockly es una herramienta que permite crear un lenguaje de programación con una leve curva de aprendizaje para el alumno que nunca antes ha escrito código.

Otra opción de herramienta, y bastante propicia para ofrecer la teoría, es el sistema de gestión de aprendizaje (LMS), que permite a los estudiantes acceder a contenidos teóricos de la materia y también evaluar sus avances por medio de cuestionarios. Se hace referencia, entonces, a Chamilo LMS, como la más indicada para esta tarea. Como se pudo apreciar en las figuras 4 y 5, se ubica como una

tendencia moderna y bastante completa, ideal para acompañar el proceso de aprendizaje en el Curso de Programación de Computadores.

4.4.4.4 Definición de las reformas al plan de curso para incorporar herramientas y métodos del software libre. En vista de que el plan del curso original plantea lenguajes y paradigmas de programación que puede resultar muy difícil para los estudiantes. Se proponen, entonces, actividades dentro y fuera de la clase con herramientas innovadoras, que promueven la participación y creatividad de los alumnos no solo en el aula sino también fuera de ella. Se propone llevar el desarrollo de las actividades de final del curso de programación de Computadores al punto en el cual los estudiantes sientan que están programando.

Como se ha partido de la mirada reflexiva hecha a otras experiencias logradas en ambientes similares, se propone que al finalizar el curso, se lleve a cabo esa implementación parcial. La actividad se dispone para ocho días aproximadamente, y se segmenta en dos principales fases.

Fase Uno: Socialización de la metodología con los estudiantes, presentación de las herramientas y técnicas para resolver los problemas planteados por el docente, quien brinda opciones para su real análisis, desarrollo e implementación, acogiendo el lenguaje de programación.

Fase Dos: Evaluación y retroalimentación; el docente evalúa el avance de los estudiantes, de los temas propuestos y repite la dinámica de la fase uno con los nuevos temas según el plan del curso.

4.4.4.5 Realización de las reformas definidas al plan de curso. Se ha previsto reformar el actual Plan de Curso, el cual dispone una serie de contenidos, con actividades en su mayoría magistrales, talleres o trabajos en sala de sistemas con problemas propuestos para dentro y fuera de la clase. El plan del curso pretende que el estudiante con una serie de actividades adquiera conocimientos en programación con el lenguaje C++ incluida el paradigma de la orientación a objetos. Al estudiante se le enseña teoría en una clase magistral, y los espacios para la práctica se disponen mediante talleres con ejercicios para resolver en papel como también talleres en el laboratorio con herramientas de escritorio como zinjai, DFD y PSINT, que son regresados al docente para que sea él quien supervise y corrija los ejercicios en el debido caso.

El presente trabajo pretende abordar esta actividad de enseñanza con una metodología similar pero con herramientas más llamativas y acordes a lo esperado

por los estudiantes, y desde luego, más innovadoras. Como es argumentado en otras etapas de este documento, el lenguaje C++ presenta una empinada curva de aprendizaje para alguien que inicia a escribir código, y tal vez esta circunstancia infunda desgano y temores frente al curso, motivando, en lugar de un aprovechamiento de las habilidades que se pueden fortalecer con el curso, aspectos poco positivos, como la deserción académica. El Plan del Curso, entonces, se ha actualizado para que el estudiante trabaje con lenguajes de alto nivel en general (lenguajes como JavaScript o Python) y amplíe sus nociones sobre estos al finalizar el curso; para llegar a ellos, se parte de un lenguaje visual como Blockly, el cual es el recomendado por ahora.

La clase magistral se seguirá ofreciendo para explicar los objetivos, compartir teoría y resolver dudas, aunque se confía en que se contará con mayor interés por parte de los estudiantes, quienes se hallarán motivados para llevar a la práctica lo que ofrezca su docente. Por ello, aparece un recurso más en el que se apoyará fuertemente el curso, se trata de un LMS reconocido como Chamilo, para ampliar la teoría y atender inquietudes de los estudiantes en contextos diversos al de la clase. En esta etapa de implementación parcial, lo que se desea es facilitar la introducción a un lenguaje; por lo tanto, no se abordan temas avanzados, se resuelven problemas con el lenguaje visual Blockly, y se proponen ejercicios para que el estudiante se familiarice con las herramientas en software libre y sus características.

4.4.4.6 Selección de los grupos de Programación de Computadores de ingeniería en los cuales se haría la implementación o prueba parcial. Se continúa sosteniendo la propuesta de población muestra y grupo control expuesta en 4.3.1, de tal forma que los grupos 3 y 4 de la jornada 5 (Anexo D), del periodo 2-2013 Ingeniería Mecánica, en la UAN sede Buganviles-Neiva, conforman la población que se intervendrá. El curso 2 se dejará como grupo control (Anexo E).

4.4.4.7 Definición de las actividades del plan de curso reformado que se realizarían en los cursos que se van a intervenir. Para esta sub-actividad, el investigador y docente, presenta los cambios al Plan del curso y las mejoras con respecto a la versión anterior; proponiendo su discusión con la que se llegue a conocer los intereses de los estudiantes y posterior presentación de las herramientas como algo innovador (intentando ganar el interés de los estudiantes). Tejida la socialización de las dos herramientas, se confirma que todo está listo para su parcial implementación. En el caso del LMS Chamilo, se invita a los estudiantes a que se registren en el sistema. Luego, el docente realiza ejercicios con Blockly solamente con el ánimo de que los estudiantes se familiaricen con la aplicación.

Se enfatiza que con Blockly se seguirá el Plan del Curso, por medio de ejercicios, avanzando en el temario del mismo, y en la complejidad de estos. En Chamilo LMS, podrán encontrar la teoría y se disponen foros por cada tema previsto del Plan del curso, de tal forma que ante un nuevo tema, un nuevo foro será creado; se espera que en estos los estudiantes planteen sus inquietudes, dudas o sugerencias. La evaluación es permanente y se pide la valoración del nuevo Plan del Curso tal como se ha presentado y definido, en una fecha final; esto, de cierta manera, se puede percibir a través de la guía que se aplica a manera de entrevista, en el último encuentro presencial con los estudiantes.

4.4.4.8 Diseño de un instrumento de recolección de información de los grupos a los que se haría seguimiento. Luego de la implementación parcial del programa curricular del curso de Programación de Computadores, se aplica una guía de preguntas de manera individual en cada uno de alumnos de los tres grupos (los dos grupos que hicieron parte de la experimentación, así como el grupo control). La idea básica consiste en poder apreciar aspectos conceptuales en torno a la temática final de la materia, así como las impresiones generales frente a la metodología adoptada por sus docentes. En el mismo formato, cada estudiante firma el consentimiento respectivo.

En estos términos, se considera como instrumento válido uno en el cual se atiendan las opiniones abiertas y sinceras de los estudiantes (Anexo F), sin imprimirle ninguna carga subjetiva del investigador. Pensando en esto, se definen los siguientes cinco puntos: 1) ¿Siente que se apropió de los contenidos propuestos en el plan de curso? ¿Por qué? 2) ¿Qué opina sobre el trabajo participativo alcanzado durante el desarrollo de la temática final? 3) ¿Qué equipos tecnológicos llegó a emplear durante el curso? ¿Cuáles durante las clases y cuáles fuera de ella? 4) ¿Cuál fue el proceso que más favoreció la comprensión del paradigma de la programación orientada a objetos? 5) ¿Qué recomendación haría al docente de programación en Computadores?

4.4.4.9 Programación y organización de las actividades que se realizarían en los cursos a los que se haría seguimiento (Grupo experimental). La Tabla 3, expone la respectiva programación. Propone tener en cuenta fecha, actividad, recursos, lugar y evaluación.

Tabla 3. Programación y organización de las actividades

Fecha	Actividad	Recursos	Lugar	Evaluación
20 al 24 nov. 2013	1. El docente propone la metodología y herramientas a la clase. 2. Los estudiantes ingresan a http://formacion.cholupa.com/ y se registran, se unen al curso "Programación de Computadores" creado con anterioridad por el docente. 3. Exploración de la plataforma. 4. Se despejan dudas de manera presencial. 5. Socialización de la herramienta Blockly. 6. Se despejan dudas y el docente plantea un problema típico de programación en el ambiente Blockly. 7. Se aprovecha la herramienta para resolver el problema. 8. El docente presenta desafíos diseñados en Blockly; "Puzle" y "Calculadora de asientos de avión" para ser resueltos en clase por parte de los estudiantes. 9. El docente deja de trabajo extra clase el desafío llamado "Laberinto." 10. Fuera de clase los estudiantes plantean y resuelven inquietudes mediante el foro de Chamilo. 11. En Chamilo el docente crea como tarea un espacio para recibir el enlace de las actividades propuestas en Blockly. 12. Los estudiantes entregan las actividades propuestas en Blockly por medio de un enlace único que éste genera; así el docente puede revisar la actividad.	Tecnológicos: Chamilo Blockly PC, video-beam. Tabletas, Teléfonos Inteligentes	Sala de Sistemas Universidad Antonio Nariño Extramuros	Se valora el entusiasmo, la participación y el nivel de cumplimiento con el registro en el portal. Se valora la resolución de los ejercicios propuestos en Blockly y publicación de respectivos enlaces en Chamilo.
25 y 26 nov. 2013	1. El docente resuelve las inquietudes de las actividades propuestas. 2. El docente expone la sección "Código" de Blockly en donde se analizaron los diferentes bloques y como corresponde cada uno de ellos con la estructura del lenguaje. Tipos de datos. Operadores. Librerías. Funciones de entrada/salida Estructuras de control. Sentencias, expresiones y asignaciones. 3. Los Estudiantes hacen preguntas y resuelven inquietudes haciendo ejercicios en Blockly. 4. El docente publica en Chamilo nuevos contenidos teóricos conforme avanza el plan del curso y crea tareas para que los estudiantes los desarrollen fuera de clase.	Tecnológicos: Chamilo Blockly PC, Tabletas, Teléfonos Inteligentes	Sala de Sistemas Universidad Antonio Nariño Extramuros	Se valora la participación activa de los estudiantes. Dominio de teoría propuesta por el docente. Entrega de tareas por su respectivo canal.

Tabla 3. (Continuación)

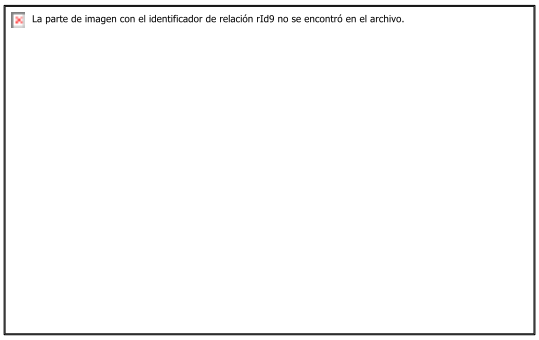
Fecha	Actividad	Recursos	Lugar	Evaluación
27 nov. al 2 dic. 2013	1. El docente resuelve las inquietudes de las actividades propuestas. 2. El docente plantea nuevos ejercicios delineado con el plan del curso para los temas: Funciones. Declaración de prototipos. Argumentos y parámetros, Arreglos. Vectores. Matrices. 3. El docente enseña cómo es de simple tomar el código fuente de ejercicios en C++ y pasarlo a Blockly. También enseña la funcionalidad que trae consigo Blockly de exportar hacia lenguajes como JavaScript o Python. 4. Los estudiantes proponen nuevos ejercicios en la clase. 5. El docente publica en Chamilo nuevas tareas que se deben desarrollar en Blockly, ahora propone que además de entregar los vínculos de las actividades, entreguen el código JavaScript resultante, debidamente documentado (comentarios en el código explicando su funcionalidad).	Tecnológicos: Chamilo Blockly PC, Tabletas, Teléfonos Inteligentes	Sala de Sistemas Universidad Antonio Nariño Extramuros	Se valora la participación activa durante la clase. Dominio de teoría propuesta por el docente. Entrega de tareas por su respectivo canal.
2 dic. 2013	1. El docente resuelve las inquietudes de las actividades propuestas. 2. El docente plantea nuevos ejercicios delineado con el plan del curso para los temas: Recursividad. Construcción y prueba de funciones recursivas. Recursividad mutua. 3. El docente plantea nuevos ejercicios los cuales los estudiantes deberán resolver en Blockly, tomar el código JavaScript resultante en un editor de texto guardarlo y ejecutarlo en un navegador. 4. El docente aborda el tema de la programación orientada a objetos usando el lenguaje JavaScript, enseña la estructura de un constructor y propone ejercicios.	Tecnológicos: Chamilo Blockly PC, Tabletas, Teléfonos Inteligentes	Sala de Sistemas Universidad Antonio Nariño Extramuros	Se valora la participación activa durante la clase. Dominio de teoría propuesta por el docente. Entrega de tareas por su respectivo canal.
3 dic. 2013	Aplicación de instrumento (Entrevista grupos a los que se hace seguimiento)	Tecnológicos: *Google spreadsheet* PC, Tabletas, Teléfonos Inteligentes	Sala de Sistemas Universidad Antonio Nariño	Responden a la entrevista

Fuente: Elaboración propia

4.4.4.10 Ejecución de las actividades definidas de incorporación del software libre en los grupos seleccionados. A través de algunas capturas de pantalla, se apoya la interesante ejecución de las actividades realizadas con los estudiantes pertenecientes al grupo control. A continuación, su síntesis.

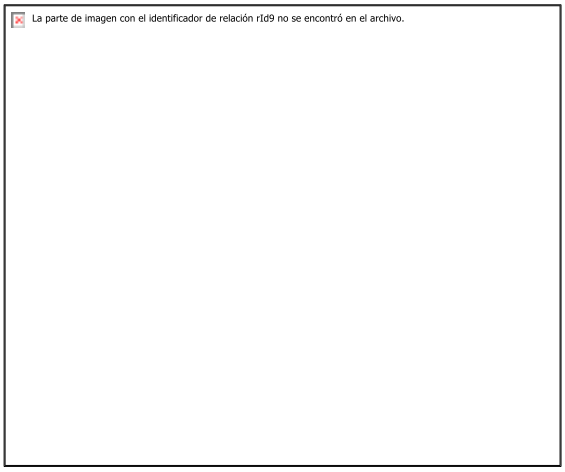
A. Se presentó la plataforma Chamilo a los estudiantes. Por medio de la url www.formacion.cholupa.com, los alumnos ingresaron y se registraron con sus datos. El registro del estudiante es usado para personalizar su intervención activa, a la vez que el docente y los demás miembros pueden identificarse mutuamente; al docente se le facilita la posterior evaluación de los mismos. (Figura 6)

Figura 6. Captura de pantalla Página Principal propuesta del docente investigador



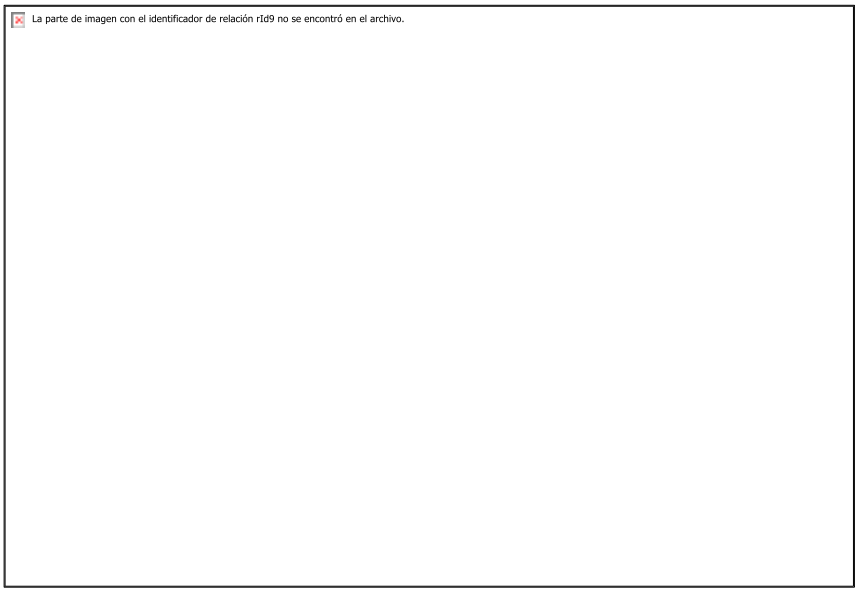
La propuesta incluye la interfaz del docente. Desde allí él puede gestionar sus cursos, admitir estudiantes, publicar material, etc. (Figura 7)

Figura 7. Interfaz para el Docente



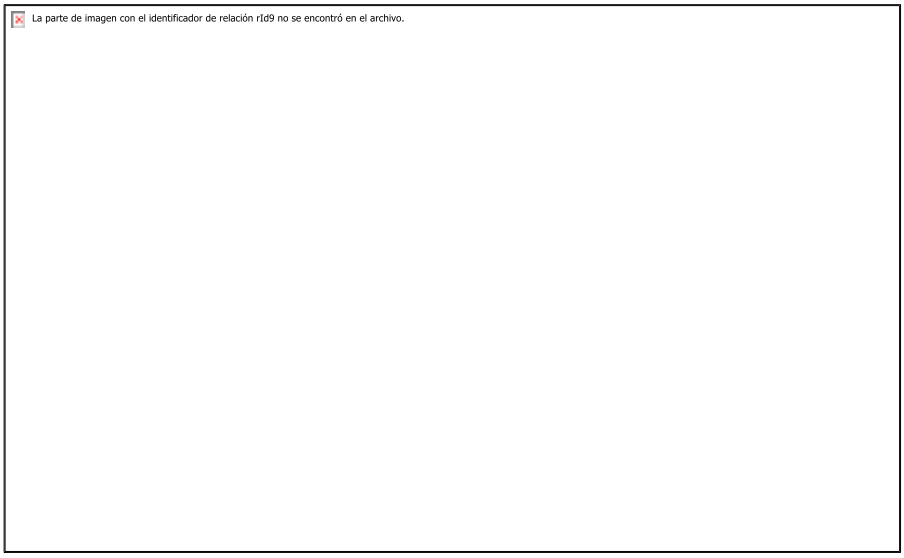
Además, el docente crea materiales para los estudiantes por medio de sencillas herramientas de edición. (Figura 8)

Figura 8. Creación de un documento



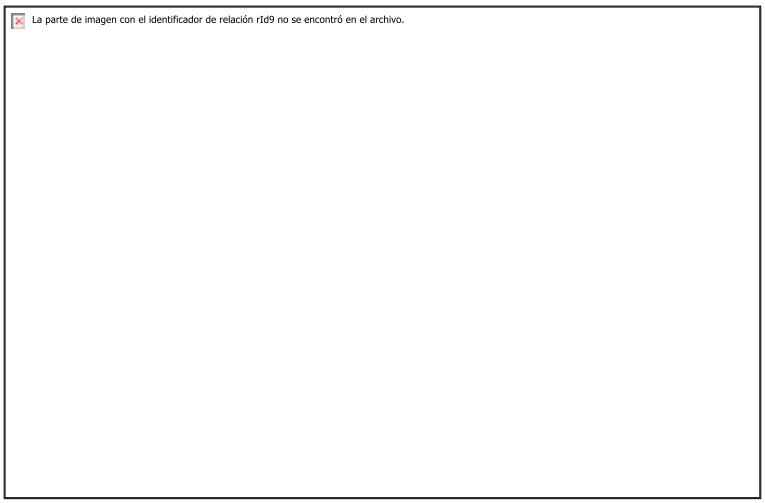
La propuesta incluye la posibilidad de que los estudiantes y docentes participen en los foros de forma activa. (Figura 9)

Figura 9. Trabajo en los Foros



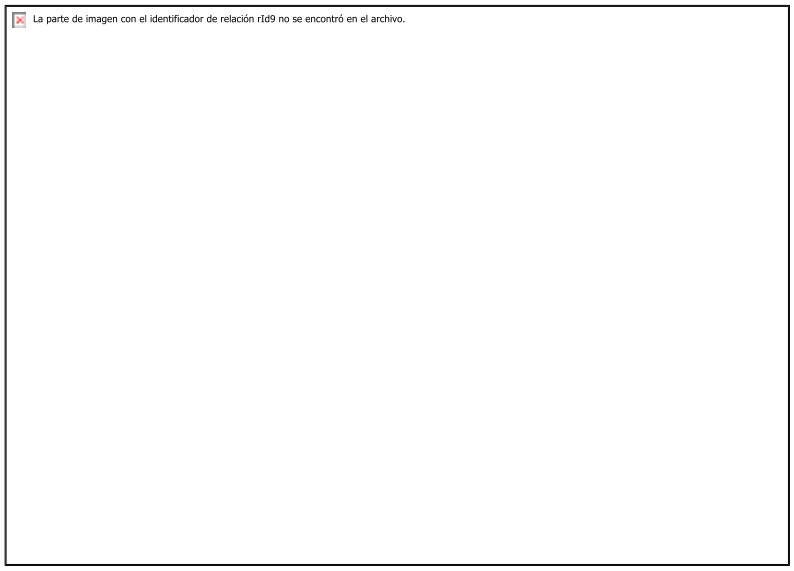
Se puede también destacar, la Vista desde el perfil de un estudiante: De esta forma, un estudiante está listo para la creación de una entrada en el foro. (Figura 10)

Figura 10. Vista desde el perfil de un estudiante



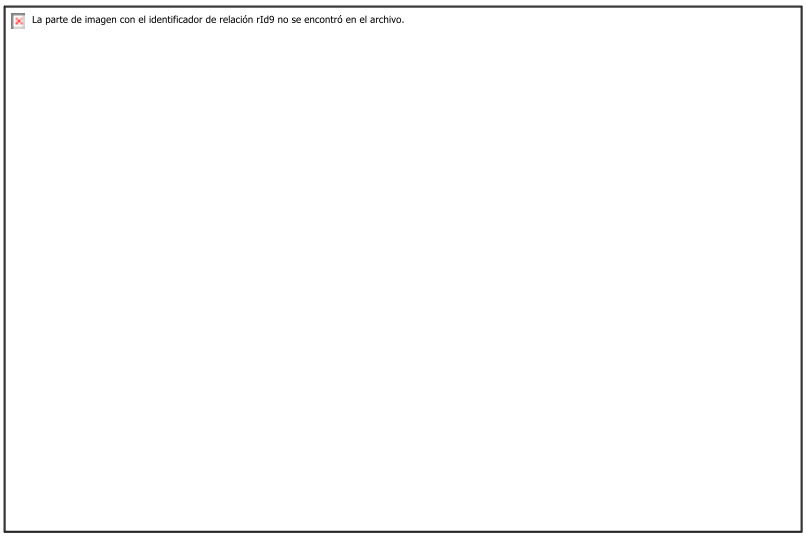
B. Como parte de la ejecución de las actividades, se destaca de manera central el aprovechamiento de la herramienta Blockly. Con un ejercicio muy simple, se ha introducido al estudiante en la mecánica de trabajo con un lenguaje visual. (Figura 11)

Figura 11. Introducción a Blockly



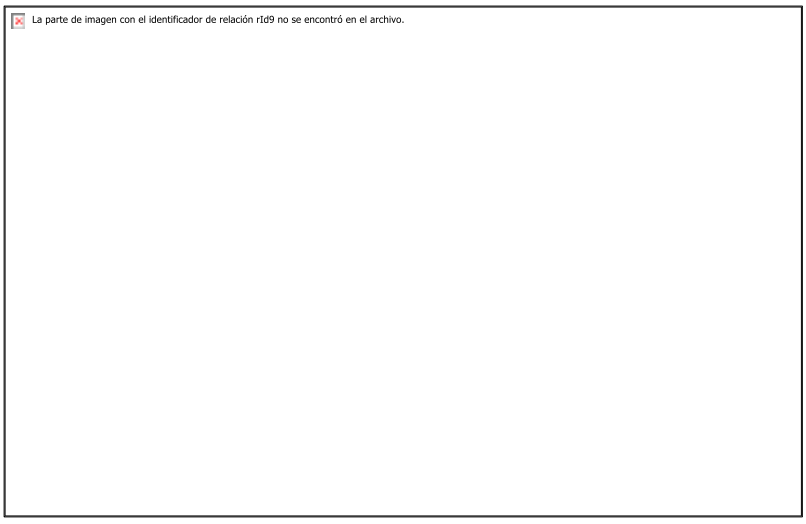
Usando la herramienta Blockly, el estudiante resuelva el problema con los bloques entregados; son 3 ejercicios en este nivel. (Figura 12)

Figura 12. Ejercicio de cálculo



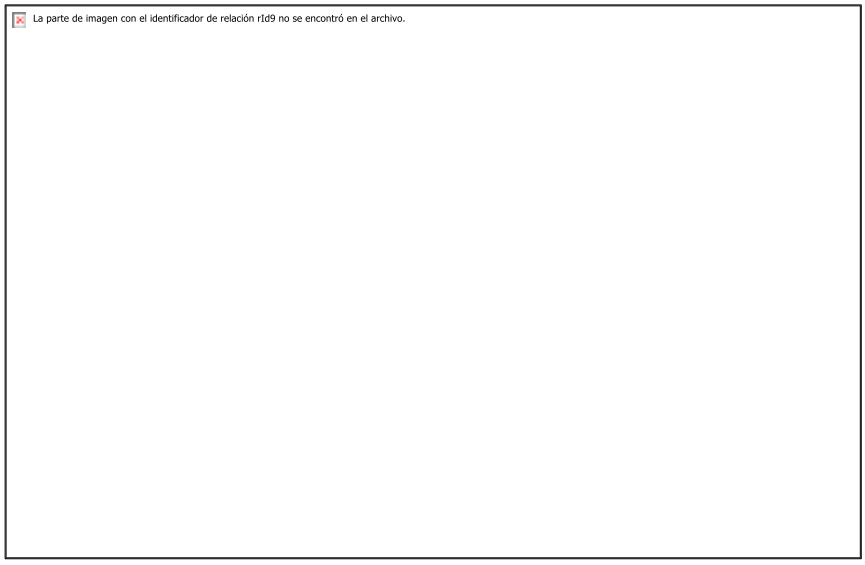
En esta etapa de la ejecución de las actividades programadas, los estudiantes han reconocido las características del lenguaje y demostraron poder desarrollar ejercicios propuestos por el docente y por ellos mismos. La Figura 13 evidencia un ejercicio propuesto por un estudiante.

Figura 13. Ejercicio Libre



En este ejercicio, el estudiante escribe un programa con Blockly y lo ha exportado hacia otros lenguajes. En la Figura 14 se puede apreciar la captura de los 3 lenguajes.

Figura 14. Creando Aplicaciones



C. Dentro de la ejecución de las actividades, se destaca la posibilidad para los actores de ejercer roles más activos y significativos. Se puede aseverar que fue creciente el interés y participación por parte de los estudiantes quienes constantemente acudían al docente para interrogarle por temas que estaban más allá de lo programado en esa tutoría.

D. Para finalizar la descripción de lo que fue en sí la ejecución de las actividades, se destaca que finalizando la programación, los estudiantes iniciaron el ejercicio de empezar a escribir código JavaScript desde cero, ya sin ayuda de la herramienta Blockly.

4.4.4.11 Aplicación del instrumento de recolección de información en los grupos seleccionados. Con la meta clara de poder apreciar en los estudiantes sus impresiones con respecto al Curso de Programación, se aplica en la fecha del último encuentro presencial del docente con los estudiantes, el instrumento guía de preguntas que se aprecia en el Anexo F, inmediatamente después de la implementación parcial. Con él se confirma si la implementación innovadora arroja resultados favorables, o si por el contrario, el problema detectado persiste en el

grupo sometido al experimento; y en cuanto al grupo control, se verifica si el diagnóstico levantado persiste o, si por el contrario, este ha variado.

El cuestionario de 5 ítems fue resuelto por 20 estudiantes del grupo control [M=12, H=8], de los 27 que iniciaron dicho curso. En cuanto al grupo experimental, se confirman 19 estudiantes [M=0, H=19] de uno de los dos cursos que estuvieron dispuestos a resolver la guía, mientras que del otro salón, se recaudaron 17 [M=1, H=16] cuestionarios atendidos. Sin embargo, del total de entrevistas resueltas [T=56], es necesario precisar que el 25% [T=14] de ellas se regresaron de manera incompleta, posiblemente a las premuras propias de los estudiantes en esta parte final del semestre.

4.4.4.12 Análisis y reflexión a partir de la información recolectada en los grupos a los que se hizo seguimiento. Con el análisis de la información recolectada a través del instrumento dispuesto, se descubre lo que podría sostenerse y, de cierta manera, lo que podría avanzarse para proponer un definitivo Plan de Curso guiado por la implementación de herramientas innovadoras, basado en el aprovechamiento del Software Libre, en la materia Programación de Computadores. Aquí, desde luego, se expone el análisis cualitativo de lo arrojado por el instrumento. Para eso, se decide tomar pregunta por pregunta, identificar las respuestas más dicientes de manera literal, y manifestar la precisa reflexión investigativa.

Pregunta Primera: Frente a la pregunta: *¿Siente que se apropió de los contenidos propuestos en el plan de curso? ¿Por qué?* Algunos participantes del grupo experimental oscilan alrededor de frase como estas:

- *Sí, esta vez aprendí más rápido… de pronto fue porque estaba viendo por segunda vez la materia, pero lo cierto es que al fin entendí. Estuvo más fácil la manera de enseñar el profe'.*
- *La verdad, sí. Me había infundido miedo al curso, pero nada de eso; me sirvió mucho trabajar desde la casa. El internet falla mucho en la U, y entonces eso es bueno.*
- *Me siento bastante satisfecho con esta clase, aunque fue corto el tiempo con las nuevas herramientas yo pude entender mejor el contenido del curso.*
- *Aprendí muchas cosas nuevas en esta clase, de forma más interactiva. apropiarme de todo un plan es difícil pero si siento que estoy cerca.*
- *Yo siento que dediqué más tiempo al final de curso, ya que tocaba estudiar en clase y en casa y repasamos todo de nuevo con las nuevas herramientas, creo que por eso aprendí más esta vez.*

De igual manera, se atienden las respuestas del grupo control, quienes a la pregunta primera presentan respuestas como estas:

- *Sí, aunque toca seguir practicando.*
- *Sí, porque sé qué es, entiendo sobre funciones, declaración de prototipos, vectores, matrices, etc.*
- *No mucho, pero es cuestión de seguir practicando.*
- *siento que aprendí cosas nuevas pero me hace falta mucha práctica para apropiarme del plan del curso*
- *Si aprendí, fue un poco difícil porque era primera vez viendo este tema y de sistemas no sé mucho.*

La investigación hasta aquí llevada, confirma que los estudiantes que pretenden aprender programación demuestran más progresos académicos cuando se acude a herramientas de Software Libre (Gómez-Albarrán, 2005). Las ventajas de la materia se constatan al final del curso con la implementación parcial, cuando los estudiantes del grupo experimental expresan que sus miedos e inseguridades, propias de quien posee incluso débiles conocimientos, se tornan menos fuertes ante el empleo de los recursos innovadores presentados por el docente. Debilidades que aún persisten en un buen número de quienes pertenecieron al grupo control.

Los alumnos pertenecientes al grupo control no están seguros de progresos significativos en el curso de Programación de Computadores, lo que lleva a pensar que un Plan de Curso apoyado en las herramientas en Software Libre, arroja mayor apropiación de los contenidos brindados. Por otra parte, se hace indiscutible la oportunidad de seguir sosteniendo para todo el currículo de la materia y para todos los grupos que se abren cada semestre en la Universidad Antonio Nariño de Neiva en su facultad de Ingeniería, el aprovechamiento del Software Libre.

Pregunta Segunda: En la pregunta *¿Qué opina sobre el trabajo participativo alcanzado durante el desarrollo de la temática final?* Se atendieron respuestas como estas (dentro del grupo experimental):

- *Pues muy bueno, pero yo esperaba que todos participáramos, pero no fue así.*
- *Con el internet de la U, no es tan fácil. Pero es muy buena la herramienta Blockly, y el Chamilo. Lo que pasa es que hay que trabajar más en casa, o donde el internet ayude.*
- *La idea es excelente, trabajar en línea, y ayudándose unos con otros, es genial. Al profesor se le notó su gran interés. Qué bueno sería que se hubiera hecho así todo el tiempo.*

- *Creo que la combinación de ambientes virtuales y presenciales es muy bueno, nos permite comunicarnos todo el tiempo y aclarar las dudas más rápido y fácil.*
- *Fue muy interactivo, era muy fácil pedir ayuda del profesor o de mis compañeros fuera de la clase.*

Mientras que en el grupo control, se hallaron posiciones como las siguientes:

- *El ratico de la clase no alcanza mucho para trabajar de manera participativa. Varias veces uno se queda con las ganas de haber participado en la clase, pero como que el profesor lleva su tiempo preciso para enseñar el tema, y no se puede atender más de tres participaciones.*
- *Me parece normal, similar a otros cursos.*
- *participamos en clase según el tiempo que teníamos, pero surgen dudas y tenía que esperar hasta la otra clase.*
- *El profesor era muy amable respondiendo mis preguntas, pero al preguntarle a mis compañeros me di cuenta que algunos no lo tenían muy claro.*
- *Tenía preguntas pero todo el mundo estaba preguntando y a veces me daba algo de pena.*

Sobre la importancia de fomentar las oportunidades de participación por parte del estudiante, se viene hablando desde hace casi dos décadas (Marqués, 1996), (GNU Operating Sistems, s/f), (Shockey, 2005), (Romero, 2006), y estudios más recientes como el de los Hermanos Cisneros Arocha (2008) y el de Vizconde et al. (2012) lo recalcan, ofreciendo mayor evidencia en cuanto a que con el aprovechamiento del Software Libre, esta práctica pedagógica es atendida.

Estudios como el de los Hermanos Cisneros Arocha (2008) anunciaban la efectividad en la implementación del software educativo, por el apoyo, refuerzo y evaluación del aprendizaje, y esta parte de la entrevista lo ratifica. Al igual, se destaca el acierto en la selección de las herramientas Blockly y Chamilo LMS, aplicadas con criterios de selección y obedeciendo a un diagnóstico que, como se aprecia con las respuestas del grupo control, conserva su vigencia. Seguirá siendo clave el acompañamiento del docente que en plena era digital, debe trascender del aula; y no se descarta la necesidad de tener en cuenta los ritmos propios de aprendizaje. Todas estas condiciones, se suplen en el acogimiento de recursos innovadoras en entornos Web.

Pregunta Tercera: Al consultar sobre las ayudas mediáticas con la pregunta *¿Qué equipos tecnológicos llegó a emplear durante el curso? ¿Cuáles durante las clases y cuáles fuera de ella?*, se obtuvo dentro del grupo experimental, respuestas como las siguientes:

- *Usé la Tablet, me sirvió mucho, aunque más fuera del aula.*
- *Además de lo normal que es el pc, use mi tablet que uso a diario.*
- *En clase y en la casa, resultó útil el Smartphone.*
- *Un compañero trae su Tablet, y pues la comparte conmigo; nos reunimos después de clase y es bien práctico aprender programación allí.*
- *Hice uso de todos los que tengo, las herramientas como están en internet, pude acceder a ellas por medio del pc del laboratorio, de la casa y de mi smartphone*.

En el grupo control, por su parte, se encontraron respuestas como estas:

- *Los equipos de la sala de sistemas y el computador de la casa para hacer tareas.*
- *Mi portátil que casi siempre llevo a clases.*
- *En el laboratorio de informática, con los computadores con nos prestan.*
- *Con mi pc en mi casa, y los de la sala de informática cuando nos llevaron.*
- *El computador asignado para la clase, y el portátil de mi casa.*

Lo anterior simplemente evidencia que los equipos como son los celulares que ya cuentan con tecnología de punta y están al alcance de los jóvenes de las Universidades, o las Tablet, y no solo los portátiles y los PC, deben ser aprovechados para el proceso de aprendizaje. Estudios como el de Salinas (2004), Cebrián (2004) y Duque (2009), citados dentro de esta investigación, ya lo hacían ver.

Pregunta Cuarta: Se indagó por el proceso que más benefició el aprendizaje de cierta temática, con la pregunta *¿Cuál fue el proceso que más favoreció la comprensión del paradigma de la programación orientada a objetos?* Ante ella, se descubrieron respuestas como estas dentro del grupo experimental:

- *Con uno o dos compañeros por el chat nos comunicábamos, y pues íbamos comparando propuestas del código escrito. Nos ayudábamos así, eso nos sirvió, y parecía que estuviéramos jugando con el Blockly.*
- *Con los que estuvieran conectados y dentro del Chamilo, íbamos contándonos las dudas, y no sentíamos ya nada de nervios. El ambiente web para trabajar fue una gran idea del profesor.*
- *La clase magistral ayudo con la teoría, pero en la práctica por medio de Blocky fue muy fácil entender la dinámica.*
- *Los ejercicios prácticos, y sencillos.*
- *Usar Blockly para hacer más fácil la programación y que cuando ya nos sentíamos seguros aprendimos JavaScript.*

En cuanto al grupo control, las respuestas giraron en torno a:

- *La explicación en clase, aunque al escribir código no me es claro.*
- *Creo que el uso de los programas que nos dio el profesor para hacer ejercicios prácticos.*
- *Los ejercicios en el pc, a pesar del poco tiempo en la sala.*
- *La teoría me ayudó a entender y con un poco de práctica en la sala me resultó un poquito más claro.*
- *La clase magistral con ejemplos que daba el profesor, pero al programar me confundí un poco.*

Se reitera que es aún más complejo el proceso del aprendizaje con prácticas alejadas del aprovechamiento brindado por el Software Libre. Se logra apreciar, en cambio, mayor seguridad en los procesos de aprendizaje dentro del grupo experimental, descubriendo con ello, que es acertado acudir a estrategias innovadoras cuando se trata de mejorar los conocimientos de los futuros Ingenieros.

Pregunta Quinta: A los estudiantes de los grupos a los que se les realizó el seguimiento, se les preguntó *¿Qué recomendación haría al docente de Programación en Computadores?* Obteniendo en el grupo experimental como respuestas las siguientes:

- *Está bien que se propongan herramientas de software libre, casi no vemos nada del tema, pues uno en la práctica es que comprueba la teoría que piensa ha comprendido bien.*
- *Yo pienso que hubiera aprendido más, si desde el principio el profesor nos hubiera enseñado sus propuestas con Blockly y el Chamilo.*
- *Que sigan implementando estas herramientas, así vamos a aprovechar más el curso, y será más fácil todo.*
- *Que gestionen para que el servicio de internet también mejore en la sala de programación. Porque vale la pena aprender en clase con estos recursos novedosos.*
- *Que siga trabajando con esas herramientas y las comparta con otros profesores.*

Dentro de la población perteneciente al grupo control, se encontraron posiciones como las siguientes:

- *Que se destine más tiempo para aclarar dudas.*
- *Que nos enseñara de una manera que fuera más fácil de entender las cosas.*
- *que propusiera más ejercicios prácticos y con ejemplos…*

- *Con video tutoriales o algo más para repasar fuera de clase.*
- *Una actualización del pensum, está como anticuado.*

Descartando las imprecisiones en la redacción de las opiniones de los alumnos pertenecientes a la población muestra[3], todos estos datos revelados con la aplicación del instrumento, ratifican que siguiendo las respuestas de los estudiantes pertenecientes al grupo control, no se descubre ninguna evidencia concreta de que la problemática detectada con el diagnóstico se haya superado. No se encuentran datos brindados desde el instrumento, que permitan pensar que las dificultades observadas al inicio del presente estudio, se venzan sin la intervención de estrategias que apoyen el acogimiento del Software Libre. Por el contrario, varios estudiantes pertenecientes al grupo experimental expresaron la necesidad de un Plan de Curso que de manera amplia (todo el tiempo), acudiera a herramientas como las empleadas al final del semestre. Por su parte, algunos alumnos del grupo control, guardan las esperanzas de una actualización en el currículo, pues expresan que lo ven "algo anticuado".

5. RESULTADOS

5.1 CUADRO DE DIAGNÓSTICO DE PROBLEMAS EN LOS CURSOS DE PROGRAMACIÓN DE COMPUTADORES EN INGENIERÍA Y OPORTUNIDADES DE USO DEL SOFTWARE LIBRE.

[3] Se trataron de traer las respuestas a este documento lo más literalmente posible, y para destacarlas en cada viñeta, se editaron en cursiva. Sólo se omitieron del original, problemas de ortografía.

Cuadro 1. Problemas y oportunidades de uso del software libre en Programación

Identificación del problema	**Descripción del problema**	**Fuente**	**Oportunidades de uso del SL**
Problemas de tipo académico Problemas proceso de enseñanza	Los estudiantes llegan con dificultades para · *"…identificar claramente un problema"* · *"…definir u ordenar y luego seguir un procedimiento para la solución de un problema."* En la clase de Programación de Computadores, · *"se acude aún a una enseñanza por imitación."* · *"algunos docentes resultan teniendo éxito en su curso, pero no sistematizan su experiencia para que sea empleada luego por otro profesor, o incluso, por él mismo tiempo después. En ocasiones la explicación no es suficiente".* · *"el profesor se dedica a hacer ejercicios en el tablero o pide ver la solución de un problema en un libro."* · *"el tema de la programación es bastante amplio, exigiendo que se deba profundizar extra clase."* · *"se queda uno con la impresión de que lo que está aprendiendo, poco aplica en la ingeniería que uno cursa".* · *"Se perciben problemas de motivación, sobre todo en los de ingenierías diferentes a sistemas; además el estudiante no encuentra interesante resolver problemas o ejercicios. Para él no resulta divertido".*	-Entrevistas a docentes -Entrevistas a estudiantes -Autorreflexión -Entrevistas a docentes -Entrevistas a estudiantes	Con **Blockly** un estudiante de programación puede concentrarse en la lógica. Puede encontrar una variedad de estructurar en forma de fichas que se pueden colocar juntos sin cometer errores, pues es como un rompecabezas. **Blockly** es posible arrastrar y soltar los distintos componentes de control, lógica, operaciones matemáticas, texto, listados y procesos para crear las operaciones deseadas, típicas de una clase de programación de computadores que luego puede ser exportada a lenguajes como JavaScript, Python o XML. Con esto el estudiante desarrolla su lógica de programación y a su vez descubre como se ve esos ejercicios en otros lenguajes modernos y de gran demanda.

Cuadro 1. (Continuación)

Identificación del problema	**Descripción del problema**	**Fuente**	**Oportunidades de uso del SL**

Problemas de tipo académico y Problemas en el proceso de enseñanza aprendizaje	• Los estudiantes llegan con problemas de motivación. (Villalobos Salcedo, 2009) • Es común hallar estudiantes insatisfechos con el aprendizaje de la programación por encontrarlo difícil y poco útil para sus estudios y vida profesional. (Bermúdez, 2007) • Se descuidan muchas veces las habilidades que deben adquirir, reduciendo los cursos a un recorrido de estructuras sintácticas de un lenguaje de programación. (Villalobos Salcedo, 2009) • El hecho de reescribir los algoritmos hasta ponerlos a punto es operativamente complicada cuando se trabaja con los elementos tradicionales como lápiz y papel, tiza y pizarrón, etc. (Moroni & Señas, 2005) • Entre un grupo es común apreciar diferencias de conocimientos entre los estudiantes (Moroni & Señas, 2005) • Generalmente se espera que quienes aprueben el curso lo hagan porque son capaces de realizar programas con un grado de dificultad mayor al que poseen la mayoría de estudiantes. (Martínez, 2005)	Revisión Bibliográfica http://www.chamilo.org/es	Con **Blockly** el docente de programación de Computadores puede crear ejercicios interactivos y gráficos animados con los que el estudiante propondrá soluciones mediante lógica; así el estudiante aprenderá programación respondiendo a los principales desafíos. Mediante **Chamilo LMS**, por su parte, es posible proporcionar toda la teoría de la materia y foros en los cuales los estudiantes y profesores pueden resolver dudas y problemas presentes en los ejercicios.

Fuente: Elaboración propia

5.2 MATRIZ QUE CONFRONTA RECURSOS DEL SL Y EL DIAGNÓSTICO REALIZADO A LOS CURSOS DE PROGRAMACIÓN DE COMPUTADORES

Cuadro 2. Métodos o herramientas considerados

Fuente: Elaboración propia

5.3 PLAN DE CURSO REFORMADO DE LA MATERIA PROGRAMACIÓNDE COMPUTADORES EN PROGRAMAS DE INGENIERÍA

PRESENTACIÓN O JUSTIFICACIÓN DE LA ASIGNATURA

La asignatura Programación de Computadores, permite entender la estructura y funcionamiento de algún lenguaje de programación de alto nivel, empleando las técnicas de programación, desarrollo de funciones, y manejo de datos compuestos para proponer soluciones informáticas.

COMPETENCIA QUE DESARROLLA

- Desarrollo del pensamiento lógico-espacial
- Desarrollo del pensamiento crítico-analítico
- Capacidad de aprender a aprender
- Desarrolla programas siguiendo las técnicas del ciclo de desarrollo, desde la identificación inicial del problema hasta la obtención de un prototipo
- Aplicación de un lenguaje de programación para proponer una solución a una problemática real.
- Reconocimiento de la importancia de la materia en la carrera de ingeniería

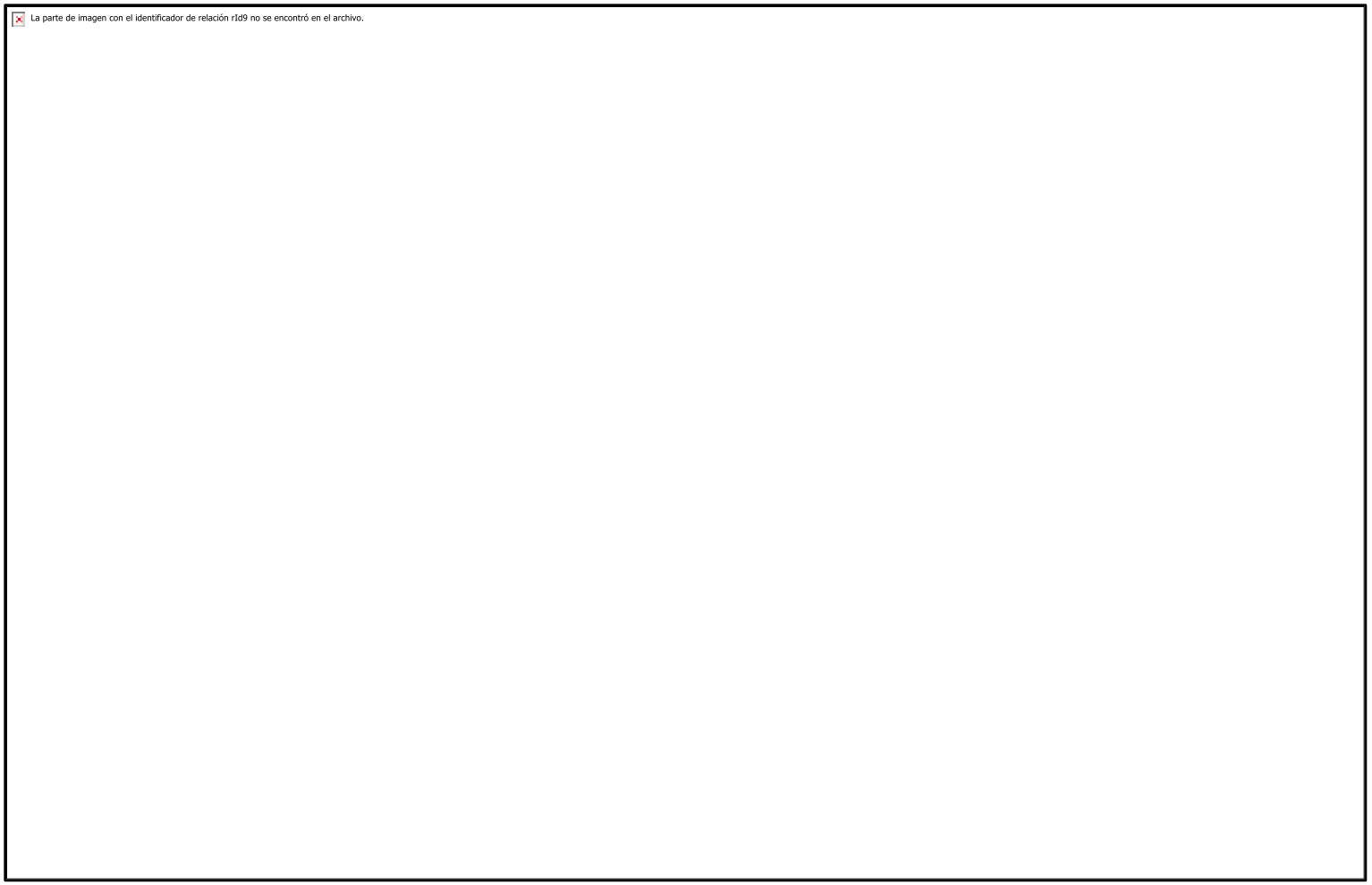
OBJETIVO GENERAL

Comprender la teoría y conceptuación de la programación orientada a objetos.

OBJETIVOS ESPECÍFICOS

- Implementar algoritmos en lenguaje de alto nivel y verificar su correcto funcionamiento.
- Utilizar correctamente los tipos de datos y las estructuras de almacenamiento proporcionadas por el lenguaje para construir soluciones eficientes.
- Desarrollar la habilidad lógica requerida para la programación de computadores.
- Utilizar adecuadamente el entorno y las facilidades de programación que ofrece el lenguaje de alto nivel.
- Aplicar elementos de la lógica de programación de computadoras en la resolución de problemas.

Cuadro 3. Planeador de la Asignatura por Contenidos



Cuadro 3. (Continuación)

Fuente: Elaboración propia

5.4 RESULTADOS DE LA IMPLEMENTACIÓN PARCIAL DEL PLAN DE CURSO REFORMADO DE PROGRAMACIÓN DE COMPUTADORES

Como se ha hecho notar a lo largo de los capítulos anteriores, para lograr implementar de manera parcial el Plan de Curso reformado, se llegó a definir una estrategia de trabajo. Y es precisamente con ella que el objetivo principal de esta idea, que giró en torno a proponer e implementar herramientas y métodos del campo del Software Libre para favorecer la enseñanza y el aprendizaje de la programación de computadores con orientación a la innovación en cursos de Ingeniería, se ha alcanzado.

Sin embargo, se hace necesario diseñar una propuesta de Plan de Curso que en realidad flexibilice la implementación gradual del Software Libre, siempre y cuando se permanezca atendiendo las necesidades de los alumnos; y con ello no se está refiriendo a la neta atención del docente, sino que se dispongan y estimulen los mecanismos de cooperación entre pares.

Otro resultado valiosísimo consiste en el reconocimiento que se dio con la implementación, al aprovechamiento de los equipos que portan los estudiantes. Es definitivo seguir apoyando el hecho de que los estudiantes usen sus *netbooks* y equipos celulares con tecnología Android, por ejemplo, pues eso se convierte en una ventaja a la hora de efectuar programaciones en la clase, apoyadas en herramientas Web. Muy seguramente en cualquier lugar en el que se encuentren, van a abordar la herramienta innovadora, van a recibir un mensaje o van a poder participar en un foro. Todo esto, reforzará al estudiante, y de paso a sus compañeros en un entorno interactivo y colaborativo, desde luego.

El docente pasa de ser un simple transmisor del saber, para jugar un papel de guía, facilitador y acompañante, de tal forma que los estudiantes abandonen su posición pasiva, rompan sus miedos e inseguridades, y se conviertan en protagonistas de su aprendizaje.

Al igual, aprovechar herramientas como Blockly y Chamilo LMS para definir y ordenar y luego seguir un procedimiento en la solución de un problema, no solo durante el tiempo de la clase sino desde el contexto del hogar, es relevante, dado que las prácticas del proceso de enseñanza aprendizaje de la programación no pueden seguirse abordando de manera tradicional, sino con propuestas innovadoras.

6. CONCLUSIONES

6.1. ESTADO EN QUE SE DEJA EL PROBLEMA DE IMPLEMENTACIÓN

Se diseñó una propuesta de intervención para el curso Programación de Computadores ofrecido en el primer año de las Carreras dispuestas en la Facultad de Ingeniería de la Universidad Antonio Nariño sede Neiva. La idea clara consistió en llevar a cabo dicha intervención, con el apoyo de recursos de software libre.

Uno de los objetivos acogidos con la implementación fue desarrollar la habilidad lógica requerida para la programación, y se hizo necesario recordar que para la Facultad de Ingeniería es de suma importancia fortalecer en sus estudiantes las destrezas que rodean al pensamiento lógico-espacial y el pensamiento crítico analítico, los cuales son fundamentales ante desafíos que hallarán en las diferentes asignaturas ofrecidas en cualquier Ingeniería, más en plena era digital, así como en el desenvolvimiento profesional del futuro ingeniero.

La cátedra de Programación de Computadores, no es ajena a estas necesidades, y se viene esforzando por favorecer el proceso de permanencia y continuación hacia los semestres superiores. Así que queda demostrado que las técnicas de programación, el desarrollo de funciones, y el manejo de datos, para proponer soluciones informáticas, exigen la incorporación de herramientas innovadoras en el aula.

Al revisar el diagnóstico realizado, se constató que era una oportunidad la de acoger a través de una acción educativa innovadora, el Software Libre. Entonces, se modificó la metodología de trabajo en el Plan de Curso de Programación de Computadores, y se dispuso la implementación parcial del mismo. Entre el análisis se aprecia que:

Se puede incorporar ejercicios y problemas en forma gradual usando la herramienta Blockly, atendiendo las dificultades manifestadas en el camino, por los estudiantes.

Con la implementación de Chamilo LMS, y también de Blockly, se atiende favorablemente la heterogeneidad de los grupos donde se experimentó el método, y se aprecia los grados reales de participación en clase o fuera de ella.

Las prácticas innovadoras favorecen la dialógica entre estudiante y docente, pero también entre pares, de manera que sean atendidas las diferencias, las propuestas y los aportes de todos, no solo en el espacio del aula, sino en otros contextos, generando mejores y mayores espacios comunicativos y, desde luego, de aprendizaje.

Las formas de enseñar y de aprender en plena era digital no pueden seguir siendo inmutables en la Universidad Antonio Nariño de la ciudad de Neiva, pues de ella toda una comunidad espera mayor adaptabilidad al ritmo acelerado de las TIC.

6.2. ESTADO EN QUE SE DEJA LA PREGUNTA E HIPÓTESIS DE INVESTIGACIÓN

Frente a la pregunta que dio origen al estudio, y que se formuló en los siguientes términos: ¿Cómo se pueden aprovechar los recursos que brinda el Software Libre para potenciar los procesos de enseñanza aprendizaje e innovación en el marco de los cursos de la Programación de Computadores en programas de ingeniería?, se puede concluir que son diversas y bastante prometedoras las opciones existentes, y que entre más avanza esta década, más se encuentran dispuestas en la red sus características y potencialidades.

Sin embargo, los paradigmas que se hace necesario vencer en cuanto a la metodología de enseñanza e incluso, de aprendizaje, son igualmente visibles. Los docentes de esta era debemos penetrar con formas más innovadoras y activas en nuestros roles; los estudiantes, por su parte, deben comenzar desde los primeros grados de su educación a vencer la pasividad, a ser más propositivos, y no esperar a que lleguen a la universidad para comenzar a fomentar habilidades y destrezas necesarias en el ciudadano digital.

6.3. RELEVANCIA DE LOS RESULTADOS DEL PROYECTO

La herramienta Blockly puede complementar los contenidos dispuestos en el Plan de Curso de Programación de Computadores, fomentando la acción de programar, gracias a un sistema con bloques, como si se tratara de armar un rompecabezas. Chamilo LMS por su parte, trasciende el contexto del aula, presentándose entre estas dos herramientas, ciertas características con las cuales la materia se fortalece en su proceso de enseñanza aprendizaje:

a. Admitirse en Chamilo LMS diferentes tipos de archivos: Texto, imagen, audio, video, animaciones, enlaces a otros documentos o a sitios diversos de la Web. Es decir, brillante por la inclusión de textos mutimediales.

b. Fácil empleo; versión mejorada: A diferencia de Moodle, Chamilo LMS posee una interfaz de usuario sencilla, para ser utilizada de manera que su empleo sea al alcance de todos. Vale la pena recordar que se han elegido herramientas recientes; Chamilo, por su parte, recoge las experiencias de su antecesor inmediato que es Dokeos, y esta a su vez, lo había hecho sobre Claroline. Igual sucede con Blockly. Blockly se ha inspirado en Lego, en Scratch, y en App Inventor.

c. Organización cronológica y temática de contenidos: La herramienta Chamilo LMS permite a futuro, que los contenidos del curso sean organizados por su docente.

d. Promueve la participación: Acoger herramientas como estas, permiten a mediano plazo la participación creciente de los estudiantes y de sus docentes. Lamentablemente ha imperado por décadas el tradicionalismo, y esta ventaja que es propia de tendencias pedagógicas donde la acción es un ingrediente fuerte, no es fácil motivar, ni lograr ver resultados a corto plazo.

6.4. APORTE DEL PROYECTO AL ESTADO DEL ARTE

Más allá de las debilidades enfrentadas, este estudio tuvo la virtud de haber alcanzado una aproximación a lo existente en cuanto a la enseñanza de la programación de computadores en Colombia y el mundo, al uso del Software Libre como estrategia educativa , y a la innovación en la enseñanza del mismo. Esta descripción ubicada como parte del marco referencial permite destacar a cualquier lector interesado que si bien es cierto en el país se comienza a hablar del aprovechamiento del Software Libre, falta mucho trecho por recorrer. Además, es evidente que en un tema que avanza a pasos agigantados como es propio del mundo digital, se hace irrefutable su estudio e implementación. Por ello, el acercamiento relatado en este documento, hace parte de los hitos que en la región surcolombiana se tejen alrededor de las TIC.

Puede considerarse como un definitivo aporte al estado del arte, el hecho de partir de un diagnóstico con el que se detectaran los problemas del proceso académico vividos en el curso de Programación de Computadores en la facultad de Ingeniería de la Universidad Antonio Nariño - sede Neiva, para terminar reconociendo que lo descubierto es proporcionable a lo que puede llegar a percibirse en cualquier otra localidad con condiciones similares a la de la región huilense. Sin embargo, por tratarse de una investigación de corte cualitativo, no acude a medidas estadísticas

con las cuales se universalicen los resultados, y por tratarse de un estudio sin precedentes precisos, no es viable pensar en la discusión de los mismos.

Finalmente, se esbozaron las posibles herramientas que en el mundo del software libre se pudieran aprovechar para la enseñanza aprendizaje de los contenidos previstos en la materia de Programación de Computadores, por lo cual, el estudio no se limitó a quedarse en el diagnóstico ni con dicha pesquisa de recursos, sino que se propuso una implementación parcial del Plan de Curso reformado, con el ánimo de vincular dos seleccionadas herramientas del software libre con las que se atendieran los problemas detectados por el diagnóstico.

Además, esta propuesta, puede ser un interesante aporte al mundo académico, en vista de que con ella se trasciende hasta el aprovechamiento de la tecnología móvil, la cual hace parte de la cotidianidad de los jóvenes de hoy. No se conforma el estudio con mostrar las bondades de la implementación de estrategias a través de simples PC, sino que la propuesta era precisamente acoger herramientas que se permitieran trabajar desde las Tablet y los celulares que hacen parte del diario vivir de los estudiantes actualmente.

7. RECOMENDACIONES

7.1 ASPECTOS DEL PROBLEMA DE INVESTIGACIÓN QUE QUEDAN PENDIENTES

En un principio se había definido que la problemática percibida por el docente de la asignatura de Programación de Computadores, se daba en torno de tres ejes: El docente de Programación de Computadores, el estudiante de dicho curso, y los problemas relacionados con los recursos innovadores y el acogimiento de software libres para el proceso enseñanza aprendizaje. Sin embargo, ante dicha apreciación, sigue palpable para futuros estudios, la problemática que rodea a la asignatura, en cuanto al docente, puesto que en él reposa gran parte de la iniciativa para abandonar actitudes propias del tradicionalismo, con tal de que su clase resulte motivante, especialmente cuando se trata de una materia vinculada esencialmente con la tecnología.

Aún se debe seguir persistiendo para que en el aspecto educativo los docentes y las instituciones a las que pertenezcan, tomen la decisión de enseñar a programar con lenguajes y herramientas libres, y se enfatice en los beneficios y en el trabajo en comunidad que es uno de las características fundamentales del software no privativo.

La necesidad que se le demanda al docente de cursos como el de Programación de Computadores, sigue siendo motivo de serias investigaciones con las que se implementen y ofrezcan estrategias para que el profesor perfeccione sus propuestas de diseño curricular con el que se alcance a manipular la realidad a través de la tecnología, considerando importante su rol para el éxito académico, la disminución de las tasas significativas de deserción, cancelación y reticencia del programa, y la consecuente efectividad de los futuros profesionales.

Sin lugar a dudas el tema de la innovación no se queda en los equipos tecnológicos ni en las simples herramientas que se renuevan de manera acelerada y sorprendente; este tema trasciende hasta el punto en que se reconoce que es en los enfoques pedagógicos en donde se deben concentrar los mayores esfuerzos para lograr ser ciudadanos verdaderamente creativos y desafiantes en la era mediática que nos corresponde vivir y preparar, si se permite, para las futuras generaciones que están en nuestras manos.

7.2 ASPECTOS DE LA PREGUNTA DE INVESTIGACIÓN QUE QUEDAN PENDIENTES

Frente a la necesidad de la asignatura Programación de Computadores en los pregrados de Ingeniería ofrecidos por la Universidad Antonio Nariño sede Neiva, se había definido el respectivo problema que motivó el proyecto de investigación, formulando como interrogante el siguiente: ¿Cómo se pueden aprovechar los recursos que brinda el Software Libre para potenciar los procesos de enseñanza aprendizaje e innovación en el marco de los cursos de la Programación de Computadores en programas de ingeniería?

Como parte del ejercicio investigativo, se hace indispensable a esta altura del estudio, reconocer que solo queda pendiente por formalizar el mencionado aprovechamiento, llevando a cabo ya no una implementación parcial, sino una intervención de principio a fin del curso, de las herramientas innovadoras que ofrece el Software Libre. Implementación total que desde luego permitiría al investigador llenarse más de argumentos para expresarle al mundo académico los momentos precisos del Plan del Curso en el que X o Y recurso de Software Libre, brinda el mejor de sus apoyos. Además, así como no sobra aspirar a fortalecer la implementación del Plan de Curso de manera global, se puede anhelar un fortalecimiento del trabajo en equipo hasta llevar a los estudiantes a la conformación de grupos de estudio o redes de colaboración.

7.3 DIFICULTADES QUE SE PRESENTARON EN EL PROYECTO Y MODOS EN QUE SE PODRÍAN RESOLVER EN FUTUROS TRABAJOS

El único impedimento tangible para que de manera sincrónica un buen número de estudiantes y cada uno desde su equipo PC, portátil, Tablet o celular, lograra entusiasmarse con las herramientas innovadoras seleccionadas, no estuvo en manos de quien dirigió el proyecto, pues se relacionó directamente con algo que los estudiantes mencionan en sus entrevistas: la velocidad del Internet. Lo que sucede, solo para hacerse el lector una idea concreta de la dimensión del problema, es que Neiva enfrenta una seria dificultad porque la empresa Electrificadora del Huila no ha facilitado la postería para instalar la fibra óptica, y por lo tanto la capital huilense no se incluye ni entre los actuales ni entre los próximos municipios beneficiados con el proyecto de fibra óptica nacional. Es decir, los estudiantes no cuentan ni en sus casas ni en espacios públicos, ni en sus establecimientos escolares, una tecnología lo suficientemente robusta y dinámica, tan apropiada para aprendizajes en línea como los previstos con este proyecto. Por supuesto, la solución de esta situación de amenaza, no está en manos de la academia, ni de los investigadores.

7.4 CÓMO SE PODRÍA CONTINUAR LA INVESTIGACIÓN EN FUTUROS TRABAJOS

Se puede pensar en trabajos futuros con los cuales se revise la posibilidad de institucionalizar el aprovechamiento del Software Libre en el Curso de Programación de Computadores, que ofrece la Universidad Antonio Nariño sede Neiva. Pero también, ante la oportunidad de poder continuar con la presente investigación, un futuro trabajo podría enmarcarse en la propuesta de estrategias prácticas con las que el docente que visualice la oportunidad de apoyarse en recursos brindados por el Software Libre, pueda dedicar más tiempo al diseño estratégico y metodológico de sus clases, ser ese facilitador del aprendizaje, y se aleje de la metodología tradicional que cada día pierde más vigencia.

Otra manera de darle continuidad al presente trabajo investigativo, es con la elaboración de rigurosas guías para el docente universitario, con las que él logre perfeccionar sus propuestas de diseño curricular que a su vez se encuentren enmarcadas en la realidad tecnológica. Es decir, un estudio que contemple la importancia de su rol para el éxito académico, la disminución de las tasas significativas de deserción, cancelación y reticencia del programa, y la consecuente efectividad de los futuros profesionales.

Posiblemente otra forma de proyectar el presente estudio, giraría en torno al tema de la innovación que dentro de los enfoques pedagógicos, rescate y acoja los equipos tecnológicos de alta gama con los que llegan los estudiantes a clase. Sería interesante concentrar mayores esfuerzos investigativos para potencializar y medir, si fuera posible, el impacto favorable que pueden tener los celulares y las Tablet en las clases universitarias.

REFERENCIAS BIBLIOGRÁFICAS

ABET Junta directiva. (12 de 06 de 2012). *Comisión de Acreditación de Ingenierías.* ABET.

ACM. (2008). *Ciencias de la Computación, Revisión Provisional del CS 2001.* IEEE Computer Society.

Adell, J., & Bernabé, I. (2007). *Software libre en educación.* Obtenido de http://elbonia.cent.uji.es/jordi/wp-content/uploads/docs/Software_libre_en_educacion_v2.pdf

Alfonso R., M. A., & Segnini R., J. (junio de 2009). *Repositorio Universidad de Oriente*. (U. d. Oriente, Editor) Recuperado el 24 de 10 de 2013, de (2009). Desarrollo de un sistema automatizado bajo entorno web para el control de la programación académica en la Universidad de Oriente núcleo de Anzoátegui: http://ri.biblioteca.udo.edu.ve/bitstream/123456789/1098/1/Tesis.DESARROLLO%20DE%20UN%20SISTEMA%20AUTOMATIZADO%20BAJO%20ENTORNO%20WEB.pdf

Anaya, R. (2006). Una visión de la enseñanza de la Ingeniería de Software como apoyo al mejoramiento de las empresas de software. *Revista Universidad Eafit ISSN: 0120-341X ed: Editorial Universidad Eafit*, 9.

Araoz Cajiao , O. L., Caballero Villalobos, J. P., González Neira, E. M., & García Cáceres, R. (2013). Solvers comerciales: ¿La mejor alternativa para la enseñanza? *Educación en ingeniería*, 84-93.

Arocha, J. V. (2008). *Videojuego Educativo para la enseñanza de la Arquitectura del Computador e Introducción a la Programación apoyadas en Software Libre.* Obtenido de http://eliascisneros.files.wordpress.com/2009/03/tesina-de-investigacion-_v3_.pdf

Asturizaga, D. (19 de 10 de 2011). *Chamilo*. Obtenido de http: //www.chamilo.org/es/experiencias-es-chamilo

Attwell, G. (2007). Entornos de aprendizaje personales para crear, consumir, remezclar y compartirActas del 2 º Taller Abierto TENCompetence, Instituto de Cibernética Educativos (pp. 36-41). *En servicio Enfoques orientados e Infraestructuras de Desarrollo de Competencias para toda la vida*, (págs. 36-41).

Benito, M., García, F., Portillo, J., & Romo, J. (2007). Descripción y características del concepto Nativo Digital. En M. B. otros, *Nativos digitales y modelos de aprendizaje* (págs. 2-3). Universidad de País Vasco / Euskal Herriko Unibertsitatea (UPV/EHU).

Bermúdez, C. P. (2007). *Aprendizaje de Programación de Computadores y Resolución de Problemas en un Ambiente de Trabajo en Colaboración.* Bogotá: Universidad de los Andes.

Bitocchi, O. S. (21 de junio de 2013). *Blog Universidad Privada del Norte*. Obtenido en línea de http://blogs.upn.edu.pe/ingenieria/2013/06/21/software-libre-la-alternativa-contra-la-pirateria-en-el-sector-educativo/

Bourque, P., Dupuis, R., & Abran, A. (1999). The Guide To The Software Engineering Body Of Knowledge - 2004 Version. *IEEE Software*.

Briet, D. s. (2006). *Un reto en la Formación del Profesorado.* Obtenido de http://www.slideshare.net/tanis0812/software-libre-tatiana-maldonado

Buendía, L., Colás, M., & Hernández, F. (1998). *Métodos de Investigación en Psicopedagogía.* Madrid: McGraw-Hill.

Camargo, E. H. (2006). *Los Institutos Tecnológicos Regionales: Educación Técnica Superior para la Provincia Mexicana.* Durango, México: Instituto Tecnológico de Durango.

Cañadas, J. F. (2005). *Jornadas Sobre Influencia De Las Nuevas Tecnologías De La Información y La Comunicación En El Campo Docente En Es.* Almería, España.

Carrasco, J. (2000). *Cómo aprender mejor. Estrategias de aprendizaje .* Madrid: Ricalp.

Castañeda, M. L. (2011). Tecnologías digitales y el proceso de enseñanza aprendizaje en la secundaria. (http://tesis.romocastaneda.es/Tomol.pdf, Ed.) Madrid, España: Universidad Nacional de educación y a Distancia.

Castaño, C., Maiz Olazabalaga, I., Palacio, G., & Villaroel, J. D. (2008). Prácticas educativas en entornos web 2.0. Madrid: Síntesis, 2008.

Castillo Ochoa, E., & González Bello, E. O. (2012). Comunicación y difusión de la innovación educativa: El desarrollo tecnológico de las instituciones de educación superior. En G. d. Gervasi, *Memorias XXIV Encuentro Nacional de la AMIC.* Saltillo, México: ISBN: 978-607-95511-2-4.

Ceja, M. (2000). Desarrollo de software educativo. *Revista tecnológica educativa. Barcelona*.

Cebrián de la Serna, M. (2004). Diseño y producción de materiales didácticos por profesores y estudiantes para la innovación educativa. In *Tecnologías para la educación: diseño, producción y evaluación de medios para la formación docente* (pp. 31-46). Alianza Editorial.

Cenatic, E. (2009). *Estudio sobre la situación actual del Software de Fuentes Abiertas en las Universidades y Centros I+D españoles.* Obtenido de http://observatorio.cenatic.es/phocadownload/informes/informe_universidad.pdf

Cisneros Arocha, J. V., & Cisneros Arocha, E. O. (2008). *Videojuego Educativo para la enseñanza de la Arquitectura del Computador e Introducción a la Programación apoyadas en Software Libre.* Caracas.

Contreras, Á. (2013). *Software Libre Educativo5dx.* Obtenido de http://es.scribd.com/doc/134317655/Software-Libre-Educativo5dx

Cuesta Benjumea, C. (2003). El Investigador Como Instrumento Flexible de la Indagación. *Universidad de Alberta. Instituto Internacional de Metodología Cualitativa*.

Chumba, R. H. (2009). El aprendizaje colaborativo y la deserción escolar en la licenciatura en Contaduría y administración del centro de Estudios Superiores CTM. Mérida de Yucatán.

Del Bianco, V., Lavazza, L., Morasca, S., & Taibi, D. (2011). *Calidad de software de código abierto: la honradez Modelo QualiPSo.* Florencia, Italia: Università degli Studi dell'Insubria, Dipartimento di Informatica.

Dolado, J. J. (2004). *"Los créditos 330 software y la ingeniería DEL.".*

Dschultz, C. (07 de 12 de 2013). *Citizen4: A citizen's blog about Champaign Unit 4*. Obtenido de http://thecitizen4blog.wordpress.com/2013/12/07/hour-of-code/

Duque, M. (2009). Ciencia, tecnología e innovación en ingeniería, como aporte a la competitividad del país. *Cómo formar ingenieros para la innovación* (pág. 50). Bogotá: Universidad de los Andes.

Duque, M. (2009). Reunión Nacional y Expoingeniería ACOFI 2009. *Formar ingenieros innovadores es solo una parte del camino hacia la innovación* (págs. 108-115). Bogotá: Universidad de los Andes.

Ezpeleta, J. (2004). Inovaciones educativas: Reflexiones sobre los contextos en su implementación. *Revista mexicana de investigación educativa*, 403-423.

Fernández, W., Larraz, P., & Silvera, E. (2009). *Proceso de Selección de Software.* Montevideo: Universidad ORT Uruguay.

GNU Operating Sistems. (s/f). Obtenido de ¿Por qué las instituciones educativas debe usar y Enseñe a Software Libre?: http://www.gnu.org/education/edu-why.es.html

Gómez-Albarrán, M. (2005). La enseñanza y el aprendizaje de la programación:. Un estudio de las herramientas de apoyo de software. *The Computer Journal , 48 (2)*, 130-144.

González Mariño, J. (04 de 2006). B-Learning utilizando Software Libre, una alternativa viable en Educación Superior. *6° Congreso Internacional Retos y Expectativas de la Universidad "El papel de la universidad en la transformación de la sociedad"*. Ciudad Victoria, México: UNIVERSIDAD AUTONOMA DE TAMAULIPAS .

González Mariño, J. (s/f). B-Learning y Software Libre, una Alternativa para la Innovación de la Educación en el Contexto de las Sociedades del Conocimiento . *Universidad Autónoma de Tamaulipas* .

González, J. (2003). *Software Libre.* Madrid: Grupo de Sistemas y Comunicaciones, Universidad Rey Juan Carlos.

Granollers, T., Lorés, J., & Perdrix, F. (2002). Modelo de proceso de la Ingeniería de la Usabilidad. *COLINE*, 11-12.

Hernández, J. M. (2005). *Software libre: técnicamente viable, económicamente sostenible y socialmente justo.* Barcelona: Zero Factory S.L.

Hernández, R. F. (2006). *Metodología de la Investigación.* México: Mc Graw Hill.

Jaque, B. Miguel. (2009). Prólogo de Estudio sobre la situación actual del Software de Fuentes Abiertas en las Universidades y Centros I+D españoles. ISBN-13: 978-84-692-9552-6. [en línea] http://observatorio.cenatic.es/phocadownload/informes/informe_universidad.pdf

Jaques, R. (24 de 02 de 2012). La calidad del código fuente abierto es tan bueno como el software propietario. *The Inquirer*.

Jiménez, J. &. Otros (2005). *Colombia Aprende.* Obtenido de Software Libre en la educación: http://www.colombiaaprende.edu.co/html/mediateca/1607/articles-108475_archivo.pdf

Jonassen, D. (2000). *El diseño de entornos constructivistas de aprendizaje: Diseño de la instrucción. Teoría y modelos.* Madrid : Aula XXI Sntillana.

Long, J. (2009). Open Source Software Development Experiences on the Students' Resumes: Do They Count? *Journal of Information Technology Education, 8*, 229-242.

Maggio, M. (2002). *El Tutor en la Educación a Distancia.* Buenos Aires: Amorrortu Editores, S. A.

Malaver, N., Rey, C., & Rodríguez, J. (2011). Enseñanza de programación en el Politécnico Grancolombiano. Situación actual y aplicación de TIC como alternativa de mejora. *open Journal Systems Vol 1, No 1*.

Manuel Ruiz González, E. M. (1989). *La innovación tecnológica y su gestión.* Barcelona: Marcombo.

Marqués, P. (1995). *Metodología para la elaboración de software educativo en SoftwareEducativo. Guía de uso y metodología de diseño.* Barcelona: Estel.

Martínez Sánchez, F., Cebreiro López, B., Prendes Espinosa, M., Roig Vila , R. I., Solano Fernández , I. M., Chacón Rivas , M., . . . Vargas Calderón , S. (2008). *Implementación de las TIC en los programas académicos del Instituto Tecnológico de Costa Rica.* Murcia.

Martínez, Y. (2005). *En busca de una nueva forma de enseñar a programar*. Obtenido de http://www.mty.itesm.mx/rectoria/dda/rieee/pdf-05/28%28DTIE%29.YolandaMtz..pdf

Mertens, D. C. (2005). *Research and evaluation in Education and Psychology: Integrating diversity with quantitative,and mixed methods qualitative. .* Thousand Oaks: Sage.

Ministerio de Educación de Chile. (09 de agosto de 2012). *¿Sabes qué es la edución técnica?* Obtenido de MiFuturo: http://www.mifuturo.cl/index.php/media-conoce-tus-opciones/sabes-que-es-la-educacion-tecnica

Mirás, N. (20 de 10 de 2012). El software libre ahorró a Brasil 225 millones de dólares en el 2010. *la voz de Galicia.es*. Obtenido de http://www.lavozdegalicia.es/noticia/vidadigital/2012/10/20/software-libre-ahorro-brasil-225-millones-dolares-2010/0003_201210G20P27991.htm

Montserrat Martínez, A. d. (19 de 12 de 2011). *Chamilo LMS*. Obtenido de http://www.chamilo.org/es/experiencias-es-chamilo

Moroni, N., & Señas, P. (2005). Estrategias para la enseñanza de la programación . *JEITICS*, 254-269.

Naula Daquilema, S. M. (2011). Software Educativo como Herramienta en el Proceso Enseñanza Aprendizaje en el Área de Lengua y Literatura de los Estudiantes del Quinto Año de Educación Básica de la Unidad Educativa “Nación Puruha” de la Cooperativa AgrícolaGalte Laime perteneciente a la. *UNIVERSIDAD ESTATAL DE BOLÍVAR Guaranda, Ecuador*, 28.

Ortega, A. B. (09 de 2013). Campus Virtuales de Software Libre en Universidades españolas. *PROYECTO FIN DE MÁSTER* . Murcia: Universidad de Murcia.

Osorio Alvarado, B. (2012). *Desarrollo de jego interactivo para facilitar el parendizaje, retención de información y motivación de estudio en los niños de primaria y educación básica.* Guatemala.

Osorio Alvarado, J. (2012). Desarrollo del juego interactivo para facilitar el aprendizaje, retención de la información, y motivación de estudio en los niños de primaria y de educación básica. Guatemala: Universidad de San Carlos de Guatemala.

Prendes Espinosa, M. P. (2009). Plataforma de Campus Virtual de Software Libre*: Análisis comparativo de la situación actual de las universidades españolas.* Madrid: P r o y e c t o EA-2008-0257d.

Proyecto ABET- ESPOL. (20 de 11 de 2013). *Proyecto ABET- ESPOL*. Obtenido de Proyecto de Acreditación Internacional de Programas de Ingeniería: http://www.abet.espol.edu.ec/inicio/2-ique-es-abet.html

Rama, C. (2006). *Universidad de la Empresa (UDE), Uruguay.* Obtenido de La Tercera Reforma de la Educación Superior en América Latina: ude.edu.uy

Renés, P. (s/f). *Herramientas de comunicación virtual en educación superior: Innovación docente en los estudios de magisterior.* España: VIII Foro sobre Evaluación de la Calidad de la Investigación y de la Educación Superior.

Revista Andalucia Educativa .Del software libre al conocimiento libre, J. A. (2005). .Del software libre al conocimiento libre. *Revista Andalucia Educativa . No 51* , Universidad Jaume I de Castellón, p.5.

Rodríguez Gómez, G., Gil, J., & García Jiménez, E. (1996). *metodología de la Investigación Cualitativa.* Granada (España): Ediciones Aljibe.

Rodríguez, S. (2011). *Diseño de un laboratorio virtual para la docencia de la materia de tecnología en educación secundaria.* Almería (España): Recuperado el 13 de septiembre de 3013 de http://repositorio.ual.es/jspui/bitstream/10835/1321/1/Proyecto.pdf.

Rodríguez, S. (10 de 09 de 2013). Obtenido de Repositorio Universidad de Almería, España. En línea: http://repositorio.ual.es/jspui/bitstream/10835/1321/1/Proyecto.pdf

Romero, T. A. (2006). Moodle, Unimos Mentes, Creamos Conocimiento Libre. *Ponencia presentada al VI Congreso Internacional Virtual de Educación*. Islas Baleares: CIVE.

Rosa, F. d., & Heinz, F. (2007). *Guía práctica sobre software libre: su selección y aplicación local en América Latina y el Caribe.* Montevideo, Uruguay: UNESCO publication.

Sabino, C. (2002). *El proceso de investigación.* Caracas: Panapo.

Salgado Lévano, A. (2007). Investigación cualitativa: diseños, evaluación del rigor metodológico y retos. *LIBERABIT: Lima (Perú) 13*, 71-78.

Salinas, J. (2004). Los recursos didácticos y la innovación educativa. *Comunicación y Pedagogía*, 36 - 39.

Serrano, J. E. (2005). *Sistema de Aprendizaje Virtual Interactivo, SAVIO.* Obtenido de http://www.umanizales.edu.co/publicaciones/campos/ingenieria/ventana_informatica/html/ventana12/articulo10.pdf

Shockey, K. &. (2005). Using open source to enhance learning. In Information Technology Based Higher Education and Training., (págs. pp. F2A-7).

SLEC. (2012). *Software de Libre Redistribución y Educación en Colombia.* Obtenido de http://www.slec.net/

Sotomayor, D. A. (2013). *Somos Libres*. Obtenido de http://www.somoslibres.org/modules.php?name=News&file=article&sid=258

Stallman, R. (1999). *El sistema operativo GNU y el movimiento de software libre de fuentes abiertas: voces desde la revolución de código abierto.* Obtenido de http://www.gnu.org/education/edu-why.html

Stallman, R. (2004). *Software libre para una sociedad libre.* Madrid: Traficantes de Sueños.

Stallman, R. (2004). *Software Libre para una sociedad libre.* Madrid: Traficantes de sueños ISBN: 84-933555-1-8.

Stallman, R. (07 de 2013). *Por qué utilizar y enseñar software libre*. Obtenido de http://www.fsf.org/resources/

Stallman, R. (s/f). *Sas Libre*. Obtenido de http://saslibre.com/index.php/bases-del-software-libre

Suryn, W., Bourque, P., Laporte, C., & Abran, A. (2002). Medición de la calidad del producto de software prácticas de calidad y evaluación utilizando TL9000 e ISO / IEC 9126. *Tecnología y Software Engineering Practice*, 156-160.

Titlestad, O. y. (2005). Transformación de la Educación a través de enfoques de código abierto. *IRIS'05: Actas del Seminario de Investigación de Sistemas de Información 28 en Escandinavia .* Escandinavia.

Toledo Tovar , A., Cuellar Rivera, J. V., & Romero Mera, J. (2011). Análisis de modelos de evaluación de calidad de Software Libre. En U. P. Colombia,

Investigación en Ingeniería de Sistemas e Informática. Tunja, Boyacá: Grupo de Investigación en Software - GIS.

Tur Viñes, V. T., Orbea Mira, J., Fernández Poyatos, M. D., & Poveda Salvá, M. (2008). Utilización del software libre Moodle en la asignatura Creatividad Publicitaria 1. *Universidad de Alicante*.

UAN. (20 de 11 de 2013). *Universidad Antonio Nariño, sede Neiva.* Obtenido de http://www.uan.edu.co/neiva

Ulloa, G. (2010). ¿Qué pasa con la ingeniería en Colombia? *Ingeniería y sociedad*, 1-4. Obtenido de aprendeenlinea.udea.edu.co/revistas/index.php/ingeso/article/viewFile/7303/6742

Universidad Viña del Mar. (junio de 2012). *Programa de asignatura Ingeniería civil informática.* Obtenido de http://escuelas.uvm.cl/ingenieria/images/stories/programas/informatica/2/27_6_606_696242programacion.pdf

Universidad Pedagógica y Tecnológica de Colombia. (enero-diciembre de 2011). Memorias de Investigación en ingeniería de sistemas e informática. Tunja, Boyacá, Colombia: editada por el Grupo de Investigación en Software - GIS, Adscrito a la Escuela de Ingeniería de Sistemas y Computación, Facultad de Ingeniería de la Universidad Pedagógica y Tecnológica de Colombia.

Villalobos Salcedo, J. A. (2009). *Proyecto CUPI2 – Una solución integral al problema de enseñar y aprender a programar.* Obtenido de http://www.colombiaaprende.edu.co/html/mediateca/1607/articles-205832_recurso_1.pdf

Vizconde Veraszto, E., Franco de Camargo , J. T., Barros Filho , J., García García, F., Ferreira do Amaral , S., & Bortoloti , A. A. (2012). Producción de animaciones modeladas por ordenador utilizando software libre. *Icono No. 14*, 198-212.

Zapata Puerta, L. N., & Recaman Chaux, H. (2011). Sistema tutorial interactivo para la enseñanza y aprendizaje de algoritmos- EvaProg. *Memorias de Investigación en ingeniería de sistemas e informática*, 4-18.

Zorzoli, P. (2003). *Investigación sobre el Movimiento del Software Libre*. Obtenido de http://www.casanas.com.ar/artsAdj/Zorzoli_-_sobre_el_movimiento_del_swl.pdf

Zorzoli, P. L. (2002). *Investigación sobre el Movimiento del Software Libre.* Buenos Aires:

oleccion.educ.ar/coleccion/CD7/img/docs/acerca_soft/mod01/mod01_softwarelibrepz/7-sl_argentina.html.

ANEXOS

Anexo A. Instrumento Guía de preguntas Entrevista a Docentes

RECURSOS DEL SOFTWARE LIBRE PARA FAVORECER LA ENSEÑANZA, APRENDIZAJE E INNOVACIÓN EN CURSOS DE PROGRAMACIÓN DE COMPUTADORES EN INGENIERÍAS

ENTREVISTA A DOCENTES

Objetivo

Diagnosticar posibles problemas que se presentan en la enseñanza aprendizaje de la materia programación de Computadores con respecto a la innovación.

Información General

Nivel estudios: Preg. _____ Espec. _____ Mag._____ Ph. D. ____
Dedicación: Tpo. Cplto._____ Mdio.Tpo. _____ Cáted. _____
Programa: _____ Años de vinculación: _____

A continuación usted encuentra una serie de preguntas abiertas, las cuales deberá contestar con la mayor sinceridad posible.

PREGUNTAS

1. ¿Cómo percibe a los estudiantes de ingenierías diferentes a la ingeniería de sistemas, ante el curso de Programación de Computadores?

2. ¿Qué habilidades son para Usted esenciales fortalecer en los estudiantes de las diversas ingenierías mientras toman el curso de Programación de computadores?

3. ¿Ante qué tipo de problemas en el proceso enseñanza aprendizaje se ha encontrado, siendo docente de esta asignatura?

4. ¿A qué recursos tecnológicos específicamente orientados a la educación en programación de computadores y con qué intensidad o periodicidad Usted cree que deben acudir estudiantes y docente en las clases y fuera de ella?

5. ¿Ha pensado en cómo puede aprovecharse los recursos que brinda el Software Libre para potenciar los procesos de enseñanza aprendizaje de la programación de computadores en programas de ingeniería?

6. ¿Cuál lenguaje o tecnología considera que es la indicada para iniciarse en la programación?

7. ¿Conoce comunidades en donde se promueva el uso de software libre, participa en alguna?

Las personas que han sido entrevistadas tienen en común que se han desempeñado como docentes de la materia Programación de Computadores. Todos han conocido la propuesta de investigación desde sus inicios y han autorizado la publicación de sus respuestas. ¡Muchas gracias!

Ing. Miguel Ángel Tovar Cardozo

Anexo B. Instrumento Guía de preguntas Entrevista a Estudiantes

RECURSOS DEL SOFTWARE LIBRE PARA FAVORECER LA ENSEÑANZA, APRENDIZAJE E INNOVACIÓN EN CURSOS DE PROGRAMACIÓN DE COMPUTADORES EN INGENIERÍAS

ENTREVISTA A ESTUDIANTES

Objetivo

Diagnosticar posibles problemas que se presentan en la enseñanza aprendizaje de la materia Programación de Computadores con respecto a la innovación.

Información General

Programa_____ Semestre _____

A continuación usted encuentra una serie de preguntas abiertas, las cuales deberá contestar con la mayor sinceridad posible.

PREGUNTAS

1. ¿Cómo percibe a los estudiantes de ingenierías diferentes a la ingeniería de sistemas, ante el curso de Programación de Computadores?

2. ¿Qué habilidades son para Usted esenciales fortalecer en los estudiantes de las diversas ingenierías que llegan a su curso?

3. ¿Qué tipo de problemas en el proceso enseñanza aprendizaje ha llegado a detectar, mientras fue estudiante de esta asignatura, o ahora cuando ya ha pasado uno o más semestres de haberla tomado?

4. ¿A qué recursos tecnológicos específicamente orientados a la educación en programación de computadores y con qué intensidad o periodicidad Usted acude durante la clase?

5. ¿Ha pensado en cómo puede aprovechar los recursos que brinda el Software Libre para potenciar los procesos de enseñanza de la programación de computadores en programas de ingeniería?

6. ¿Cuál lenguaje o tecnología considera que es la indicada para iniciarse en la programación?

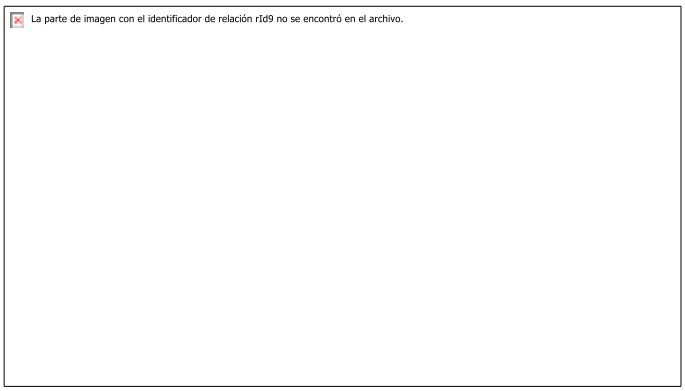
7. ¿Conoce comunidades en donde se promueva el uso de software libre? ¿Participa en alguna?

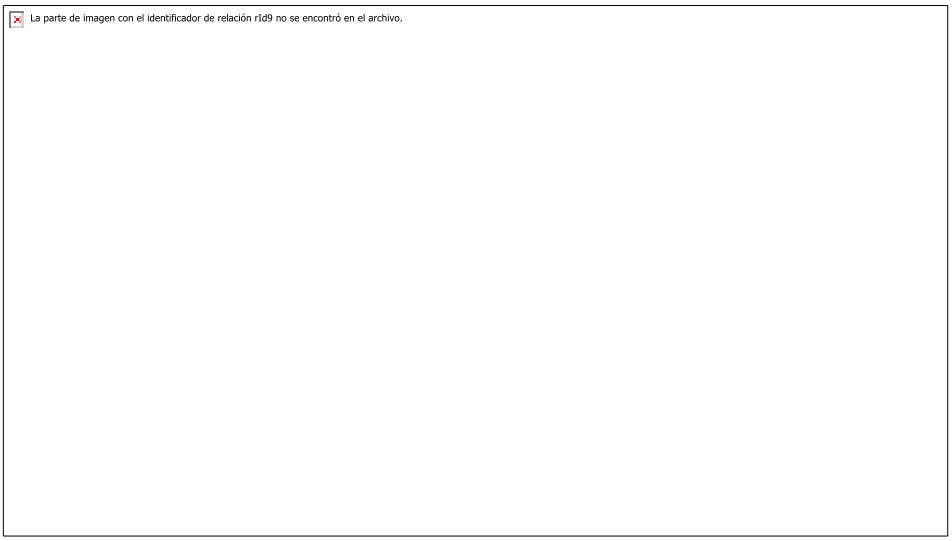
¡Muchas gracias!

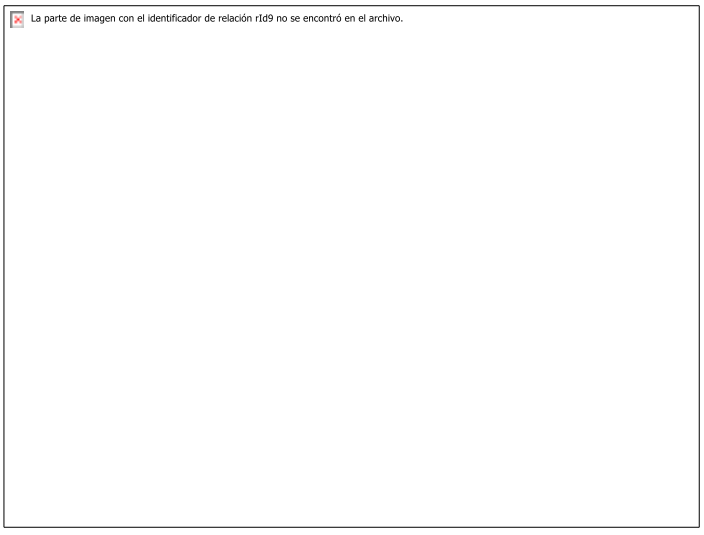
Observación: Las personas que han sido entrevistadas tienen en común que siendo estudiantes de diversas ingenierías, han tomado antes la materia Programación de Computadores. Todos han conocido la propuesta de investigación desde sus inicios y han autorizado la publicación de sus respuestas.

Ing. Miguel Ángel Tovar Cardozo

Anexo C. Cursos de Programación de Computadores (Población muestra)

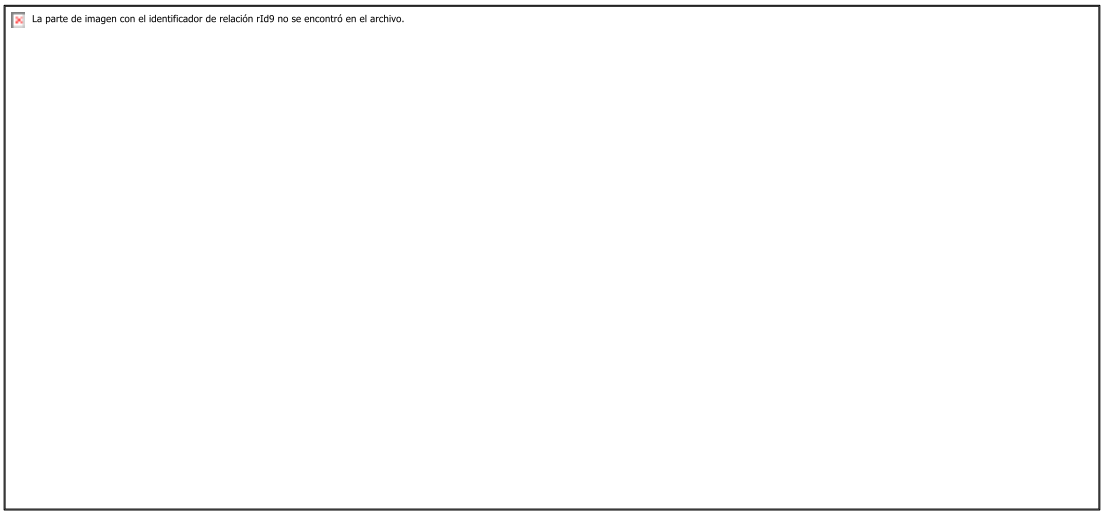


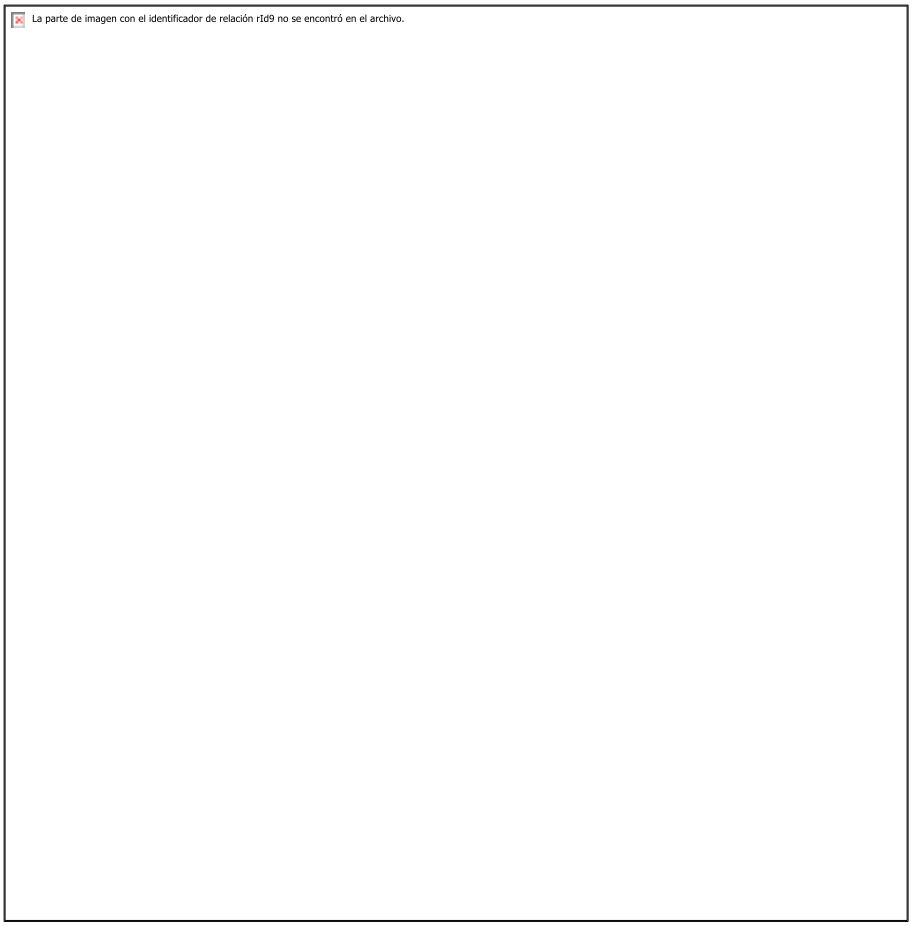




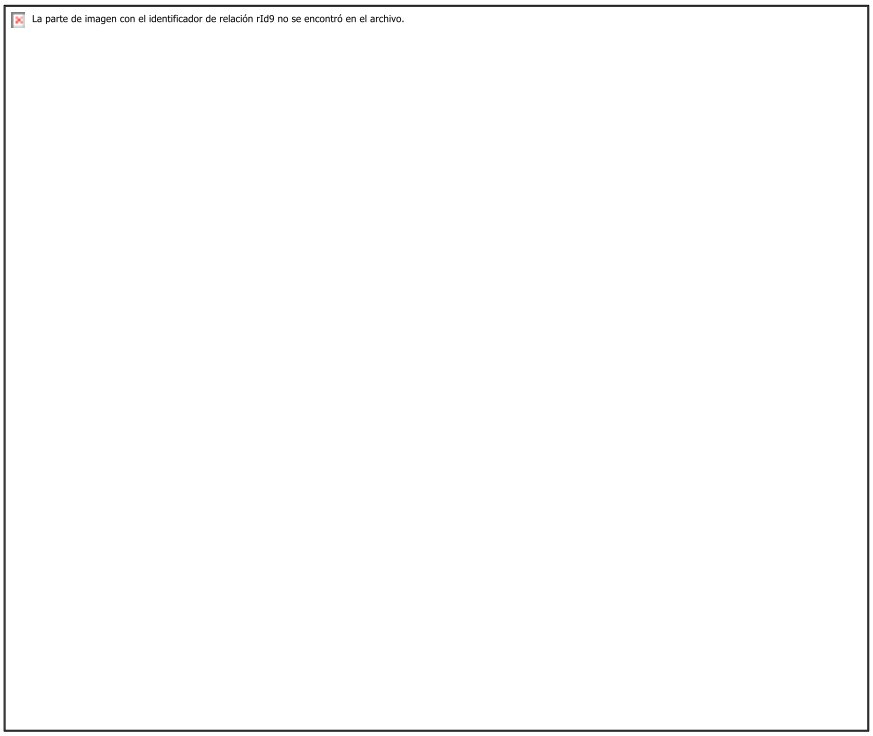
Fuente: Registro fotográfico del autor. Grupo experimental. Neiva, 2013

Anexo D. Selección estudiantes de Programación de Computadores



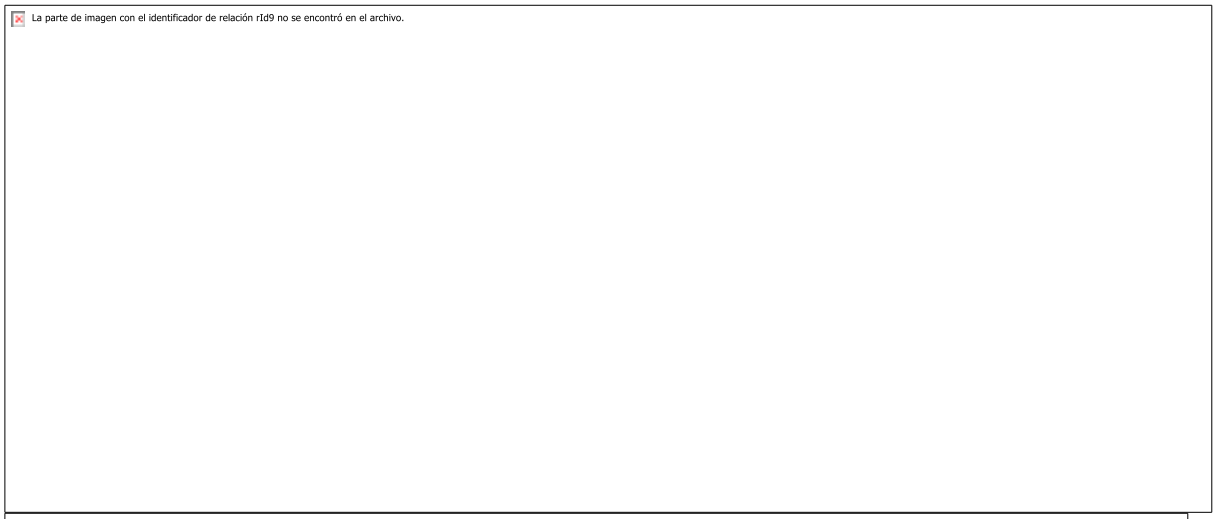


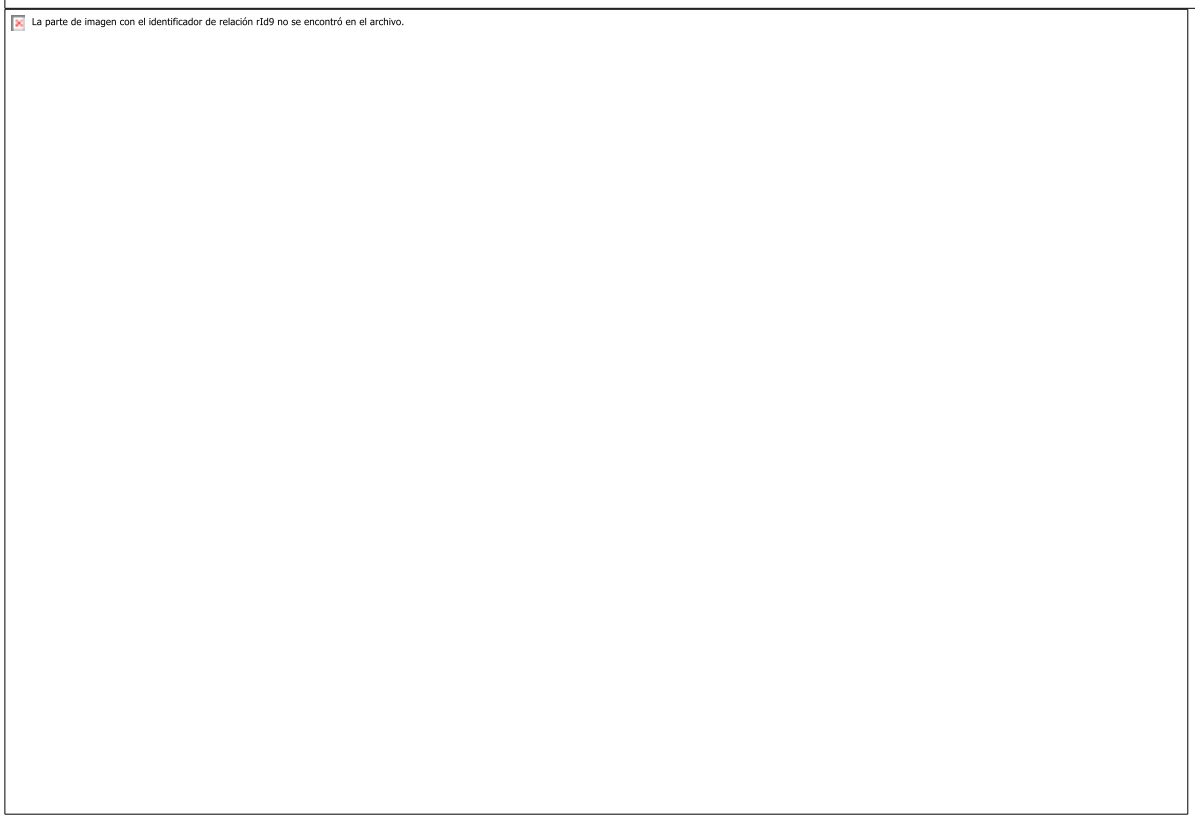
Fuente: Página Virtual de la UAN: http://virtual.uan.edu.co
Anexo D. (Continuación)



Fuente: Página Virtual de la UAN: http://virtual.uan.edu.co

Anexo E. Grupo control





Fuente: Página Virtual de la UAN: http://virtual.uan.edu.co

Anexo F. Instrumento Guía de preguntas Entrevista a Población Muestra

RECURSOS DEL SOFTWARE LIBRE PARA FAVORECER LA ENSEÑANZA, APRENDIZAJE E INNOVACIÓN EN CURSOS DE PROGRAMACIÓN DE COMPUTADORES EN INGENIERÍAS

ENTREVISTA A POBLACIÓN MUESTRA

Objetivo: **Recolectar información en los grupos (Experimental y Control)**

A continuación, Usted encuentra una serie de preguntas abiertas, las cuales deberá contestar con la mayor sinceridad posible.

PREGUNTAS

- ¿Siente que se apropió de los contenidos propuestos en el plan de curso? ¿Por qué?
- ¿Qué opina sobre el trabajo participativo alcanzado durante el desarrollo de la temática final?
- ¿Qué equipos tecnológicos llegó a emplear durante el curso? ¿Cuáles durante las clases y cuáles fuera de ella?
- ¿Cuál fue el proceso que más favoreció la comprensión del paradigma de la programación orientada a objetos?
- ¿Qué recomendación haría al docente de Programación en Computadores?

CONSENTIMIENTO INFORMADO

Yo, _____, voluntariamente acepto participar en la aplicación del cuestionario– encuesta, aplicado por el docente Miguel Ángel Tovar Cardozo, de la Universidad Antonio Nariño, sede Neiva. He recibido una explicación clara y completa sobre el carácter general y los objetivos de este instrumento y de la manera en que se utilizarán los resultados.

Reconozco que estoy en libertad de negarme a contestar cualquier pregunta. También se me ha informado que los datos que proporcione aquí serán empleados exclusivamente con fines investigativos y académicos; incluso, se me ha especificado que no hay necesidad de dar a conocer mi nombre y que no recibiré retribución económica por este hecho.

Nota del Autor: Las personas que han sido entrevistadas toman en la actualidad el Curso Programación de Computadores en la facultad de Ingeniería de la Universidad Antonio Nariño, sede Neiva. Algunos han conocido la propuesta de investigación desde sus inicios (grupo experimental), pero otros solo hasta esta fecha son conocedores oficialmente del estudio dirigido por quien entrevista. Se les solicita al final del cuestionario, la autorización para publicar posiblemente sus respuestas, exclusivamente con fines académicos.

¡Muchas gracias!

Ing. Miguel Ángel Tovar Cardozo