UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

# TESINA DE LICENCIATURA

**TÍTULO:** Studium: un sistema de gestión de aprendizaje que integra herramientas de enseñanza de programación en las aulas

**AUTORES:** Alan Martin Varela

**DIRECTOR:** Claudia Banchoff, Claudia Queiruga

**CODIRECTOR:**

**ASESOR PROFESIONAL:**

**CARRERA:** Licenciatura en Sistemas

## Resumen

En los últimos años han surgido numerosas políticas educativas federales, vinculadas con las habilidades relacionadas con las tecnologías digitales, con especial atención en la enseñanza de programación desde temprana edad en las aulas. Debido a esto es posible encontrar múltiples iniciativas que proponen el uso de herramientas para enseñar a programar utilizando el paradigma de programación visual basado en bloques. Estas herramientas están orientadas a diferentes niveles educativos y favorecen introducir los conocimientos iniciales de programación de forma intuitiva e interactiva a los estudiantes.
Si bien existen numerosas herramientas para desarrollar actividades de enseñanza de programación en las aulas, los docentes no cuentan con herramientas sistematizadas que permitan realizar un seguimiento de su uso. Los sistemas LMS/CMS son entornos que podrían acompañar al docente durante el proceso de enseñanza de programación en la escuela resultando un aporte relevante para la actividad. El objetivo de esta tesina es desarrollar un sistema LMS que acompañe la actividad docente en la enseñanza de la programación en el ámbito escolar, favoreciendo el uso de herramientas de programación visual basadas en bloques, popularmente adoptadas en la escuela.

## Palabras Clave

Enseñanza de programación, Sistema LMS, Sistema CMS, Chamilo, Google, Blockly Games, Aprendizaje en línea.

## Conclusiones

Se estudiaron e investigaron herramientas de enseñanza de programación basada en bloques popularmente utilizadas en las escuelas y sus posibilidades de integración a sistemas LMS/CMS. Se eligió la herramienta Blockly Games, entre varias opciones, para la integración propuesta en este trabajo debido a las funcionalidades ofrecidas y a las posibilidades de modificación para integrarse a un sistema LMS/CMS. Se relevaron y analizaron las características de varios sistemas LMS/CMS y se seleccionaron 3 sistemas que fueron instalados localmente y evaluados, teniendo en cuenta la funcionalidad ofrecida. Se seleccionó el sistema Chamilo para ser utilizado como base del desarrollo de Studium. Luego de analizarse distintas tecnologías para la integración, esta se implementó utilizando un servicio web SOAP para la comunicación entre ambas herramientas. QuickForm fue utilizado para gestionar los formularios dentro de Chamilo, y JQuery permitió hacerlos más dinámicos. Para ayudar a que la ejecución de los juegos de Blockly Games dentro de Studium fuera transparente para los usuarios, se modificaron las plantillas de visualización de la herramienta usando Composer Templates. Studium se instaló sobre un Servidor Web HTTP Apache para las pruebas durante el desarrollo. Se planificó y diseñó la evaluación de Studium para ser realizada por profesores y estudiantes ejecutando la funcionalidad implementada y contestando a un cuestionario con una escala de Likert.

## Trabajos Realizados

Análisis de herramientas de enseñanza de programación y elección de una para realizar la integración propuesta en este trabajo.
Análisis de sistemas LMS/CMS y elección de uno para utilizar como base para el sistema desarrollado en este trabajo.
Instalación e interpretación del código fuente del sistema LMS y la herramienta de enseñanza de programación seleccionados.
Selección y análisis de las tecnologías utilizadas durante el desarrollo del sistema propuesto.
Definición de Casos de Uso y Diagramas de Flujo para el sistema desarrollado.
Desarrollo del Studium integrando el sistema LMS y la herramienta de enseñanza de programación seleccionados.
Diseño de modelo de evaluación para la herramienta desarrollada.

## Trabajos Futuros

Extender la funcionalidad de Studium para obtener métricas sobre la resolución de cada juego de Blockly Games.
Implementar una galería en Studium para que los jugadores puedan compartir sus creaciones en los juegos de ”Tortuga”, “Película” y “Música” de Blockly-Games con otros estudiantes inscriptos en el curso.
Traducir la documentación ofrecida por Blockly Games para los juegos “Tutor de Estanque” y “Estanque” a idioma español.
Integrar otras herramientas de enseñanza de programación en las aulas a Studium.
Realizar la evaluación de Studium en las aulas.

**Fecha de la presentación:** Marzo 2020

# Índice de Contenidos

# índice de Figuras

# Índice de Tablas

# Capítulo 1. Introducción

## 1.1 Motivación

En los últimos años han surgido numerosas políticas educativas federales, vinculadas a la importancia que han adquirido las habilidades relacionadas con las tecnologías digitales en la formación ciudadana, y más específicamente las vinculadas con la enseñanza de la programación en la escolaridad obligatoria. Ejemplo de ellos son el proyecto "Program.AR"[1], el "Programa Conectar Igualdad"[2] y el "Plan Nacional Integral de Educación Digital" (PLANIED)[3], entre otros. En el año 2018 comenzó a dictarse la primera "Especialización docente en Didáctica de las Ciencias de la Computación" dirigida a docentes de nivel primario y secundario en la provincia de Buenos Aires, Córdoba, Neuquén y Rosario, comenzando a atender un área de vacancia en la temática en relación a la formación docente en un campo que aparece como nuevo en el sistema de educación obligatorio (Queiruga, Banchoff Tzancoff, Venosa, Gomez y Morandi, 2018) . Estas iniciativas demuestran la importancia dada a la enseñanza de programación en las aulas de las escuelas, desde temprana edad. La disponibilidad de materiales adecuados para acercar la programación a la escuela es otro tema de interés asociado a la enseñanza de programación. Es por ello que es posible encontrar múltiples iniciativas que proponen herramientas para enseñar a programar utilizando el paradigma de programación visual basado en bloques que se encastran al estilo LEGO, tales como PilasBloques[4], Blocky Games[5], Scratch[6], APPInventor[7], entre otras. Estas herramientas están orientadas a diferentes niveles educativos como inicial, primario o secundario y favorecen introducir los conocimientos iniciales de programación de forma intuitiva e interactiva aumentando así el atractivo.

Si bien, como se mencionó anteriormente, se cuentan con numerosas herramientas para desarrollar actividades de enseñanza de programación en las aulas, los docentes no cuentan con herramientas sistematizadas que permitan realizar un seguimiento de su uso. Los sistemas LMS/CMS (Learning Management Systems/Content Management Systems) son entornos que permiten administrar de forma sistemática todo el proceso de enseñanza virtual. Éstos brindan a los docentes la posibilidad de gestionar y utilizar diferentes herramientas para el desarrollo de cursos y actividades que luego serán presentados a los estudiantes. A su vez estos sistemas permiten automatizar ciertas tareas rutinarias como por ejemplo la evaluación del progreso de los estudiantes y la obtención de métricas grupales e individuales. (Watson, Lee y Reigeluth, 2006) (Coble, 2016)

Como ejemplo de algunos entornos existentes que integran en parte las utilidades propias de los sistemas LMS/CMS con las herramientas de aprendizaje de programación

[1] Sitio oficial: http://program.ar/, último acceso 11/03/2020.
[2] Sitio oficial: https://www.argentina.gob.ar/educacion/aprender-conectados/conectar-igualdad, último acceso 11/03/2020.
[3] Sitio oficial:http://planied.educ.ar/, último acceso 11/03/2020.
[4] Sitio oficial: http://pilasbloques.program.ar/, último acceso 21/02/2019.
[5] Sitio oficial: https://blockly-games.appspot.com/, último acceso 21/02/2019.
[6] Sitio oficial: https://scratch.mit.edu/, último acceso 11/03/2020.
[7] Sitio oficial: https://appinventor.mit.edu/, último acceso 11/03/2020.

podemos mencionar la iniciativa code.org[8], que provee una propuesta con algún grado de seguimiento para los docentes. Sin embargo, no es posible recibir avisos o reportes de progresos, que resultan de suma utilidad para los docentes y estudiantes. También se puede mencionar como antecedente el proyecto Code Combat[9], que propone un acercamiento a la programación de juegos utilizando los lenguajes Javascript y Python y que incluye herramientas de seguimiento.

Disponer de un sistema que acompañe al docente durante el proceso de enseñanza de programación en la escuela resulta un aporte relevante para la práctica docente.

## 1.2 Objetivos

El objetivo de esta tesina es desarrollar un sistema LMS/CMS que acompañe la actividad docente en la enseñanza de la programación en el ámbito escolar, favoreciendo el uso de herramientas de programación visual basadas en bloques, popularmente adoptadas en este ámbito.

Para lograr este objetivo se proponen las siguientes actividades:

- Realizar un relevamiento de las herramientas LMS/CMS y de programación visual basada en bloques de código fuente abierto popularmente adoptadas en el ámbito escolar.
- Evaluar las herramientas relevadas de acuerdo a un criterio de selección previamente elaborado.
- Desarrollar Studium, un sistema basado en un LMS/CMS existente, que integre una herramienta de programación visual basada en bloques.
- Evaluar Studium en el contexto de proyectos de extensión de la Facultad de Informática vinculados con el acercamiento de la informática a la escuela.

## 1.3 Estructura del informe

La estructura del presente informe se divide en seis capítulos y un anexo abarcando los siguientes temas:

- **Capítulo 1 - Introducción**

  En este capítulo se describe la motivación y los objetivos de la presente tesina.

- **Capítulo 2 - Herramientas de enseñanza de programación**

  En este capítulo se realiza una introducción a las herramientas de enseñanza a la programación basadas en bloques visuales. Se analiza la herramienta a ser utilizada en esta tesina y los requisitos necesarios para su instalación y ejecución. Se detallan las diferentes funcionalidades que provee la misma.

[8] Sitio oficial: https://code.org/, último acceso 14/02/2019.
[9] Sitio oficial: https://codecombat.com/, último acceso 14/02/2019.

- **Capítulo 3 - Sistemas de Administración de Aprendizaje y de Administración de Cursos**

  En este capítulo se presenta un marco teórico sobre los sistemas de Administración de Aprendizaje y de Administración de Cursos. Se analizan varios de ellos para seleccionar uno como base para la implementación del sistema propuesto en esta tesina.

- **Capítulo 4 - Studium**

  En este capítulo se presentan los diferentes aspectos del diseño de Studium, como las tecnologías utilizadas, casos de uso, funcionalidad y otros detalles de su implementación.

- **Capítulo 5 - Evaluación de Studium**

  En este capítulo se detallan las pruebas propuestas para Studium.

- **Capítulo 6 - Conclusiones y trabajos futuros.**

  En este capítulo se describen las conclusiones, resultados obtenidos y trabajos futuros planteados para mejorar Studium.

- **Bibliografía.**

  Se detalla toda la bibliografía utilizada para el desarrollo de este proyecto.

- **Anexo**

# Capítulo 2. Herramientas de enseñanza de programación

Enseñar programación a niñas y niños en las escuelas presenta muchos desafíos. Entre ellos la diversidad en los niveles de habilidad y aptitud, la gran cantidad de herramientas disponibles, la cantidad de tiempo necesario para realizar actividades de programación, y por supuesto la dificultad de motivar a los estudiantes. Existen múltiples plataformas y herramientas orientadas a la enseñanza de programación, encontrándose entre las más adoptadas aquellas que utilizan el enfoque de la enseñanza basada en juegos y desafíos.

El aprendizaje basado en juegos está relacionado con el uso de los mismos, no para entretenimiento, sino con fines educacionales. Los profesionales de la educación han destacado en diversas investigaciones varias características que permiten usar los juegos como herramientas de aprendizaje. Por ejemplo, los juegos son atractivos visualmente (Dickey, 2005) y motivadores (Prensky, 2003). También proveen muchas experiencias diferentes (Arena and Schwartz, 2013) y una excelente devolución de cómo se desenvuelve el jugador (Shute, 2011). (Combefis, Beresnevicius, Dagiene, 2016)

Teniendo en cuenta esta información se seleccionó, para el propósito de este trabajo, una herramienta de enseñanza de programación basada en juegos que utiliza la programación en bloques para sus lecciones.

La elección de una herramienta que utilice la programación basada en bloques sobre la programación basada en texto, se debe a que la programación basada en bloques permite crear código de una forma visual, intuitiva y simple como contrapartida a los lenguajes de programación profesionales que requieren enfrentarse a una sintaxis rígida en idioma inglés. También se tuvo en cuenta en esta elección que actualmente la programación basada en bloques se constituye como un enfoque fuertemente adoptado para introducir la programación en las escuelas y es utilizado por múltiples herramientas alrededor del mundo.(Grover, Pea, Cooper, 2015)

Programar en estos ambientes toma la forma de la metáfora de “encastre”, seleccionar y arrastrar bloques visuales que pueden ser acoplados y así crear programas como puede verse en la Figura 1. Si dos bloques no pueden ser unidos para formar una sentencia sintácticamente correcta, la interface impedirá que se unan. Usando este tipo de herramientas, los estudiantes pueden aprender y practicar los principales conceptos de la programación, como instrucciones, contenedores (variables, constantes y listas), sentencias condicionales, repeticiones, operadores lógicos y datos de entrada/salida.(Weintrop, Wilensky, 2018)

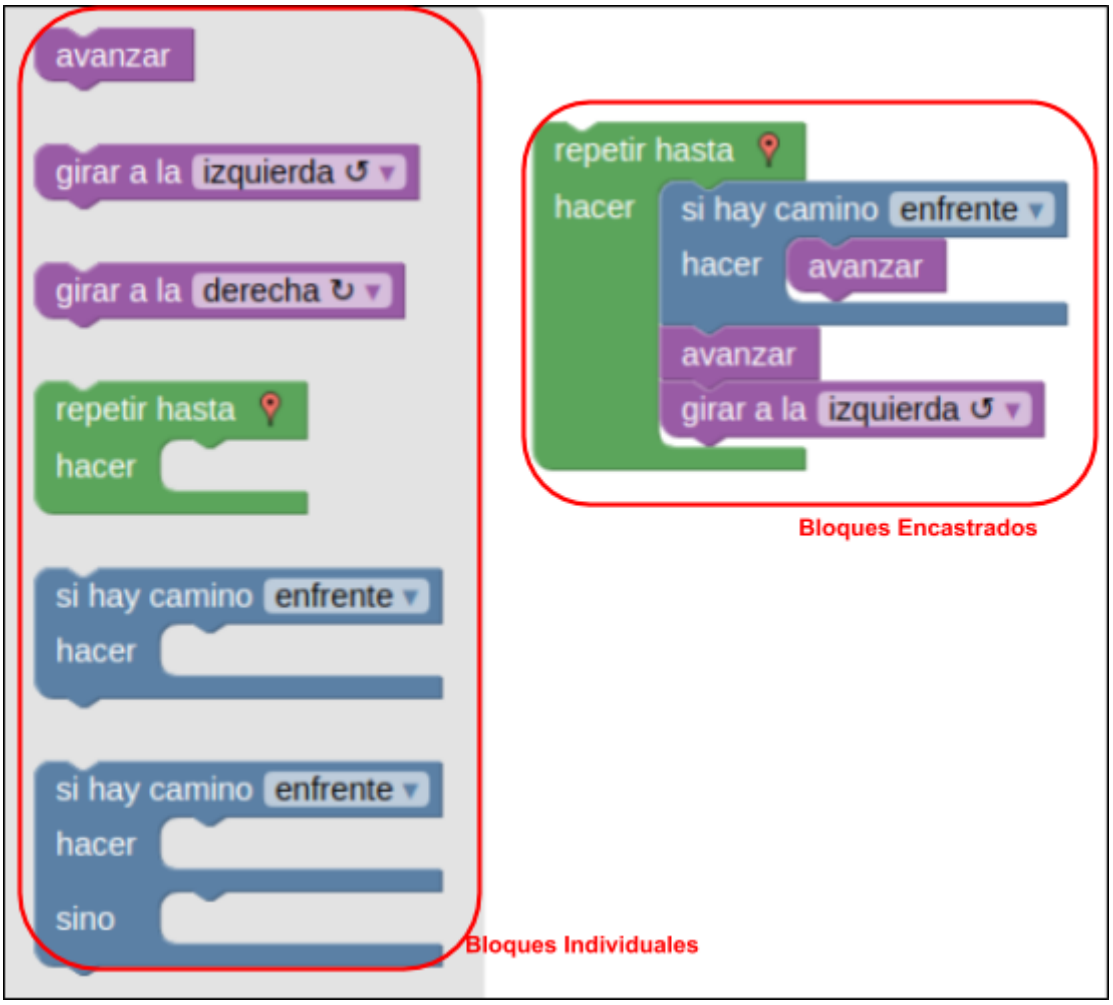


*Figura 1. Captura Navegador - Editor de Bloques.*

Considerando estos aspectos de la programación en bloques se concluye que ésta representa una introducción atractiva a los conceptos de la programación sin tener que enfrentar aspectos más complejos, como la sintaxis rígida de los lenguajes textuales. (Grover, Basu, 2017).

## 2.1 Criterios de Elección

Para la elección de la herramienta de enseñanza de programación se consideró que ésta debe ofrecer un código abierto y gratuito, además de que su funcionalidad esté disponible en idioma español o fácilmente adecuable a dicho idioma. Las dos posibles candidatas a utilizarse fueron:

- Pilas Bloques
- Blockly Games

La definición de estas herramientas como candidatas se basó en que ambas cumplen con las consideraciones mencionadas y que éstas son utilizadas en diferentes proyectos llevados a cabo por la Facultad de Informática de la UNLP relacionados a la enseñanza de programación en las aulas.

Si bien las dos herramientas cuentan con una interfaz sencilla e intuitiva y pueden ser utilizadas para el desarrollo de esta tesina se decidió integrar sólo una de ellas en el sistema a desarrollar.

Luego de un análisis preliminar se constató que la herramienta Blockly Games cuenta con un código más organizado, sencillo y fácil de leer que Pilas Bloques. Adicionalmente, si bien ambas herramientas están desarrolladas utilizando Javascript, Blockly Games utiliza Closure Templates[10] para administrar sus vistas, mientras que Pilas Bloques utiliza el framework Ember[11] para este propósito.

Considerando esta información, y teniendo en cuenta la experiencia previa del autor de este trabajo con Closure Templates, se decidió utilizar la herramienta Blockly Games para la integración propuesta durante esta tesina.

## 2.2 Blockly Games

### 2.2.1 Introducción

Blockly Games es una herramienta que ofrece una serie de juegos educacionales para enseñar a programar. Está diseñada para personas que no han tenido experiencias previas con la programación computacional y prepara a los jugadores para usar lenguajes de programación convencionales.

Éste es un proyecto creado por Google para alentar a los futuros programadores brindándoles un entorno donde puedan aprender por sus propios medios y a su propio ritmo. Se trata de un proyecto open source y utiliza una librería que permite construir editores compuestos por bloques visuales ensamblables, correspondientes a las distintas instrucciones de manera que programar se reduce a seleccionar y ensamblar ordenadamente los bloques que se quieren ejecutar.(Blockly Games, Google, 2019)

La lógica detrás de esta herramienta está implementada principalmente en Python y Javascript. Adicionalmente ésta utiliza Closure Templates de Google para la visualización de su interfaz de usuario.

Todos los juegos ofrecidos por Blockly Games están disponibles en más de 50 idiomas, incluyendo español.

### 2.2.2 Análisis

#### Instalación y detalles de implementación

Para el análisis de Blockly Games se realizó una instalación local del mismo en un sistema Linux. Para mayor detalle sobre los requerimientos del sistema y los pasos realizados para la instalación ver el Anexo A de esta tesina.

Los directorios principales que pueden observarse en el directorio raíz de la herramienta (Figura 2) son:

- **appengine** contiene la implementación de la mayor parte de la lógica y las plantillas de visualización separadas en directorios para aquellas que son de uso general y aquellas que son propias de cada juego.

[10] Sitio oficial: https://github.com/google/closure-templates, último acceso 15/03/2020.
[11] Sitio oficial: https://emberjs.com/, último acceso 15/03/2020.

- **json** contiene archivos json con las constantes de textos utilizadas para traducir la aplicación a cualquiera de los idiomas disponibles.
- **third-party** contiene las librerías necesarias para la ejecución de la herramienta, ésta se actualiza durante el proceso de instalación.

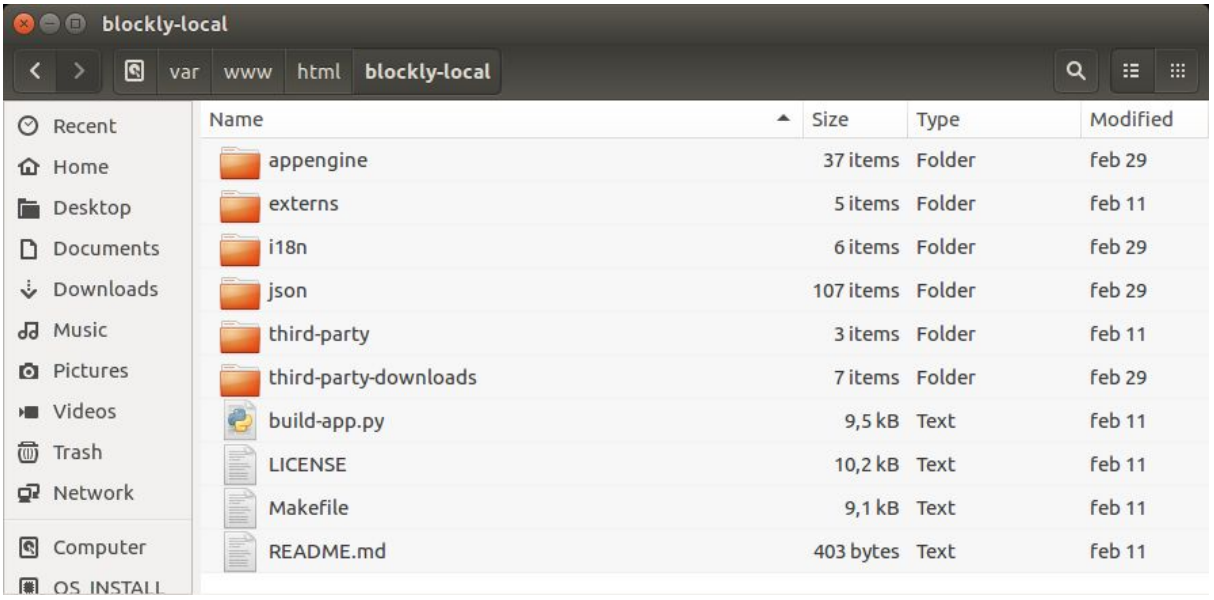


*Figura 2. Captura Escritorio - Directorio Raíz Blockly Games.*

Blockly Games es una herramienta que no requiere el registro de usuarios para acceder a su funcionalidad. Ésta registra toda la información de quien ejecute sus juegos en el almacenamiento local del navegador a través de cookies, de una manera totalmente transparente para el jugador. Si el jugador desea comenzar de nuevo los juegos durante una sesión éste tiene acceso a la funcionalidad de “Borrar datos” en la pantalla principal de la herramienta, la cual elimina toda la información almacenada localmente.

Adicionalmente Blockly Games ofrece la posibilidad de guardar el estado de un juego para que pueda ser accedido posteriormente desde otro navegador. Esta opción está disponible a través del botón “Enlazar y guardar los bloques” presente en la interfaz de todos sus juegos, a excepción de los juegos Rompecabezas y Estanque. Para procesar este pedido, la herramienta obtiene el código del juego en formato xml y genera un requerimiento, usando el objeto XMLHttpRequest, a su servidor en AppEngine. Como respuesta ésta obtiene un link con el cual se puede acceder al juego y restaurar el estado guardado en cualquier momento. Es importante destacar que esta funcionalidad solo está disponible en la versión web de Blockly Games y solo funciona si ésta se hace sobre un servidor AppEngine.

### Juegos Disponibles

Como puede verse en la Figura 3, Blockly Games ofrece ocho juegos que pueden ser accedidos desde su página principal.


*Figura 3. Captura Navegador - Página principal Blockly Games.*

Los juegos ofrecidos se centran en desarrollar diferentes habilidades de programación mediante la resolución de problemas. Cada juego, a excepción del juego de Rompecabezas y el juego de Estanque, cuentan con niveles en los cuales la dificultad de los problemas presentados se va incrementando.

Blockly Games ofrece los siguientes juegos:

- **Rompecabezas** es una introducción a los bloques de Blockly que permite comprender cómo combinarlos.
- **Laberinto** permite a los jugadores recorrer laberintos cada vez más difíciles. Este juego se centra en los bucles y condicionales básicos.
- **Pájaro** ofrece la posibilidad de controlar el vuelo de un pájaro y guiarlo hacia objetivos cada vez más difíciles de alcanzar. El juego busca desarrollar un conocimiento más profundo de los condicionales.
- **Tortuga** propone comprender más profundamente los bucles dibujando imágenes cada vez más complejas.
- **Película** permite animar figuras en películas cada vez más difíciles. Éste se centra en brindar una introducción a las ecuaciones matemáticas.
- **Música** permite realizar composiciones musicales cada vez más difíciles a través del uso de instrucciones agrupadas en funciones básicas.
- **Tutor del Estanque** introduce en la programación basada en texto y sirve como introducción al juego Estanque. Los niveles van y vienen entre el código en bloques y un editor de Javascript.
- **Estanque** permite programar un pato para luchar contra otros patos en una batalla por la supervivencia. Da la opción de usar programación de Bloques o un editor de Javascript y funciona como una integración de todos los contenidos presentados en los otros juegos.

Interfaz de Juego

Para mostrar la interfaz de un juego de Blockly Games se eligió el juego **Laberinto** que puede verse en la Figura 4. Entre la funcionalidad más importante se puede observar:

- **En la parte superior de la pantalla**: el menú que permite navegar entre los niveles disponibles para este juego y el campo que permite cambiar el idioma de Blockly Games en cualquier momento.
- **En la parte izquierda de la pantalla**: la visualización del juego donde se observa la ejecución del código creado por el jugador y el botón de Ejecutar el programa que inicia la ejecución.
- **En la parte media de la pantalla**: los tipos de bloques disponibles para la resolución de este juego.
- **En la parte derecha de la pantalla**: el editor de bloques donde el jugador deberá crear su programa utilizando los tipos de bloques disponibles.

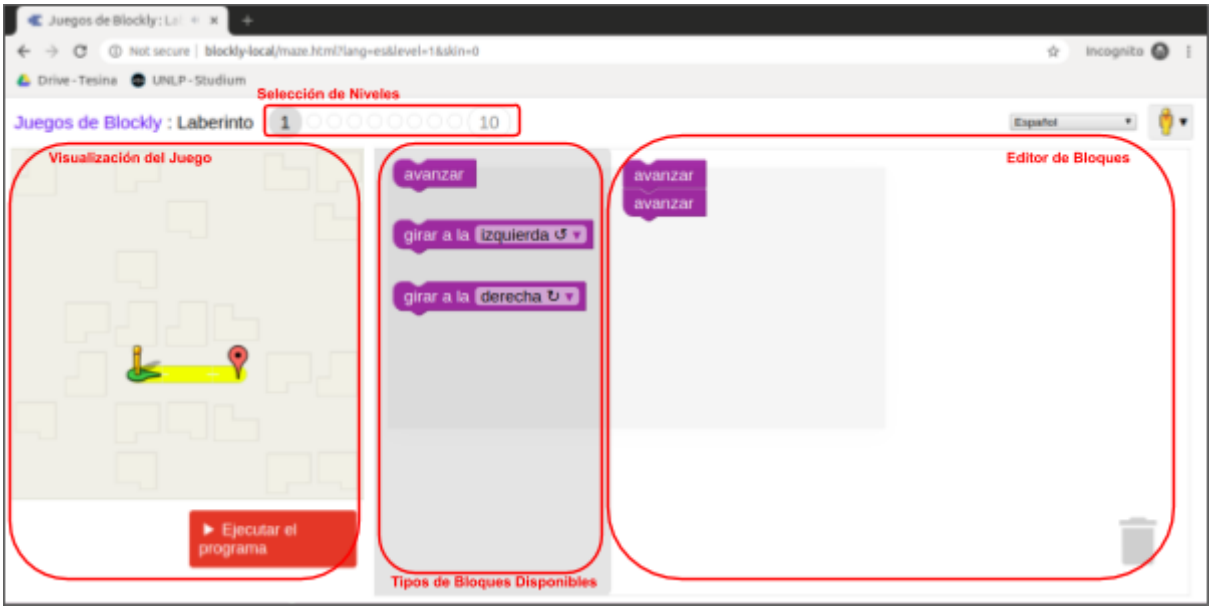


*Figura 4. Captura Navegador - Juego Laberinto Blockly Games.*

Una vez que el usuario termina de crear su programa puede ejecutarlo, para ello debe hacer clic en el botón “Ejecutar el programa” y validar la respuesta. Si la ejecución fue exitosa y se alcanzó el objetivo deseado se muestra una pantalla con un mensaje de Felicitaciones como puede verse en la Figura 5. En este mensaje también se muestra el código fuente en Javascript y se propone avanzar al próximo nivel.

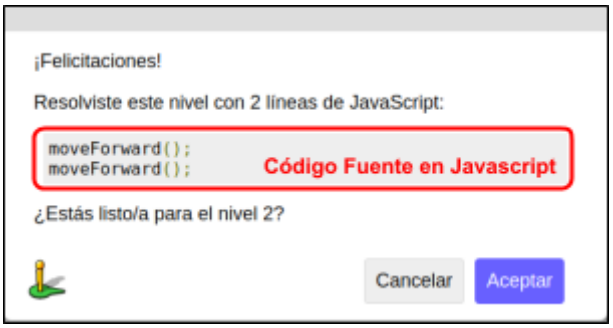


*Figura 5. Captura Navegador - Alerta Ejecución Exitosa Blockly Games.*

En caso que la ejecución no haya sido exitosa la herramienta muestra pistas sobre los cambios a realizarse en el programa para solucionar el problema y arribar a una ejecución exitosa.

## 2.3 Conclusión Final

Basándose en el análisis previo se puede concluir que Blockly Games presenta una interfaz de juego sencilla y simple de usar. Además, los juegos disponibles en la herramienta cubren diversos conceptos basicos de la programación y permiten a los usuarios tener una noción de cómo se verían los programas generados en bloques en un lenguaje textual.

Si bien la funcionalidad de gestión de estado de los juegos solo funciona al instalar la aplicación en un servidor, pago, de AppEngine, ésta puede adaptarse para almacenar y obtener el estado de una base de datos en otro tipo de servidor.

Adicionalmente, otra ventaja del uso de Blockly Games para el desarrollo de este trabajo es que ésta mantiene una comunidad activa y su código es actualmente mantenido por los desarrolladores, quienes pueden ser fácilmente contactados ante cualquier duda o inconveniente a través del repositorio de GitHub.

# Capítulo 3. Sistemas de Administración de Aprendizaje y de Administración de Cursos

Los sistemas de Administración de Aprendizaje y de Administración de Cursos, LMS/CMS, son entornos que permiten administrar de forma sistemática todo el proceso de enseñanza virtual. Éstos brindan a los docentes la posibilidad de gestionar y utilizar diferentes herramientas para el desarrollo de cursos y actividades que luego serán presentadas a los estudiantes. A su vez estos sistemas permiten automatizar ciertas tareas propias de la enseñanza, como por ejemplo la evaluación del progreso de los estudiantes y la obtención de métricas grupales e individuales. (Watson, Lee y Reigeluth, 2006) (Coble, 2016).

En este capítulo se analiza la funcionalidad de diferentes sistemas LMS/CMS y se determina cuál de ellos será utilizado como base para Studium justificando esta elección.

De 14 sistemas LMS/CMS inicialmente seleccionados se eligieron 3 de ellos como candidatos a ser utilizados en esta tesina por ser los que mejor se ajustan a las necesidades de desarrollo de Studium. Los tres sistemas candidatos que se analizan en esta sección son:

- Python-LMS
- ATutor
- Chamilo

Los sistemas candidatos se destacan por contar con algunas características tales como ser de código fuente abierto y licencia libre y tener funcionalidad en idioma español o un bajo esfuerzo de traducción. A su vez estos sistemas presentan, en forma parcial o total, cierta funcionalidad básica tal cómo:

- Manejo de Roles de Usuario:
    - Rol Estudiante
    - Rol Profesor
    - Rol Administrador
- Administración de Usuarios:
    - A/B/M de Usuario
    - Login
    - Listado de Usuarios
- Administración de Cursos:
    - A/B/M de Cursos
    - A/B/M de Ejercicios
    - Listado de Cursos Existentes
    - Listado de Cursos Actuales
    - Búsqueda de Cursos

De los 14 sistemas inicialmente seleccionados para analizar, 11 fueron descartados por presentar licencias pagas, no permitir acceso a su código fuente, no contar con funcionalidad en español y/o no ofrecer las funcionalidades básicas propias de los sistemas LMS/CMS, entre otras razones. Los 11 sistemas descartados son:

- TCExam
- Forma

- Moodle
- ILIAS
- Efront
- Docebo
- Dokeos
- Open Elms
- Gibbon
- OpenSIS
- Claroline

Para la evaluación de los 3 sistemas elegidos como candidatos, se elaboró la Tabla 1 que sistematiza las características deseables para Studium.

| | SI | PARCIALMENTE | NO |
|---|---|---|---|
| A/B/M Usuarios | | | |
| A/B/M Cursos | | | |
| A/B/M Ejercicios | | | |
| Login/Logoff | | | |
| Listado de Usuarios | | | |
| Listado de Cursos | | | |
| Listado de Ejercicios | | | |
| Búsqueda de Cursos | | | |
| Perfil Administrador | | | |
| Perfil Estudiante | | | |
| Perfil Profesor | | | |
| Licencia Código Abierto | | | |
| Funcionalidad Idioma Español | | | |

*Tabla 1. Características Sistemas LMS/CMS.*

## 3.1 Python-LMS

### 3.1.1 Introducción

Python-LMS es un Sistema de Administración de Aprendizaje, creado por la estudiante Norteamericana Abhimanyu Deora, para las competencias nacionales de la Technologic Student Association (TSA) del 2014. La TSA es una organización de estudiantes interesados en ciencia, tecnología, ingeniería y matemática. Esta organización comprende más de 250.000 estudiantes, de primaria y secundaria de los Estados Unidos, que realizan actividades centradas en el aprendizaje de diferentes aspectos tecnológicos.

Las competencias nacionales de la TSA agrupan a los proyectos ganadores de cada estado participante en una conferencia en Washington, DC. Estos proyectos luego son

juzgados por educadores especializados en tecnología e ingeniería además de por representantes de la industria relacionados a los mismos.(TSA, 2019)

Python-LMS fue desarrollado bajo la licencia abierta CC Attribution-NonCommercial-ShareAlike 4.0 International y su última actualización del código tiene fecha de febrero 6 del 2015.

### 3.1.2 Análisis

Instalación y Requerimientos de Sistema

Para el análisis de este sistema se realizó una instalación local de Python-LMS en un sistema operativo Linux. El código para la instalación se encuentra disponible para su descarga en el repositorio de git:

- https://github.com/adeora/Python-LMS.

El sistema está implementado utilizando el framework Flask[12] para Python. Adicionalmente utiliza la tecnología sqlite3[13] para el manejo de su base de datos.

La instalación fue realizada configurando un ambiente virtual de Python con Flask utilizando las siguientes versiones de estas tecnologías:

- python 3.5.2
- flask 1.1.1

En la Figura 6 se puede observar el contenido del directorio raíz de la instalación de Python-LMS. Entre su contenido más importante se puede mencionar:

- **Directorio application**: contiene toda la lógica relacionada a la vista y la funcionalidad del sistema.
- **Directorio venv**: éste fue creado para contener el ambiente virtual de Python utilizado en la instalación local del sistema,
- **Archivo database.db**: este archivo representa la base de datos del sistema y es manipulado utilizando sqlite3.
- **Archivo run.py**: es el punto de acceso al sistema e importa toda la lógica del sistema desde el directorio application.

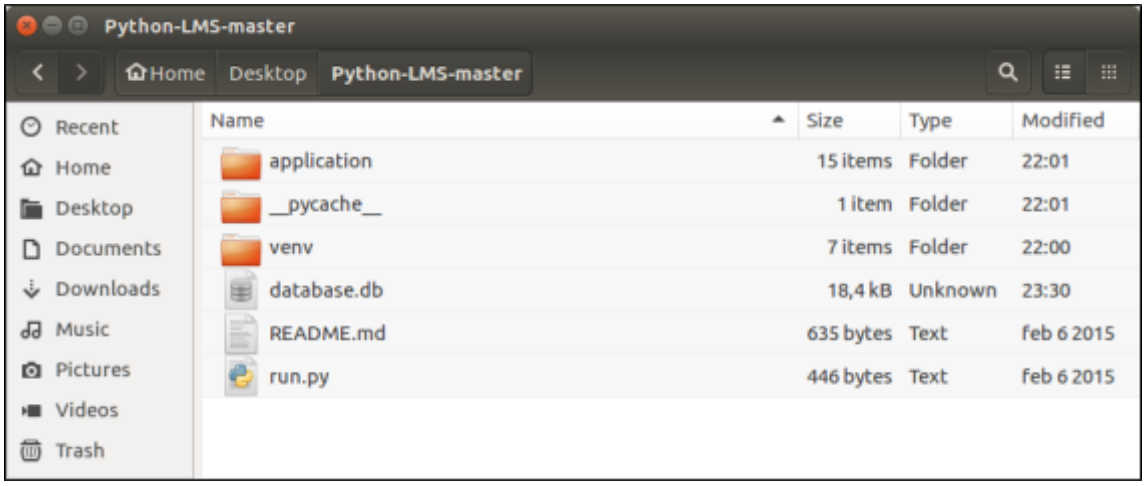


*Figura 6. Captura Escritorio - Directorio Root Python-LMS.*

[12] Sitio oficial: https://flask.palletsprojects.com/, último acceso 05/03/2020.
[13] Sitio oficial: https://www.sqlite.org/, último acceso 05/03/2020.

## Aspectos de uso

El sistema ofrece los perfiles de Estudiante y Profesor sin embargo la funcionalidad de ambos perfiles está disponible sólo en idioma inglés.

## Perfil Profesor

Los usuarios con el rol Profesor tienen la capacidad de crear clases que luego se harán disponibles a los Estudiantes. Estas clases tienen contenido agrupado en temas que pueden contener tareas y/o cuestionarios.

Como puede verse en la Figura 7, la página principal del profesor presenta un listado de las clases administradas por él mismo además de la opción de generar una nueva clase. Al crear una nueva clase el sistema devuelve al usuario el identificador de la misma y la agrega a su lista de clases.

Adicionalmente, esta pantalla presenta las opciones de navegación hacia los menús de administración de temas, asignaciones y cuestionarios. Estos menús permiten crear y modificar cada tipo de contenido y asignarlos a cualquiera de las clases o temas existentes.

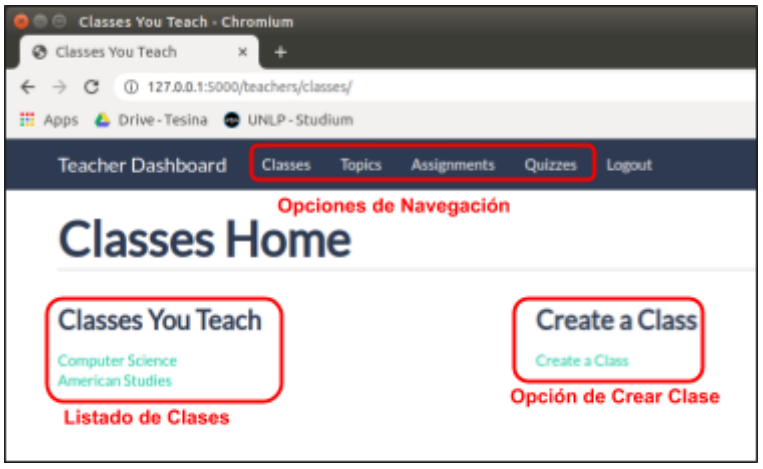


*Figura 7. Captura Navegador - Página Principal Profesor Python-LMS.*

Los cuestionarios creados en Python-LMS pueden tener varias preguntas de tipo “multiple choice” con una sola respuesta correcta. Los resultados de los estudiantes se muestran en porcentaje de respuestas correctas sobre el total de preguntas y son calculados automáticamente. En cuanto a las asignaciones, éstas constan de un título y una consigna para el estudiante. La entrega de las mismas se realiza por fuera del sistema.

Al acceder a cada clase el profesor puede ver la información relacionada a la misma y agregar las notas de las asignaciones relacionadas a la clase.

Una característica de Python-LMS es que no implementa la baja de clases, temas, asignaciones o cuestionarios.

## Perfil Estudiante

El perfil de Estudiante permite a los usuarios inscribirse a las clases y realizar sus actividades. Como puede observarse en la página principal de este perfil, mostrada en la Figura 8, para inscribirse a una clase, el usuario deberá utilizar el identificador generado

durante la creación de la misma. Este identificador debe ser informado al usuario previamente, fuera del sistema, por el profesor que creó la clase.

Adicionalmente esta pantalla muestra la lista de clases en las que se encuentra inscripto el estudiante, permitiendo navegar entre ellas.

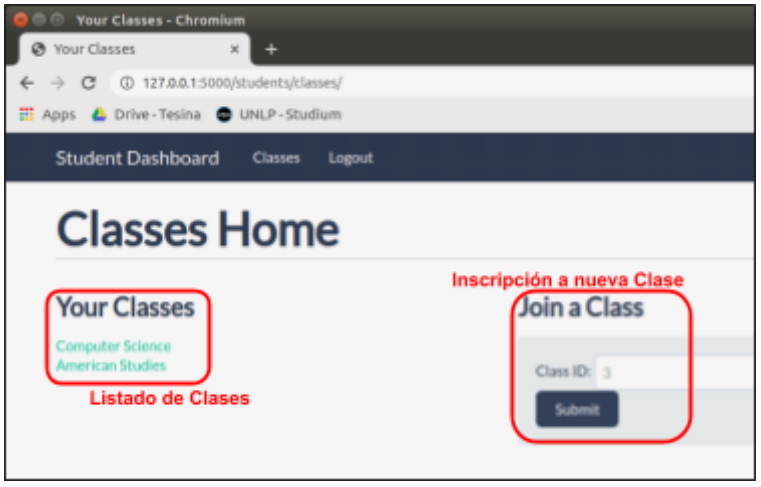


*Figura 8. Captura Navegador - Página Principal Estudiante Python-LMS.*

Al ingresar en una clase, el estudiante podrá ver los cuestionarios y asignaciones de la misma. Al resolver un cuestionario, el sistema muestra una pantalla con el porcentaje de respuestas correctas. Los cuestionarios pueden ser respondidos solo una vez y las respuestas no pueden ser modificadas.

### 3.1.3. Conclusión

La Tabla 2 sintetiza las características de Python-LMS.

| | SI | PARCIALMENTE | NO |
|---|---|---|---|
| A/B/M Usuarios | | X | |
| A/B/M Cursos | | X | |
| A/B/M Ejercicios | | X | |
| Login/Logoff | X | | |
| Listado de Usuarios | | | X |
| Listado de Cursos | | X | |
| Listado de Ejercicios | X | | |
| Búsqueda de Cursos | | | X |
| Perfil Administrador | | | X |
| Perfil Estudiante | X | | |
| Perfil Profesor | X | | |
| Licencia Código Abierto | X | | |
| Funcionalidad Idioma Español | | | X |

*Tabla 2. Características Sistemas Python-LMS.*

Entre las características destacables de Python-LMS se puede observar que implementa el manejo de usuarios con perfiles de estudiante y profesor y permite crear cursos y ejercicios en forma de clases y cuestionarios. Además ofrece la posibilidad de acceder a un listado de cuestionarios en cada clase.

Por otro lado Python-LMS ofrece cierta funcionalidad que no ha sido desarrollada completamente, como por ejemplo el alta de usuarios está disponible en el código fuente pero no es accesible desde ningún menú del sistema, o la baja de usuarios, clases y cuestionarios no ha sido implementada en absoluto. En cuanto al listado de cursos, si bien el sistema permite observar las clases relacionadas a cada usuario, no hay forma de obtener un listado de todas las clases del sistema. Adicionalmente, se puede destacar que si bien el sistema permite generar asignaciones, este no da la posibilidad de procesar sus entregas.

Finalmente se observa que al carecer de perfil administrador, Python-LMS no cuenta con un listado de usuarios del sistema. Esta funcionalidad permitiría tener un mejor control de los usuarios. Hay que mencionar, además, que aunque no represente un esfuerzo de traducción prohibitivo, toda la funcionalidad del sistema esta en idioma inglés

## 3.2 Atutor

### 3.2.1 Introducción

ATutor es un sistema LMS de código fuente abierto cuyo diseño está guiado por diferentes aspectos de accesibilidad y adaptabilidad. Este sistema surgió luego de dos estudios, Courseware Accessibility Study (1999)[14] e Inclusion in an Electronic Classroom (2000)[15], que indagaban en la accesibilidad provista por los sistemas LMS más populares a persona con discapacidades. El primero de estos estudios constó de una auditoría técnica que cuantificó la accesibilidad de estos sistemas utilizando las pautas de Accesibilidad del Contenido Web de la W3C (WCAG 1.0). El segundo de éstos se basó en un estudio de usuarios con diversos tipos de discapacidades. Los usuarios participaron en un curso online de 6 semanas en el que la unidad de cada semana era presentada en un nuevo sistema LMS.(ATutor, 2020)

Atutor soporta varios estándares de accesibilidad e interoperabilidad entre los que se pueden mencionar:

- W3C WCAG 1.0
- W3C WCAG 2.0
- ISO/IEC 24751
- OpenSocial 1.0
- SCORM Content Packaging
- IMS Question Test Interoperability (QTI) 1.2/2.1
- W3C XHTML 1.0

[14] Documento del estudio: https://atutor.github.io/research/crseval/crseval.html, último acceso 05/03/2020.
[15] Documento del estudio: https://atutor.github.io/research/access_study/inclusion.html, último acceso 05/03/2020.

Entre otras características notables, ATutor permite desarrollar y aplicar nuevos temas para cambiar fácilmente el estilo del sistema y la integración de módulos con nuevas funcionalidades para el mismo.

ATutor fue desarrollado bajo la licencia GNU General Public License (GPL), que permite copiar, distribuir y modificar su código.

La última actualización del código de este sistema tiene fecha del 9 de Septiembre del 2019.

### 3.2.2 Análisis

#### Instalación y Requerimientos de Sistema

Para el análisis de ATutor se realizó una instalación local del mismo en un servidor Apache sobre un sistema operativo Linux. El código de este sistema está disponible para su descarga en el repositorio git:

- https://github.com/atutor/ATutor

El sistema fue implementado en PHP[16] 5 y utiliza MySQL[17] como motor de base de datos. Si bien la última versión de ATutor fue implementada parcialmente para utilizar tanto PHP 5 como PHP 7, ésta aún presenta gran número de errores al ser instalada en un ambiente con PHP 7. Considerando esto se decidió utilizar PHP 5 para este análisis.

Se configuró un Host Virtual de Apache, para la instalación local, utilizando las siguientes versiones de las tecnologías requeridas:

- PHP 5.6
- MySQL 5.7.29
- Apache 2.4.41

#### Aspectos Generales

En la Figura 9 se puede observar el contenido del directorio raíz de la versión 2.2.4 de ATutor. Entre los componentes más importantes del sistema podemos mencionar:

- **Directorio admin:** contiene la funcionalidad propia del perfil de usuario administrador.
- **Directorio include:** éste contiene las clases, librerías y archivos html que implementan gran parte de la funcionalidad del sistema.
- **Directorio jscripts:** contiene toda la lógica implementada utilizando Javascript.
- **Directorio install:** éste contiene los archivos utilizados durante la instalación del sistema y la configuración de la base de datos.
- **Directorio users:** contiene toda la funcionalidad propia de los perfiles de usuario estudiante e instructor.

[16] Sitio oficial: https://www.php.net/, último acceso 05/03/2020.
[17] Sitio oficial: https://www.mysql.com/, último acceso 05/03/2020.

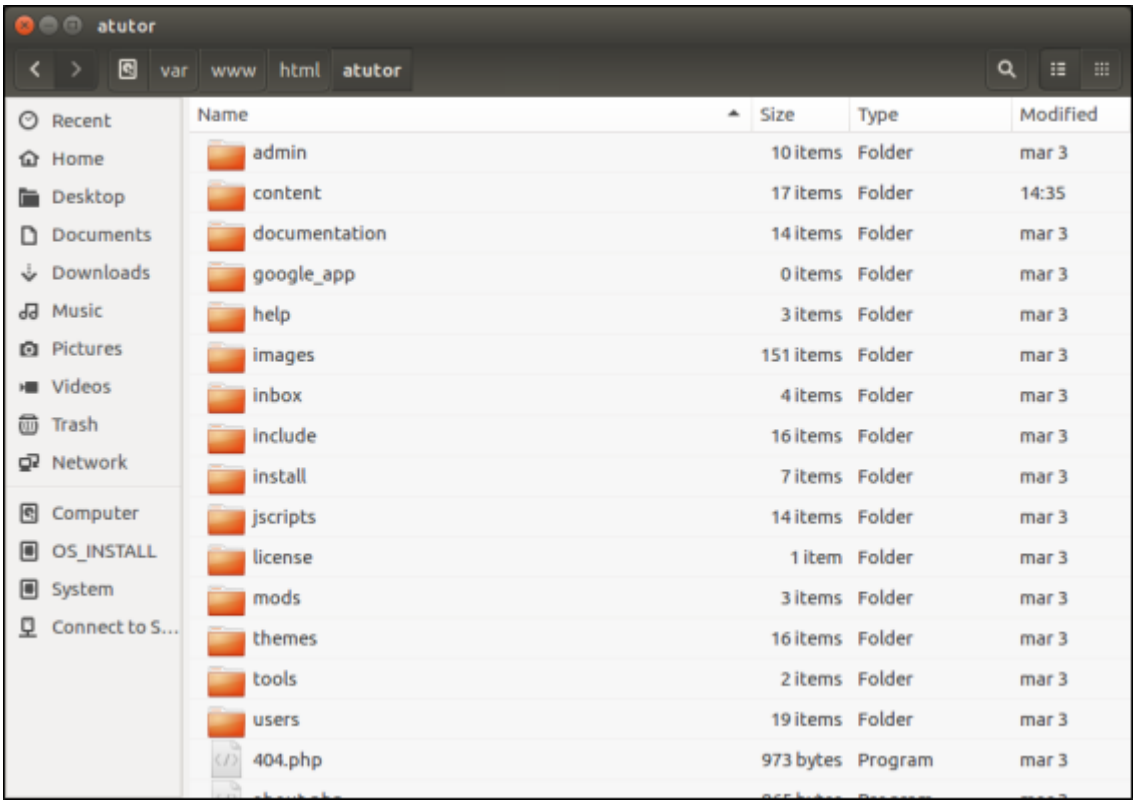


*Figura 9. Captura Escritorio - Directorio Root ATutor.*

ATutor ofrece paquetes de idiomas que pueden ser descargados del repositorio git:

- https://github.com/atutorlangs

Este repositorio contiene paquetes en más de 50 idiomas, entre los que se incluye el español. Una vez descargado un paquete se puede incorporar el mismo al sistema utilizando la funcionalidad de Administración de Lenguajes.

Al intentar incorporar el paquete de lenguaje con el idioma español se detectó un error en el sistema que no permitió continuar con la operación. Debido a esto y teniendo en cuenta que el presente análisis se enfocará en la funcionalidad provista por el sistema no se realizó la configuración de este idioma en la instalación local. A pesar de esto se tendrá en cuenta la posibilidad de obtener la funcionalidad del sistema en español en la conclusión final.

### Aspectos de uso

ATutor provee la funcionalidad de registro y autentificación de usuarios. Se ofrecen 3 tipos de perfiles de usuarios: perfil administrador, perfil instructor y perfil estudiante.

### Perfil Administrador

El perfil Administrador permite a un usuario supervisar y gestionar cada detalle del sistema. Como puede verse en la Figura 10, que muestra la pantalla principal de un administrador, éste tiene acceso a estadísticas e información técnica de la instalación y a diferentes menús de configuración del sistema.

Entre la funcionalidad destacada, accesible desde esta pantalla, se puede mencionar:

- Administración de usuarios.
- Administración de cursos.
- Administración de preferencias del sistema.

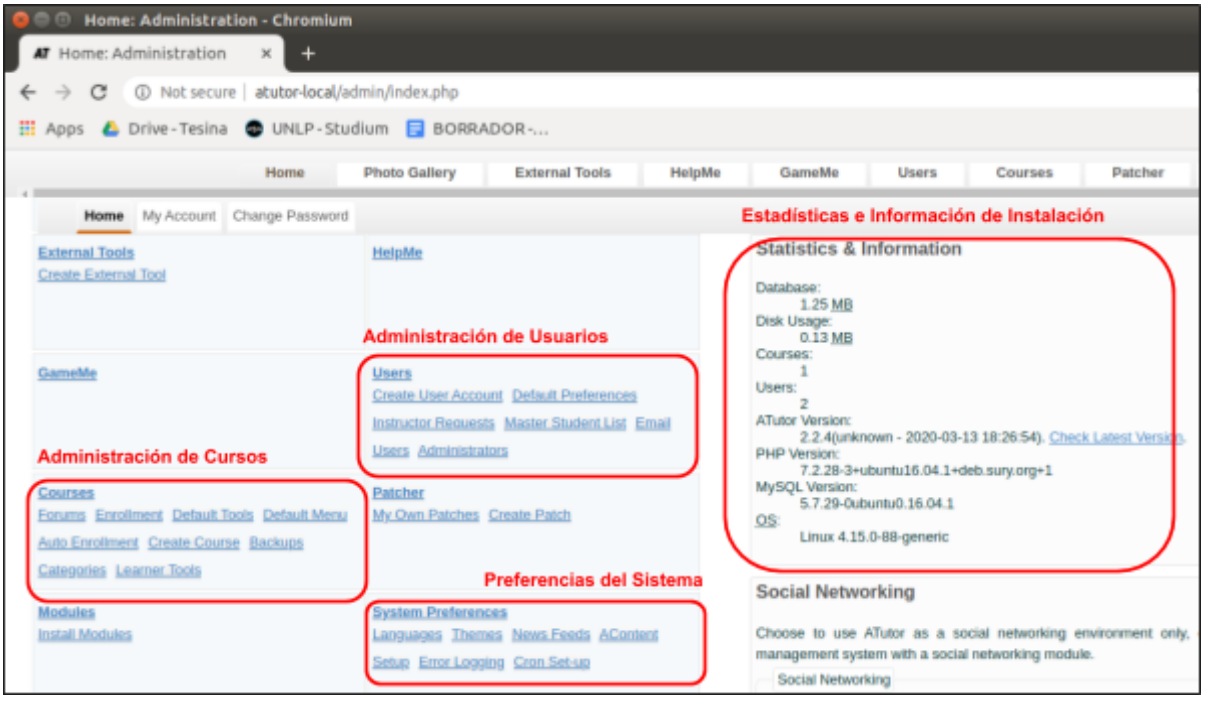


*Figura 10. Captura Navegador - Menú Principal Administrador ATutor.*

El menú de administración de usuarios permite acceder a un listado de todos los usuarios del sistema mostrando la información más importante de cada uno de ellos. Este menú también brinda la posibilidad de realizar búsquedas complejas de usuarios filtrando por diferentes parámetros tales como estado de la cuenta, último login, nombre y email, entre otros.

Adicionalmente esta pantalla permite navegar entre funcionalidades tales como configurar los campos por defecto de los usuarios, enviar mensajes globales a grupos de usuarios y gestionar los pedidos de registro de instructores.

### Perfil Instructor

El perfil instructor permite a un usuario gestionar cursos dentro del sistema. Como puede observarse en la Figura 11 éste tiene acceso a la opción que permite crear un curso en su menú principal. Los cursos creados por este usuario serán mostrados en la lista de “Mis Cursos”, indicando en la columna de estado su rol de instructor.

Los instructores también pueden inscribirse a otros cursos disponibles en el sistema como “Estudiante”. En este caso los cursos se muestran en la misma lista pero con un rol de estudiante.

Esta pantalla también permite navegar a otra funcionalidades del sistema como buscar cursos, editar el perfil de usuario o acceder al calendario con actividades de los cursos a los que el usuario se encuentra inscripto entre otras.

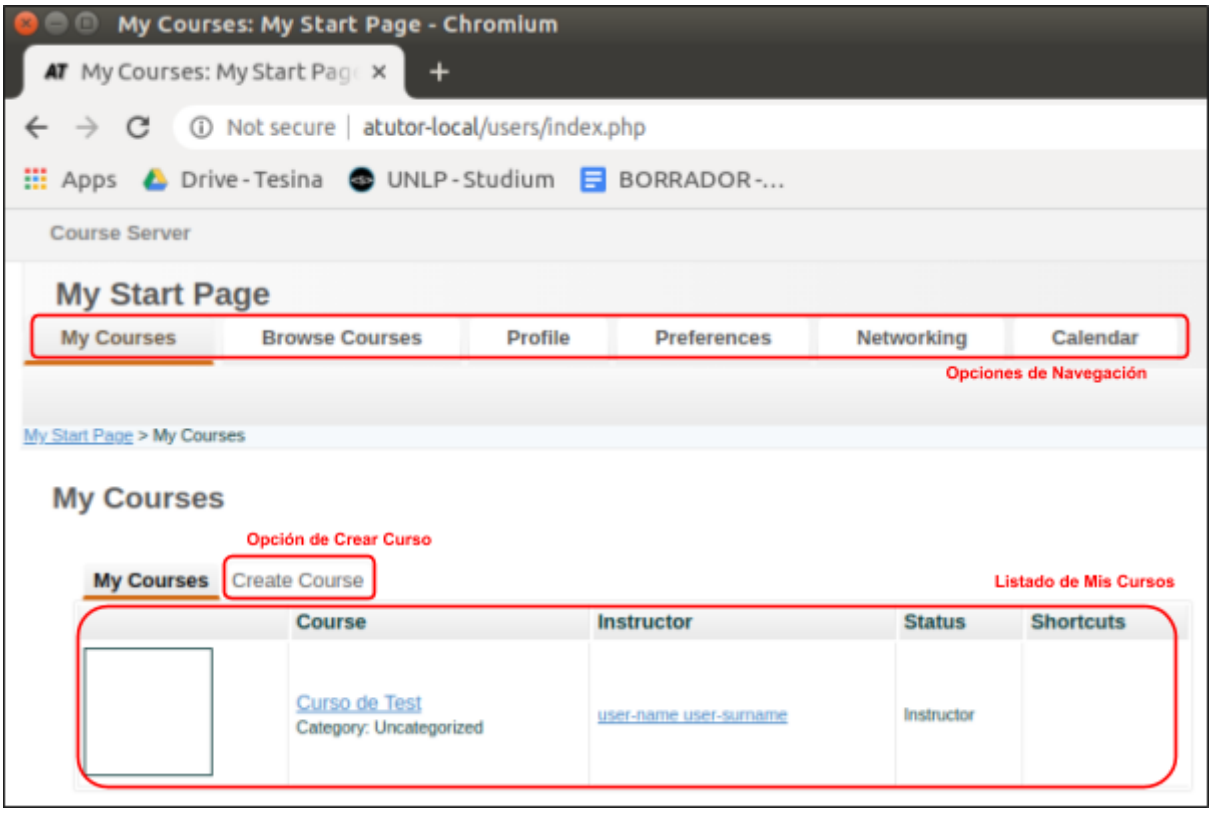


*Figura 11. Captura Navegador - Menú Principal Instructor ATutor.*

Al acceder a un curso con el rol de instructor, éste puede administrar diferentes aspectos del mismo. La funcionalidad de mayor interés para este análisis se encuentra en la sección de “Pruebas y Cuestionarios” que permite visualizar y administrar los ejercicios pertenecientes a un curso específico.

Al seleccionar un ejercicio se puede acceder a la visualización de los resultados registrados para el mismo donde se listan las respuestas a un ejercicio y permite asignar una nota a cada una. Al finalizar la corrección de un ejercicio se informa al estudiante de su nota final.

Una de las funcionalidades más importantes de la pantalla de “Pruebas y Cuestionarios” es el Banco de Preguntas. Esta pantalla permite crear una nueva pregunta o editar las preguntas ya creadas por este usuario. ATutor ofrece la creación de 8 tipos de preguntas, entre los que se incluyen opción múltiple, respuesta múltiple, coincidencia de respuestas, entre otras. Una vez creada una pregunta, ésta es agregada al banco de preguntas y puede ser utilizada en los cursos y ejercicios administrados por el instructor.

Adicionalmente, la pantalla de administración del curso también contiene la funcionalidad de crear nuevos ejercicios, gestionar las preguntas disponibles en el curso y las categorías que las agrupan.

ATutor también ofrece otras funcionalidades para sus cursos, además de las ya mencionadas, entre ellas permite el manejo de asignaciones, que pueden ser entregadas utilizando Dropbox o un gestor de archivos integrado en el sistema, la creación de anuncios en formato html que se envían a todos los usuarios inscriptos en el curso y, la administración de grupos de chat y de casillas de mensajes.

Perfil Estudiante

Al registrar un usuario en ATutor, éste se crea inicialmente con el perfil de Estudiante. Como se muestra en la Figura 12, un estudiante puede generar un pedido para registrarse como Instructor que es analizado y procesado por el administrador del sistema.

Entre las opciones de navegación, un estudiante puede buscar y registrarse en diferentes cursos. Todos los cursos en los que se encuentra registrado se muestran en su lista de “Mis Cursos”, desde donde puede acceder a cada uno de ellos.

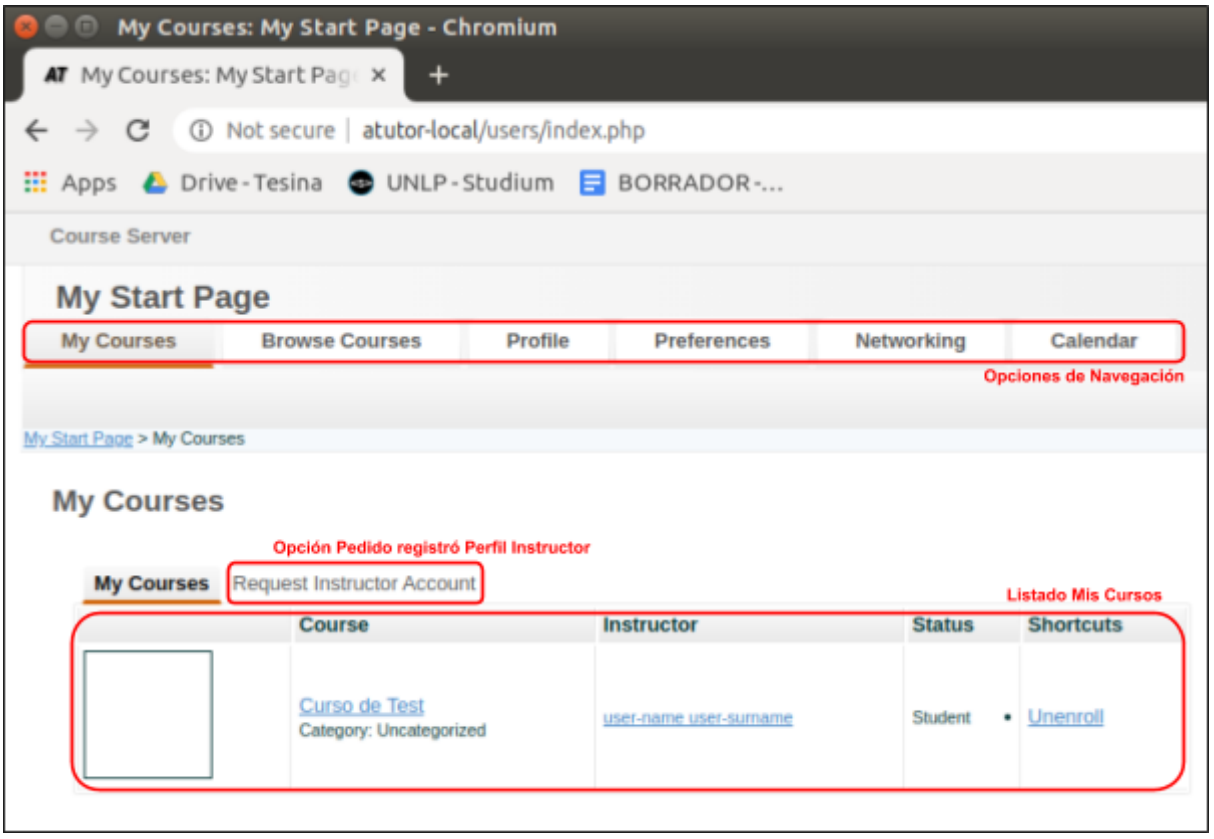


*Figura 12. Captura Navegador - Menú Principal Estudiante ATutor.*

Dentro de un curso, un estudiante puede acceder al directorio del mismo para acceder a los contenidos cargados por el instructor, y cargar archivos utilizando el administrador de archivos del curso.

Adicionalmente el estudiante puede acceder al menú de “Pruebas y Encuestas” del curso para visualizar todos los ejercicios disponibles en el mismo. Para cada ejercicio se puede observar el estado, la fecha de disponibilidad del mismo y la cantidad de intentos de respuesta posibles. Este menú muestra también los intentos realizados para cada ejercicio, donde se puede observar, entre otras cosas, la nota obtenida una vez que el instructor del curso realiza la corrección.

Al realizar un ejercicio, según la configuración del mismo, se muestra una pregunta por página o todas las preguntas en una misma página. El ejercicio consta de un encabezado con su nombre y las instrucciones a seguir. Por cada pregunta del ejercicio se visualiza un encabezado con el tipo de pregunta y una consigna.

### 3.2.3 Conclusión

Como puede observarse en la Tabla 3, ATutor es un sistema con una funcionalidad muy completa que cumple con los requisitos buscados para ser utilizado como base de Studium.

| | SI | PARCIALMENTE | NO |
|---|---|---|---|
| A/B/M Usuarios | X | | |
| A/B/M Cursos | X | | |
| A/B/M Ejercicios | X | | |
| Login/Logoff | X | | |
| Listado de Usuarios | X | | |
| Listado de Cursos | X | | |
| Listado de Ejercicios | X | | |
| Búsqueda de Cursos | X | | |
| Perfil Administrador | X | | |
| Perfil Estudiante | X | | |
| Perfil Profesor | X | | |
| Licencia Código Abierto | X | | |
| Funcionalidad Idioma Español | X | | |

*Tabla 3. Características Sistema Atutor.*

Entre las características más importantes, ATutor implementa de forma completa y efectiva toda la funcionalidad de A/B/M y gestión de Usuarios, Cursos y Ejercicios. Al ofrecer diferentes tipos de perfiles de usuarios permite una división clara de las funcionalidades disponibles para cada uno definiendo así su rol en el sistema.

ATutor también ofrece la posibilidad de traducir la funcionalidad disponible a varios idiomas, incluido el español. Adicionalmente otra funcionalidad que puede resultar de interés para los usuarios es que toda la funcionalidad de ATutor se encuentra documentada en detalle en su sitio y en múltiples foros de usuarios, y esto junto al estar implementado bajo una licencia de código fuente abierto, facilita la incorporación de mejoras en futuros trabajos. .

Por otro lado, se encontró durante el análisis del sistema numerosos errores de la aplicación debido a que ésta fue implementada para tecnologías que ya han sido reemplazadas por nuevas versiones. De esta forma se puede notar, durante la ejecución del sistema, páginas que no cargan, numerosos errores de la base de datos y funcionalidades que si bien están implementadas, son inaccesibles. La última versión estable de ATutor utiliza PHP 5.4 el cual se encuentra obsoleto y presenta grandes riesgos de seguridad.

Si bien ATutor comenzó a migrar a tecnologías nuevas de forma organizada bajo la supervisión de sus desarrolladores originales, éstos no llegaron a completar el trabajo debido a que se encuentran actualmente trabajando en otros proyectos dejando así la

actualización de ATutor en manos de los usuarios interesados en colaborar con la actualización.

## 3.3 Chamilo

### 3.3.1 Introducción

Chamilo LMS es una plataforma o campus virtual orientado a la formación online/e-learning.

El proyecto fue lanzado oficialmente en el 2010 por parte de la comunidad activa del proyecto Dokeos[18] creándose así un fork o división de este sistema.

La asociación Chamilo inicialmente fue la encargada de gestionar el desarrollo de Chamilo desde Bruselas, Bélgica. Se definió como principal objetivo proveer una plataforma de e-learning sencilla y popularmente adoptada bajo la fórmula del software libre, haciendo de ella un motor para el desarrollo educativo de las personas de cualquier parte del mundo. En 2014, la asociación se reubico en España creándose así la Asociación Chamilo España que gestiona actualmente la comunidad. (Chamilo, 2019)

El sistema cuenta con más de 21 millones de usuarios en todo el mundo. Chamilo se distribuye bajo licencia GNU/GPL v3 que permite a cualquier usuario o empresa usar, estudiar, modificar, mejorar y/o redistribuir su código.(Chamilo, 2019)

El código del sistema actualmente es mantenido por desarrolladores de BeezNest[19], un sponsor de la asociación Chamilo. Estos desarrolladores también brindan un soporte activo de la aplicación a través de su repositorio en GitHub.

### 3.3.2 Análisis

#### Instalación y Requerimientos

Para el análisis de la aplicación se realizó una instalación local de Chamilo en un servidor Apache sobre un sistema Linux. Los pasos detallados seguidos durante la instalación pueden verse en el Anexo B de este documento.

El código para la instalación se encuentra disponible en el siguiente link de la página del sistema:

- https://chamilo.org/es/descargar/

Chamilo está implementado en lenguaje de programación PHP y utiliza el framework TWIG[20] para administrar sus vistas.

TWIG es un motor de plantillas que permite compilar plantillas html a partir de un código PHP optimizado. Este utiliza una sintaxis muy sencilla que incorpora la funcionalidad necesaria para generar plantillas complejas. El framework fue desarrollado por el creador de Symfony, Fabien Potencier, y es distribuido bajo la licencia de software libre BSD. (Symfony, 2020)

[18] Sitio oficial: https://www.dokeos.com/, último acceso 06/03/2020.
[19] Sitio oficial: https://beeznest.com/es/, último acceso 06/03/2020.
[20] Sitio oficial: https://twig.symfony.com/, último acceso 15/03/2020.

Adicionalmente Chamilo utiliza MySQL como gestor de base de datos. Para la instalación de su última versión estable se tuvieron en cuenta los siguientes requerimientos de sistema:

- Apache 2.2+
- MySQL 5.6+ o MariaDB 5+
- PHP 5.6, 7.1, 7.2 o 7.3

Aspectos Generales

Se puede apreciar en la Figura 13 el contenido del directorio raíz del Chamilo. Entre los directorios más importantes de este destacan:

- **Directorio app:** este contiene los archivos de configuración del sistema y los datos de caché del mismo.
- **Directorio main:** contiene toda la lógica propia de chamilo, los templates usados por TWIG y los archivos de traducción para cada lenguaje.
- **Directorio vendor:** este contiene todas las librerías externas utilizadas por el sistema.

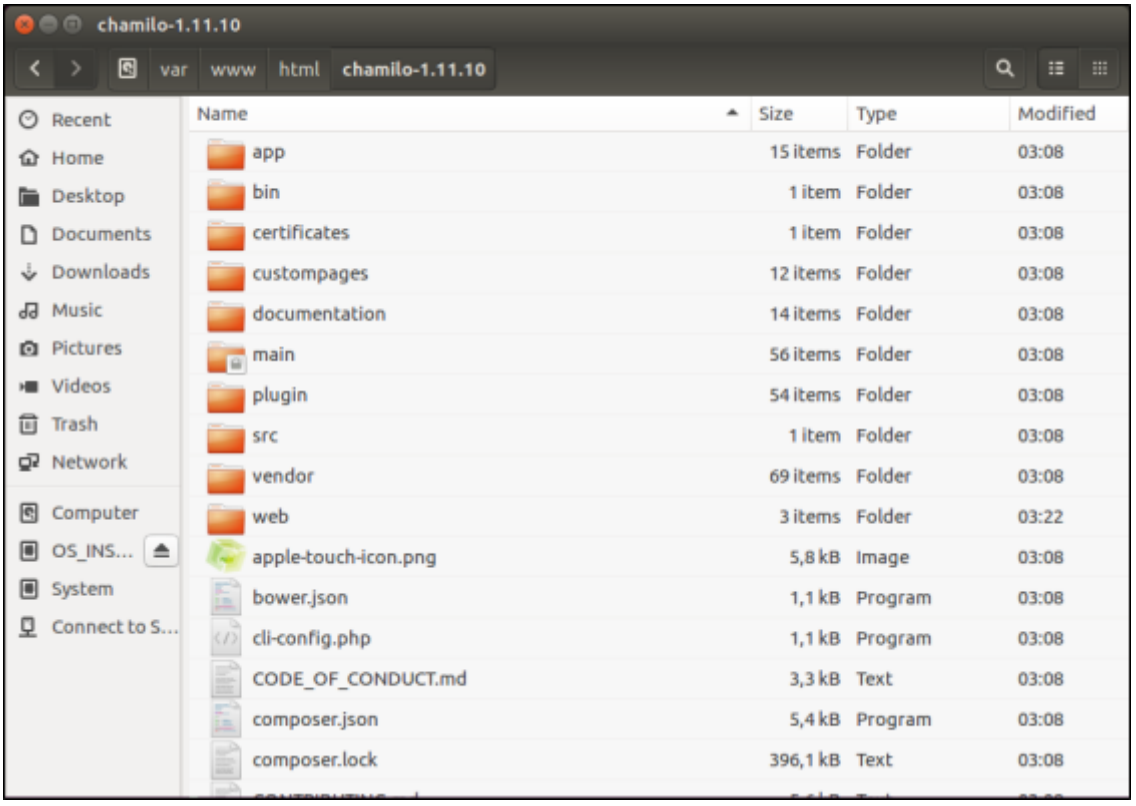


*Figura 13. Captura Escritorio - Directorio Raíz Chamilo.*

Chamilo dispone de su funcionalidad en más de 34 idiomas, incluyendo el español. Estos idiomas están incluidos en la instalación del sistema y no se requiere ninguna configuración adicional para acceder a los mismos. Los usuarios pueden seleccionar el idioma deseado al autenticarse en el sistema o definir un idioma predeterminado para su cuenta.

Aspectos de uso

Los usuarios del sistema pueden ser registrados con varios perfiles entre los que se incluyen los perfiles de Profesor y Estudiante. Solo existe una cuenta de administrador del sistema que se configura durante la instalación del mismo.

Perfil Administrador

Como puede observarse en la Figura 14, el administrador del sistema dispone de menús de administración para diferentes aspectos de éste. Desde esta pantalla es posible gestionar los campos disponibles, añadir diferentes contenidos y obtener estadísticas y reportes, entre muchas otras funcionalidades.

Entre los menús con la funcionalidad más destacable se pueden mencionar los menús de administración de Usuarios, Cursos y de la Plataforma.

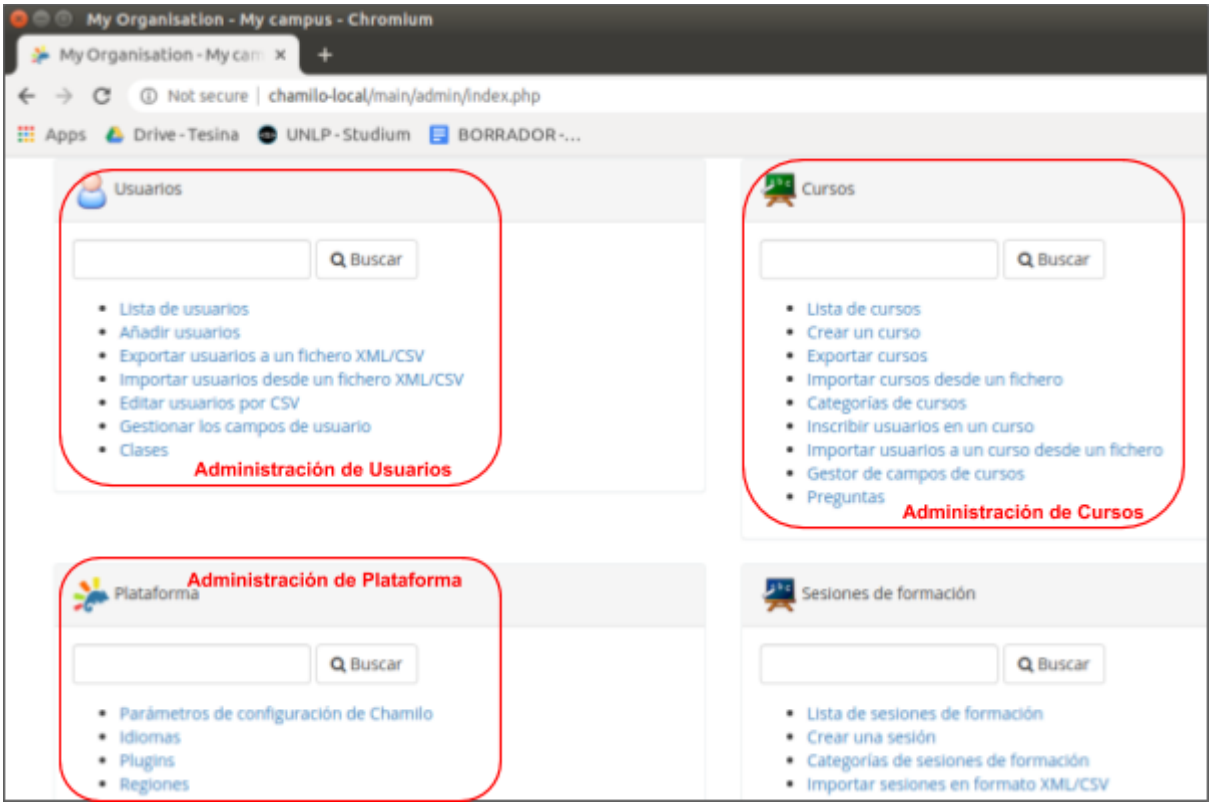


*Figura 14. Captura Navegador - Página Administración Chamilo.*

La “Lista de usuarios”, que puede observarse en el menú de administración de usuarios de la Figura 15, muestra todos los usuarios registrados en el sistema. Por cada usuario se pueden obtener sus datos más relevantes, junto a su perfil asignado y su estado.

A partir de la página de Administración también es posible realizar búsquedas de usuarios usando diferentes criterios. Adicionalmente es posible acceder a acciones para cada usuario, tales como listar sus cursos, iniciar sesión como si fuese el usuario, obtener informes o eliminar la cuenta, entre otras funcionalidades.

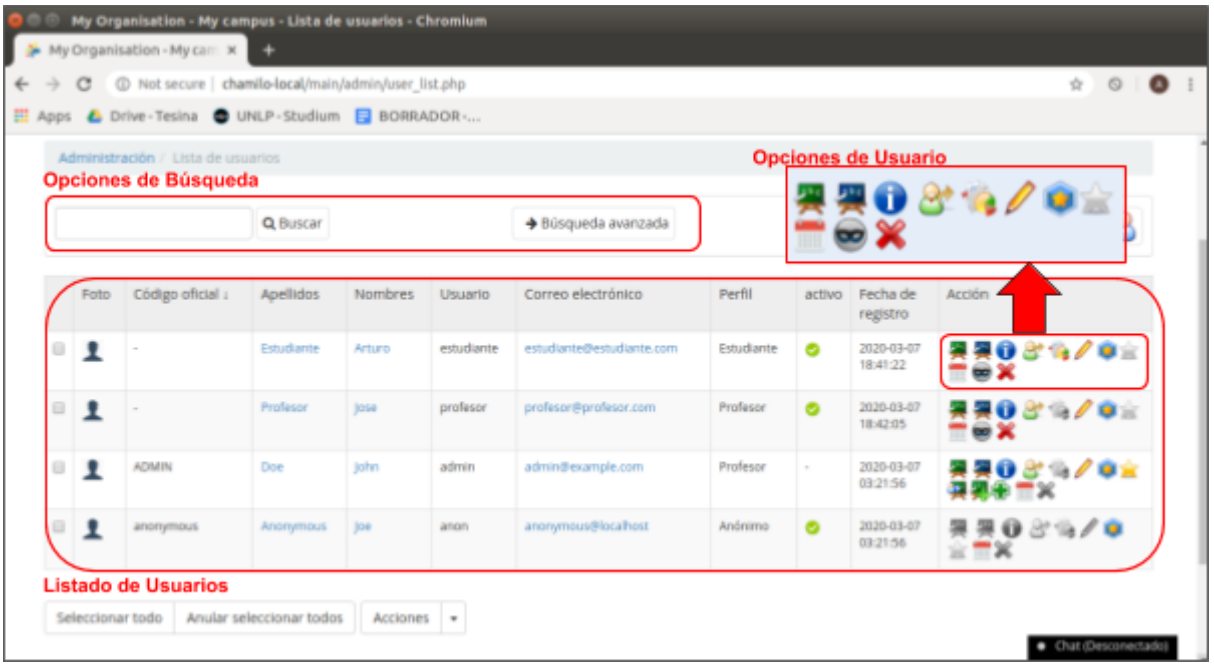


*Figura 15. Captura Navegador - Listado de Usuarios Chamilo.*

Una de las opciones presentes para un administrador, que puede observarse en la Figura 16, es el menú de administración de idiomas de la plataforma. Este permite seleccionar qué idiomas están disponibles en Chamilo para los usuarios. Adicionalmente también brinda la funcionalidad de definir un idioma predeterminado para el sistema.

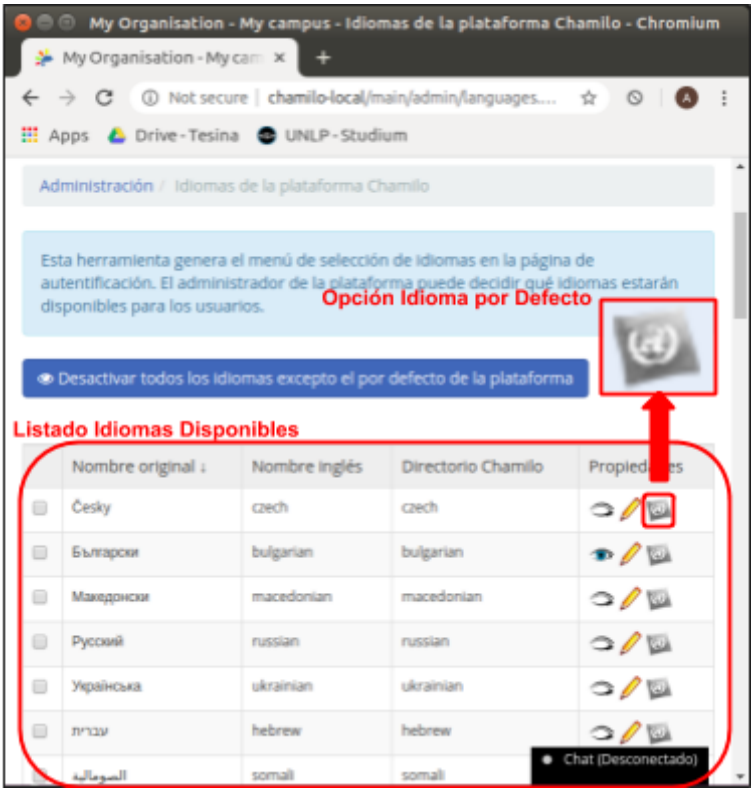


*Figura 16. Captura Navegador - Administración de Lenguajes Chamilo.*

Perfil Profesor

Los usuarios con perfil Profesor tienen como principal funcionalidad la creación de cursos y la gestión de su contenido. A su vez los profesores también puede registrarse en los cursos del sistema como estudiantes.

Como puede observarse en la Figura 17, los profesores pueden acceder a la opción de crear un curso desde su menú principal. Adicionalmente, desde esta pantalla, se puede acceder a la casilla de mensajes y editar el perfil de usuario y los datos personales, siendo estas funcionalidades propias de todos los perfiles de usuarios de Chamilo.

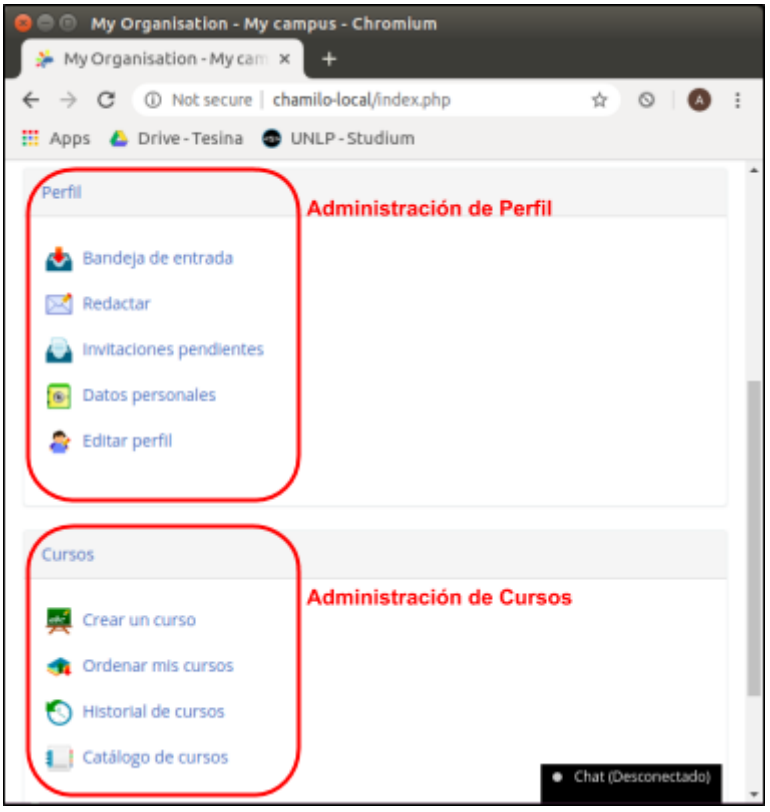


*Figura 17. Captura Navegador - Menú Principal Profesor Chamilo.*

Al crear un curso, éste se agrega automáticamente al listado de “Mis Cursos” del profesor, como se observa en la Figura 18. Este listado también muestra aquellos cursos a los que el profesor se halla registrado como estudiante y permite acceder a cada uno de ellos.

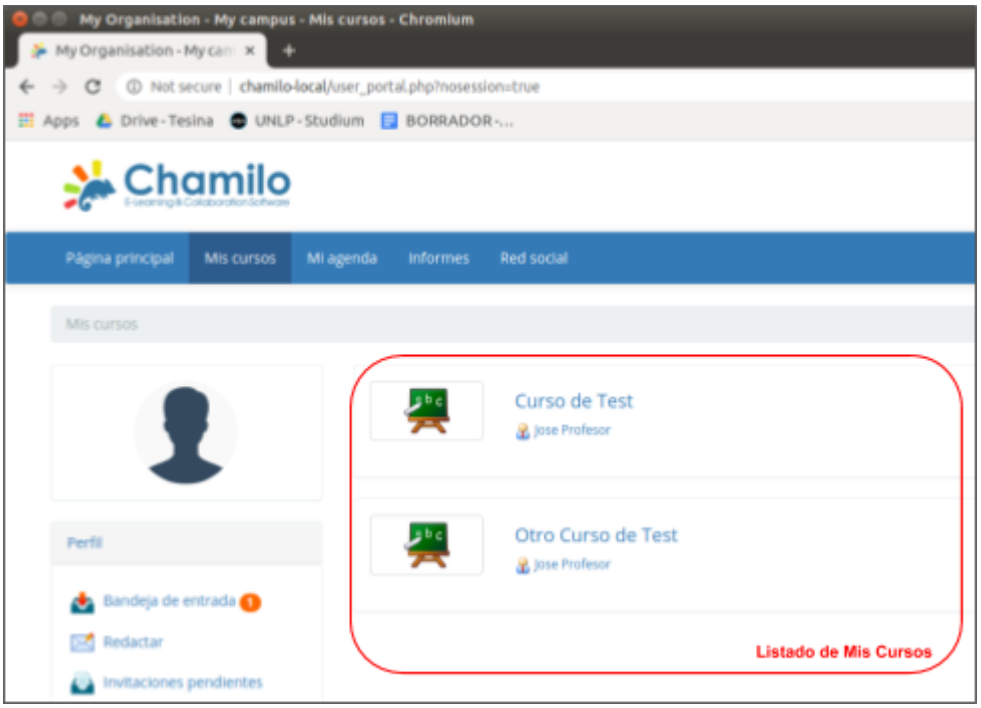


*Figura 18. Captura Navegador - Listado de Cursos Profesor Chamilo.*

En el menú de administración del curso, que se muestra en la Figura 19, los profesores pueden acceder a las pantallas de visualización y gestión de los diferentes tipos de contenidos del mismo. Los contenidos de un curso contienen documentos, lecciones, ejercicios, anuncios para los usuarios registrados al mismo y listas de enlaces, entre otras.

Este menú permite también definir qué tipos de contenidos estarán visibles para los estudiantes al acceder al curso.

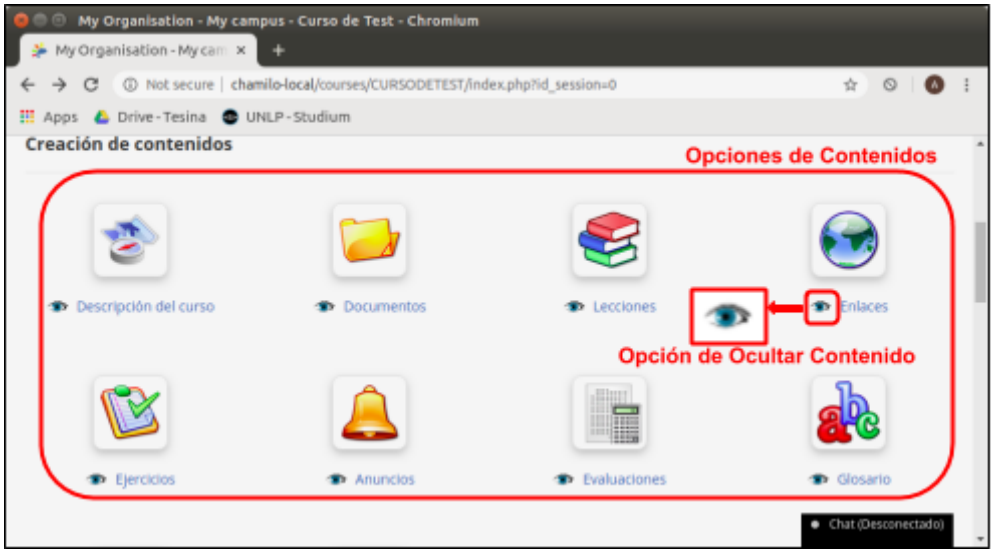


*Figura 19. Captura Navegador - Administración de Curso Chamilo.*

La administración de ejercicios de un curso, que puede observarse en la Figura 20, presenta una lista de los ejercicios disponibles. Desde esta página es posible acceder a

funcionalidad como A/B/M de ejercicios, creación de preguntas y visualización de resultados, entre otras.

Los ejercicios de Chamilo solo requieren de un nombre y una pregunta para ser creados. Adicionalmente se puede realizar una configuración más avanzada de los mismos configurando campos como la cantidad de intentos disponibles o la forma de mostrar el resultado.

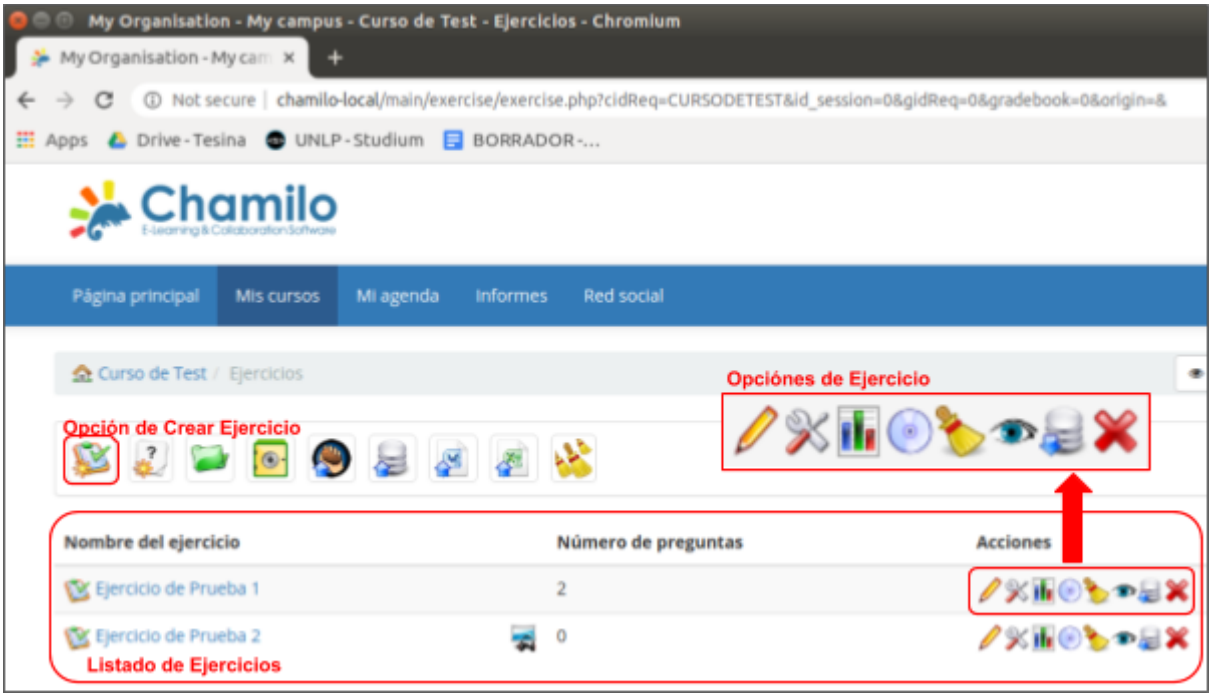


*Figura 20. Captura Navegador - Administración de Ejercicios Chamilo.*

Al editar las preguntas de un ejercicio se despliega un menú que muestra el listado de las preguntas incluidas en el ejercicio como se muestra en la Figura 21. Por cada pregunta del listado puede apreciarse el grado de dificultad, la puntuación máxima, y opciones para editar, copiar o eliminar cada pregunta.

La pantalla de “Administración de Preguntas” también muestra una serie de íconos que representan los tipos de preguntas ofrecidas por Chamilo. Para agregar una pregunta a un curso basta con hacer clic en el icono del tipo de pregunta deseado y el sistema redirige al formulario de creación correspondiente. Una vez que la pregunta es creada, ésta puede visualizarse en la lista de preguntas del ejercicio.

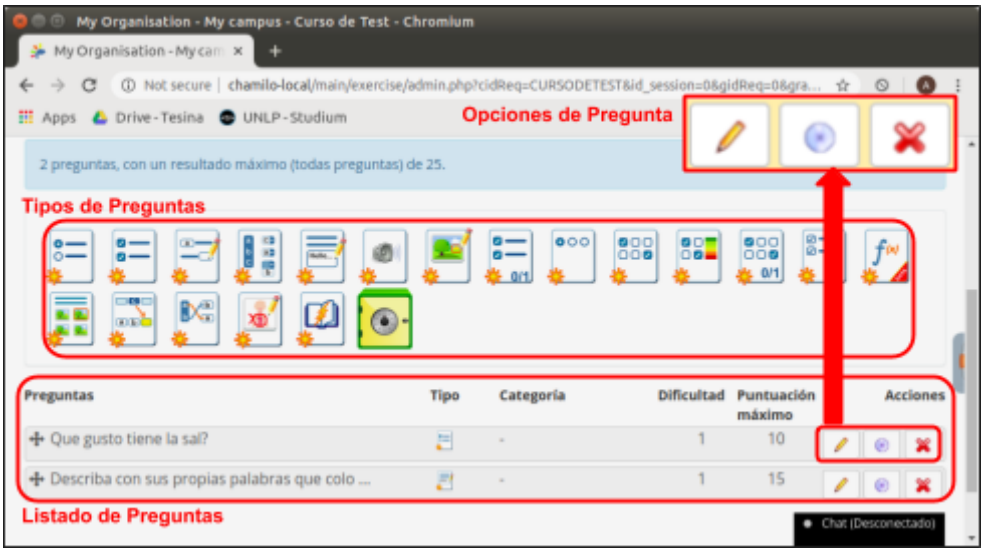


*Figura 21. Captura Navegador - Administración de Preguntas Chamilo.*

Al acceder a la sección de resultados de un ejercicio se muestra una lista de los intentos registrados por los estudiantes del curso al resolverlo. Como puede observarse en la Figura 22, la lista de resultados muestra el nombre del estudiante, la fecha de inicio y fin, la puntuación obtenida y el estado de corrección. A su vez ofrece acceso a funcionalidades como calificar, recalcular y eliminar resultado.

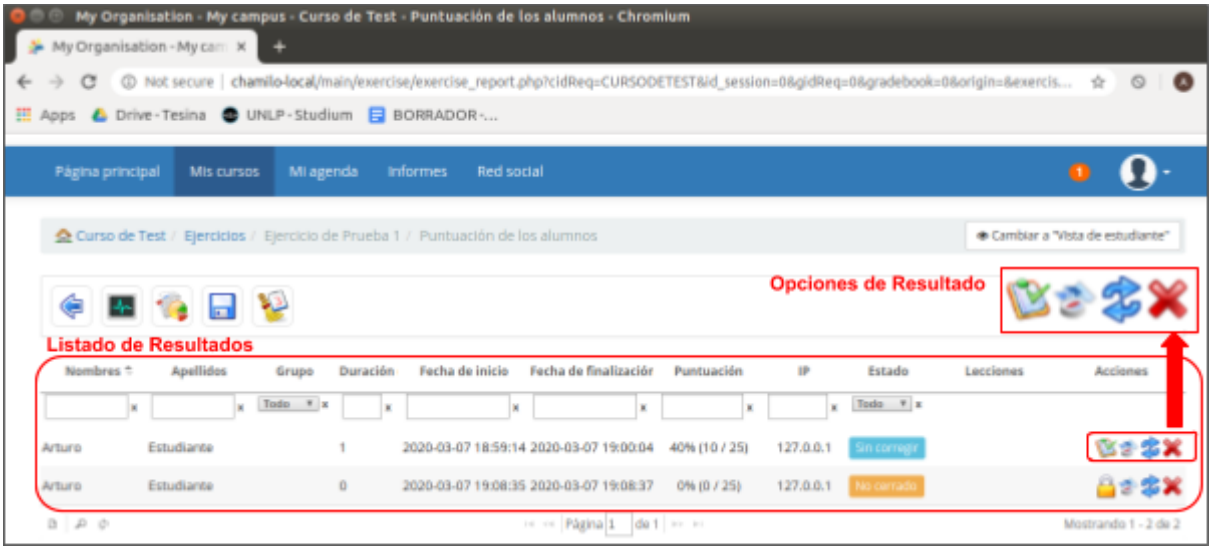


*Figura 22. Captura Navegador - Resultados de Ejercicios Chamilo.*

La corrección de un resultado muestra la lista de preguntas del ejercicio con las respectivas respuestas del estudiante, además de las opciones de corrección del profesor. Según el tipo de pregunta, y la configuración definida al crear la misma, las correcciones pueden realizarse automáticamente o no. En el caso de que una pregunta no cuente con corrección automática, el profesor deberá definir el puntaje de esa pregunta manualmente utilizando la opción de “Corregir y Puntuar”. Adicionalmente el profesor puede agregar comentarios a la corrección de cada de pregunta. Cuando todas las preguntas de un intento

son calificadas se envía automáticamente un informe al estudiante con su puntuación final y los comentarios recibidos.

Además de la funcionalidad mencionada, los profesores pueden acceder a otras funcionalidades dentro de un curso, entre ellas se puede mencionar la gestión de diferentes medios de comunicación para los estudiantes y profesores, tomar asistencia, gestionar una wiki, crear encuestas y obtener métricas de un curso.

### Perfil Estudiante

Desde el perfil de “Estudiante” un estudiante puede registrarse a un curso y buscar cursos del catálogo como puede verse en la Figura 23. El catálogo permite filtrar los cursos por categoría y realizar búsquedas por título.

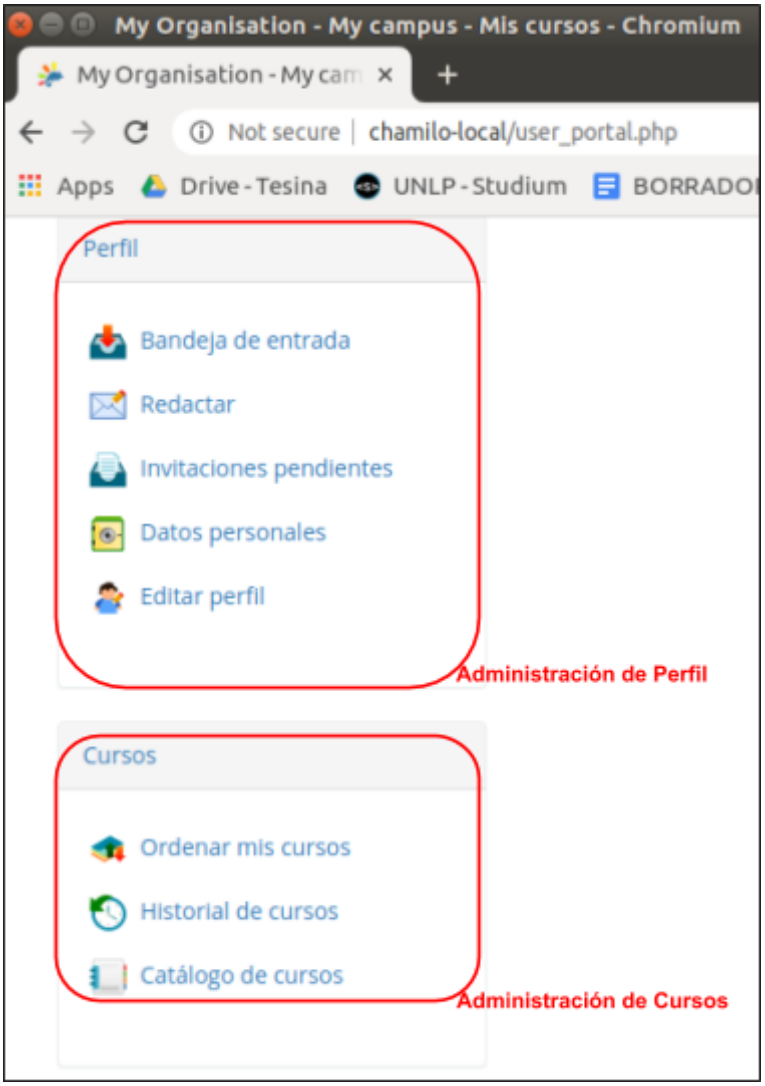


*Figura 23. Captura Navegador - Página Principal Estudiante Chamilo.*

Al ingresar a un ejercicio dentro de un curso, el estudiante puede comenzar la ejecución del mismo con el botón “Iniciar la prueba”. En esta pantalla se puede observar un listado de los intentos de resolución realizados, con su puntuación y estado, y acceder a los mismos.

Al ejecutar un ejercicio el estudiante accede a las preguntas de éste. Como se puede observar en la Figura 24, por cada pregunta se muestra su número, la consigna y los campos para completar la respuesta. Estos últimos varían según el tipo de pregunta. Cuando se finaliza el ejercicio, el sistema notifica al profesor que un intento ha sido registrado y se muestra el intento en la lista de la pantalla de visualización del ejercicio.

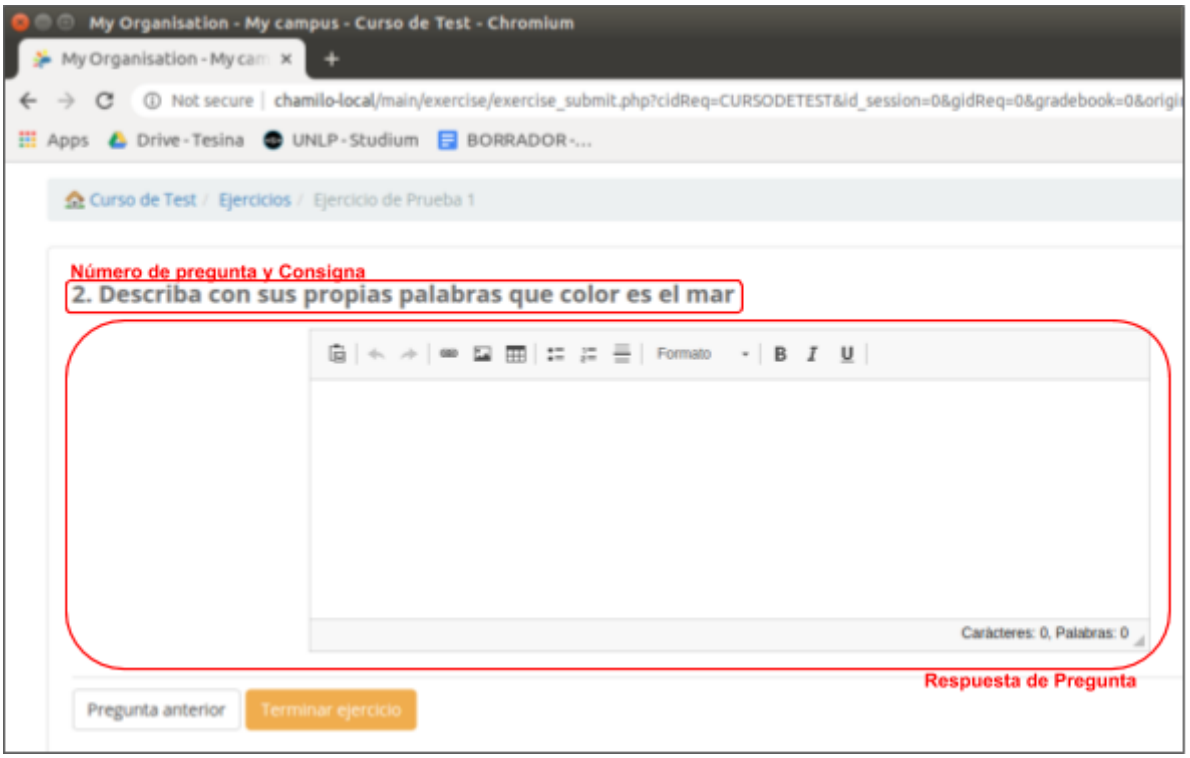


*Figura 24. Captura Navegador - Ejecución Ejercicio Chamilo.*

Adicionalmente, un estudiante puede acceder a todo el contenido del curso, compartir documentos, comunicarse con otros estudiantes y con el profesor, además de tomar notas personales, entre otras funcionalidades disponibles.

### 3.3.3 Conclusión

Chamilo LMS se presenta como un sistema con una funcionalidad completamente desarrollada y que cumple con todos los requisitos planteados para ser utilizado como base de Studium, tal y como puede observarse en la Tabla 4.

| | SI | PARCIALMENTE | NO |
|---|---|---|---|
| A/B/M Usuarios | X | | |
| A/B/M Cursos | X | | |
| A/B/M Ejercicios | X | | |
| Login/Logoff | X | | |
| Listado de Usuarios | X | | |
| Listado de Cursos | X | | |
| Listado de Ejercicios | X | | |
| Búsqueda de Cursos | X | | |
| Perfil Administrador | X | | |
| Perfil Estudiante | X | | |
| Perfil Profesor | X | | |
| Licencia Código Abierto | X | | |
| Funcionalidad Idioma Español | X | | |

*Tabla 4. Características Sistema Chamilo.*

Entre las funcionalidades más destacadas identificadas en este análisis pueden mencionarse que, a través de una interfaz de usuario sencilla e intuitiva, se tiene la posibilidad de gestionar completamente un curso, realizar A/B/M de usuarios con perfiles de profesores y estudiantes, generar ejercicios con múltiples tipos diferentes de preguntas definidas por el sistema y configurar la plataforma con el usuario administrador.

Chamilo permite cambiar el idioma del sistema fácilmente, restringir los idiomas disponibles y definir un idioma predeterminado para el sistema. Adicionalmente, presenta gran cantidad de funcionalidad para los usuarios dentro y fuera de los cursos, convirtiéndolo en un sistema que puede ser usado en un ambiente muy dinámico. También debe mencionarse que el código fuente de Chamilo es mantenido por una comunidad activa y que cuenta con una gran cantidad de usuarios que comparten información a través de diferentes páginas y foros.

Si bien Chamilo ofrece numerosas cualidades, se debe considerar que la documentación del mismo está disponible en forma completa sólo en idioma inglés. A su vez esta documentación no presenta gran detalle para los desarrolladores lo cual puede resultar en un problema en caso que su código tenga que ser modificado para agregar la nueva funcionalidad.

## 3.4 Conclusión Final

En función del análisis precedente, el criterio de selección del sistema LMS que será adoptado como base para Studium, tiene en cuenta que además de proveer las funcionalidades requeridas, sistematizadas en la Tabla 1, debe ofrecer las funcionalidades necesarias para integrar una herramienta de programación visual basada en bloques y contar con la capacidad de internacionalización dado que Studium estará disponible en español.

Python-LMS requiere implementar gran parte de la funcionalidad básica requerida de un sistema LMS además de realizar la integración de la herramienta de aprendizaje de programación que se plantea en este trabajo. Adicionalmente se debe traducir toda la funcionalidad del sistema al idioma español. Basándose en estos hechos se tomó la decisión de descartar Python-LMS para la implementación de Studium.

Si bien tanto ATutor como Chamilo presentan una funcionalidad de entorno de aprendizaje muy completa hay que considerar que ATutor tiene algunas fallas muy serias. Su principal desventaja se debe a que su código fuente ya no está mantenido por sus desarrolladores, se debería refaccionar el mismo para garantizar su funcionalidad básica antes de poder iniciar el desarrollo. En contraposición a Chamilo que cuenta con un grupo de desarrollo activo.

Adicionalmente Chamilo presenta una interfaz de usuario más intuitiva y simple y, un conjunto de herramientas destinadas a los docentes y administrativas más completa que ATutor.

A partir de este análisis se concluye que se utilizará el sistema Chamilo LMS como base para la implementación de Studium.

# Capítulo 4. Studium

## 4.1 Tecnologías utilizadas

En esta sección se mencionan las tecnologías más relevantes utilizadas durante el desarrollo de Studium. Adicionalmente se intenta explicar los motivos detrás de la elección de cada tecnología y para qué fue utilizada.

### GitHub

Para versionar el desarrollo y seguir los cambios que se fueron haciendo sobre la aplicación se decidió utilizar la plataforma de GitHub.

GitHub permite distribuir Studium a quien esté interesado y llevar cuenta de todos los incidentes de forma ordenada. Además brinda la ventaja de poder hacer rollback ante cualquier inconveniente y de continuar con el desarrollo del código desde cualquier estación de trabajo.

### Apache Web Server

El Servidor Web HTTP Apache fue utilizado para desplegar Studium y ejecutar los tests a lo largo del desarrollo. Un host virtual fue creado para realizar la instalación de la plataforma. Se consideró que este tipo de instalación era la más adecuada dado que es la recomendado para la instalación de la aplicación base, Chamilo.

### QuickForm y JQuery

Para la administración de los formulario utilizados en Chamilo se utiliza la tecnología QuickForm de Pear. Debido a que varios de estos formularios tuvieron que ser modificados durante el desarrollo, esta tecnología fue utilizada también para gestionar los nuevos formularios necesarios para la integración de Blockly Games dentro de la plataforma.

Adicionalmente se utiliza JQuery para lograr una respuesta dinámica de los formularios dentro de Studium. Teniendo en cuenta que Blockly Games se encuentra implementado principalmente en javascript, también se utilizó JQuery para modificar parte de la lógica de la herramienta.

### SOAP y Ajax

Chamilo ofrece servicios web ya implementados para los protocolos SOAP y REST. Después de analizar en detalle estas implementaciones se constató que el único servicio que funciona correctamente es el que implementa el protocolo SOAP. Considerando este hecho se utilizó el protocolo SOAP para implementar la comunicación entre Blockly Games y Chamilo.

Por otro lado, Chamilo utiliza Ajax para procesar algunos de sus formularios que requieren de la comunicación entre diferentes componentes del sistema. Se decidió continuar con este modelo para procesar alguna de la funcionalidad desarrollada en Studium.

Closure Templates

Durante el desarrollo fue necesario modificar las plantillas que implementan la interfaz de Blockly para restringir la navegabilidad de los usuarios y asegurarse que la herramienta se perciba como una parte integral de Studium.

Closure Template es el framework utilizado por Blockly Games para administrar sus vistas y fue el utilizado para realizar estas modificaciones.

Gimp

Gimp es un programa para edición de imágenes digitales libre y gratuito. Este fue utilizado para la creación de todos las imágenes relacionados a la nueva funcionalidad implementada en Studium, incluido el logo, el favicon y aquellos iconos que pueden verse en los diferentes menús del sistema.

## 4.2 Diseño: Casos de Uso

En esta sección se definen los casos de usos que describen el funcionamiento de Studium.

Debido a que no es el objetivo de esta tesina realizar un estudio exhaustivo de la funcionalidad ofrecida por Chamilo LMS o por Blockly Games, las acciones de usuarios detalladas en esta sección corresponden solamente a la funcionalidad relacionada a la lógica implementada durante el desarrollo de Studium.

Studium ofrece la posibilidad de crear varios tipos de perfiles de usuarios. La funcionalidad desarrollada solo afecta a los perfiles de **Profesor** y **Estudiante**, por lo que solo estos serán considerados en este análisis.

### 4.2.1 Actor Profesor

El actor **Profesor** es el encargado de crear y administrar los cursos que estarán disponibles en Studium. Las preguntas de Blockly Games fueron introducidas en Studium de forma tal que estas pudieran utilizarse dentro del contexto de un ejercicio. Como puede verse en la Figura 25 un profesor será capaz de crear o editar preguntas que integren un juego de Blockly Games, así como también visualizar y corregir las respuestas a dichas preguntas.


*Figura 25. Casos de Uso - Profesor.*

Crear Pregunta

Un profesor puede crear una pregunta de Blockly Games y agregarla a un ejercicio. Luego este ejercicio puede estar disponible a los estudiantes directamente, dentro de un curso, para su resolución. Por otro lado un ejercicio también puede ser incluido dentro de una lección de un curso con la funcionalidad provista por Chamilo.

| |
|---|
| **Nombre:** Crear Pregunta |
| **Descripción:** El Profesor crea una pregunta de tipo BlocklyQuestion. |
| **Actores:** Profesor, Sistema |
| **Precondición:** El Profesor debe haber creado un ejercicio previamente. |
| **Curso de Eventos:**<br>1. El Profesor accede al sistema con su usuario y contraseña.<br>2. El Sistema valida la información del usuario y redirige al menú Página Principal.<br>3. El Profesor hace clic en el menú “Mis Cursos”.<br>4. El Sistema redirige al menú de cursos del profesor y muestra una lista con los cursos disponibles.<br>5. El Profesor hace clic en el curso deseado.<br>6. El Sistema redirige al menú del curso seleccionado con las opciones disponibles.<br>7. El Profesor hace clic en la opción “Ejercicios”.<br>8. El Sistema redirige al menú de ejercicios y muestra la lista de ejercicios disponibles en el curso seleccionado.<br>9. El Profesor hace clic en el botón de Editar de un ejercicio.<br>10. El Sistema redirige al menú de edición del ejercicio mostrando la lista de preguntas disponibles.<br>11. El Profesor hace clic en el icono de “Juegos de Blockly Games”.<br>12. El Sistema redirige al menú de Agregar Pregunta del tipo seleccionado y muestra el formulario de creación de la pregunta. |

13. El Profesor completa los campos mandatorios y hace clic en el botón Añadir Pregunta.
14. El Sistema redirige al menú de edición del ejercicio mostrando la lista de preguntas del mismo.

**Postcondición:** El Profesor recibe el mensaje de “Elemento Añadido” y puede observar la nueva pregunta agregada a la lista de preguntas del ejercicio.

Editar Pregunta

Las pregunta creadas pueden ser modificadas en cualquier momento por el profesor siempre y cuando éstas no pertenezcan a un ejercicio añadido a una Lección. En este último caso no es posible modificar el ejercicio.

**Nombre:** Editar Pregunta

**Descripción:** El Profesor edita una pregunta de tipo BlocklyQuestion.

**Actores:** Profesor, Sistema

**Precondición:** El Profesor debe haber creado una pregunta de tipo BlocklyQuestion previamente. La pregunta creada no debe pertenecer a un ejercicio que haya sido agregado a una lección.

**Curso de Eventos:**

1. El Profesor accede al sistema con su usuario y contraseña.
2. El Sistema valida la información del usuario y redirige al menú Página Principal.
3. El Profesor hace clic en el menú “Mis Cursos”.
4. El Sistema redirige al menú de cursos del profesor y muestra una lista con los cursos disponibles.
5. El Profesor hace clic en el curso deseado.
6. El Sistema redirige al menú del curso seleccionado con las opciones disponibles.
7. El Profesor hace clic en la opción “Ejercicios”.
8. El Sistema redirige al menú de ejercicios y muestra la lista de ejercicios disponibles en el curso seleccionado.
9. El Profesor hace clic en el botón de Editar de un ejercicio.
10. El Profesor hace clic en el botón de Editar de un ejercicio.
11. El Sistema redirige al menú de edición del ejercicio mostrando la lista de preguntas del mismo.
12. El Profesor hace clic en el icono para modificar la pregunta.
13. El Sistema redirige al menú de Modificar Pregunta del tipo seleccionado y muestra el formulario de la pregunta.
14. El Profesor modifica los campos deseados y hace clic en el botón “Modificar Pregunta”.
15. El Sistema redirige al menú de edición del ejercicio mostrando la lista de preguntas del mismo.

**Postcondición:** El Profesor recibe el mensaje de “Elemento actualizado” y puede observar la nueva pregunta agregada a la lista de preguntas del ejercicio.

Visualizar y Corregir Resultado

Un profesor recibe una notificación de aviso por mail cada vez que un ejercicio fue completado por un estudiante. Luego pueden visualizarse la respuestas a las preguntas en Blockly Games y realizar la corrección de las mismas en Studium.

Para la corrección de una respuesta se debe introducir un puntaje y un comentario relacionado a cada pregunta de un ejercicio. La corrección finaliza notificando al estudiante, por mail, que se encuentra disponible.

| |
|---|
| **Nombre:** Visualizar y Corregir Resultado |
| **Descripción:** El Profesor visualiza y corrige el resultado de una respuesta a una pregunta tipo BlocklyQuestion. |
| **Actores:** Profesor, Sistema |
| **Precondición:** Un ejercicio con una pregunta tipo BlocklyQuestion ha sido contestado por uno o más Estudiantes. |
| **Curso de Eventos:**<br>1. El Profesor accede al sistema con su usuario y contraseña.<br>2. El Sistema valida la información del usuario y redirige al menú Página Principal.<br>3. El Profesor hace clic en el menú “Mis Cursos”.<br>4. El Sistema redirige al menú de cursos del profesor y muestra una lista con los cursos disponibles.<br>5. El Profesor hace clic en el curso deseado.<br>6. El Sistema redirige al menú del curso seleccionado con las opciones disponibles.<br>7. El Profesor hace clic en la opción “Ejercicios”.<br>8. El Sistema redirige al menú de ejercicios y muestra la lista de ejercicios disponibles en el curso seleccionado.<br>9. El Profesor hace clic en el botón de Resultados de un ejercicio.<br>10. El Sistema redirige al menú de “Puntuacion de los Alumnos” mostrando la lista de respuestas al ejercicio.<br>11. El Profesor hace clic en el icono de “Calificar” de una respuesta no corregida.<br>12. El Sistema redirige al menú de Resultado y muestra la lista de preguntas del ejercicio con sus respuestas a corregir.<br>13. El Profesor hace clic en el link “Haga clic” para ir al juego de la respuesta a la pregunta tipo BlocklyQuestion.<br>14. El Sistema abre el juego de Blockly Games en otra solapa o ventana y hace foco en ella.<br>15. El Profesor valida que el ejercicio haya sido realizado correctamente y hace clic en el botón Volver al ejercicio de Studium.<br>16. El Sistema cierra la ventana o solapa con el juego de Blockly Games y hace foco en la ventana del menú de Resultado del ejercicio que se está corrigiendo.<br>17. El usuario hace clic en el botón Corregir y puntuar de la pregunta tipo BlocklyQuestion.<br>18. El Sistema muestra el formulario de corrección para esa pregunta con un campo de comentario y un campo de Puntaje.<br>19. El Profesor completa ambos campos del formulario y marca el checkbox “Enviar |

mail”.
20. El Sistema muestra el texto del mail a ser enviado al Estudiante que creó la respuesta que está siendo corregida.
21. El Profesor hace clic en el botón Corregir este ejercicio.
22. El Sistema redirige al menú de Puntuacion de los Alumnos mostrando la lista de respuestas al ejercicio.

**Postcondición:** El Profesor recibe el mensaje de Mensaje enviado y Lección actualizada. El Profesor puede observar la respuesta corregida con estado Corregido en la lista de respuestas al ejercicio. El Estudiante recibe el mail de notificación.

### 4.2.2 Actor Estudiante

El actor **Estudiante** es el usuario que ejecuta los juegos de Blockly Games dentro del contexto de los ejercicios de un curso en el que se encuentra registrado.

Como puede observarse en la Figura 26, el estudiante tiene la posibilidad de contestar las preguntas que integren un juego de Blockly Games directamente en un ejercicio o dentro de una lección. Además el estudiante podrá visualizar los resultados registrados y el puntaje obtenido para cada pregunta según donde se ubica el ejercicio.

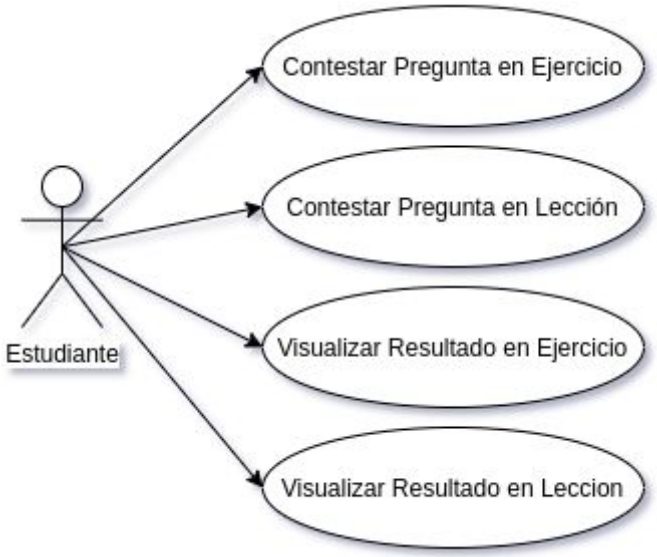


*Figura 26. Casos de Uso - Estudiante.*

Contestar Pregunta en Ejercicio

Al contestar una pregunta de Blockly Games, directamente dentro de un ejercicio, el estudiante será redirigido al juego. Dentro del mismo el estudiante podrá utilizar los bloques o el editor de javascript, según corresponda, para completar la consigna, probar el resultado y finalmente volver al ejercicio de Studium. Al retornar al ejercicio se guardará el estado del juego como respuesta a la pregunta.

**Nombre:** Contestar Pregunta en Ejercicio

**Descripción:** El estudiante responde un ejercicio con una pregunta tipo BlocklyQuestion y guarda el resultado.

**Actores:** Estudiante, Sistema

**Precondición:** El Estudiante debe estar registrado en un curso con un ejercicio con una pregunta de tipo BlocklyQuestion.

**Curso de Eventos:**

1. El Estudiante accede al sistema con su usuario y contraseña.
2. El Sistema valida la información del usuario y redirige al menú Página Principal.
3. El Estudiante hace clic en el menú “Mis Cursos”.
4. El Sistema redirige al menú de cursos del estudiante y muestra una lista con los cursos en los que se encuentra inscripto.
5. El Estudiante hace clic en el nombre del curso deseado.
6. El Sistema redirige al menú del curso seleccionado con las opciones disponibles.
7. El Estudiante hace clic en la opción Ejercicios.
8. El Sistema redirige al menú de ejercicios y muestra la lista de ejercicios disponibles en el curso seleccionado.
9. El Estudiante hace clic en el ejercicio a ser ejecutado.
10. El Sistema redirige al menú del ejercicio y muestra la lista de intentos realizados por el estudiante.
11. El Estudiante hace clic en el botón Iniciar la prueba.
12. El Sistema inicia el intento de resolución del ejercicio y redirige a la primera pregunta del mismo.
13. El Estudiante hace clic en el botón “Haga clic” aquí para ir al juego de la pregunta BlocklyQuestion.
14. El Sistema redirige al juego de Blockly Games relacionado a la pregunta de tipo BlocklyQuestion.
15. El Estudiante completa el ejercicio y hace clic en el botón “Ejecutar el programa” para validar la respuesta.
16. El Sistema ejecuta el programa escrito por el Estudiante y emite una alerta indicando que el programa fue ejecutado exitosamente y muestra las líneas de código en javascript del programa ejecutado.
17. El Estudiante hace clic en el botón Volver al ejercicio de Studium.
18. El Sistema muestra el mensaje “El juego fue guardado con éxito”.
19. El Sistema redirige a la pregunta del ejercicio relacionada al juego ejecutado y muestra el mensaje: “Esta pregunta ya tiene una respuesta guardada. Si desea modificarla vuelva a hacer clic en el botón para ir al juego. Sino continúe con el ejercicio”.
20. El Estudiante hace clic en el botón “Terminar Ejercicio”.
21. El Sistema finaliza el intento de respuesta y redirige a la pantalla de Resultado mostrando la lista de respuestas del ejercicio con sus respuestas.

**Postcondición:** El Estudiante es capaz de guardar el ejercicio con éxito y ver el menú de Resultado con el mensaje de Guardado. El Profesor recibe un mail con el mensaje “Un estudiante ha contestado una pregunta” y la referencia a la respuesta generada por el Estudiante.

Contestar Pregunta en Lección

Si el estudiante responde a la pregunta dentro de una lección, el juego se ejecutará en la misma ventana de la lección sin necesidad de ser redirigido fuera de Studium.

| |
|---|
| **Nombre:** Contestar Pregunta en Lección |
| **Descripción:** El estudiante responde un ejercicio con una pregunta tipo BlocklyQuestion dentro de una lección y guarda el resultado. |
| **Actores:** Estudiante, Sistema |
| **Precondición:** El Estudiante debe estar registrado en un curso con una lección que cuenta con un ejercicio con una pregunta de tipo BlocklyQuestion. |
| **Curso de Eventos:**<br>1. El Estudiante accede al sistema con su usuario y contraseña.<br>2. El Sistema valida la información del usuario y redirige al menú Página Principal.<br>3. El Estudiante hace clic en el menú Mis Cursos.<br>4. El Sistema redirige al menú de cursos del estudiante y muestra una lista con los cursos en los que se encuentra inscripto.<br>5. El Estudiante hace clic en el nombre del curso deseado.<br>6. El Sistema redirige al menú del curso seleccionado con las opciones disponibles.<br>7. El Estudiante hace clic en la opción Lecciones.<br>8. El Sistema redirige al menú de lecciones y muestra la lista de lecciones disponibles en el curso seleccionado.<br>9. El Estudiante hace clic en el nombre de la lección a ser ejecutada.<br>10. El Sistema redirige a la vista de la lección y muestra la lista de ítems de la lección seleccionada en la izquierda de la ventana y el contenido del primer ítem en la derecha de la ventana.<br>11. El Estudiante hace clic en el ítem que contiene el ejercicio.<br>12. El Sistema muestra el ejercicio en la parte derecha de la ventana.<br>13. El Estudiante hace clic en el botón Iniciar la prueba.<br>14. El Sistema inicia el intento de resolución del ejercicio y redirige a la primera pregunta del mismo en la parte derecha de la ventana.<br>15. El Estudiante hace clic en el botón “Haga clic aquí” para ir al juego de la pregunta BlocklyQuestion.<br>16. El Sistema ejecuta al juego de Blockly Games relacionado a la pregunta de tipo BlocklyQuestion en la parte derecha de la pantalla.<br>17. El Estudiante completa el ejercicio y hace clic en el botón Ejecutar el programa para validar la respuesta.<br>18. El Sistema ejecuta el programa escrito por el Estudiante y emite una alerta indicando que el programa fue ejecutado exitosamente y mostrando las líneas de código del programa ejecutado en javascript.<br>19. El Estudiante hace clic en el botón Volver al ejercicio de Studium.<br>20. El Sistema muestra el mensaje El juego fue guardado con éxito.<br>21. El Sistema redirige a la pregunta del ejercicio relacionada al juego ejecutado en la parte derecha de la ventana y muestra el mensaje: “Esta pregunta ya tiene una respuesta guardada. Si desea modificarla vuelva a hacer clic en el botón para ir al |

juego. Sino continúe con el ejercicio".
22. El Estudiante hace clic en el botón "Terminar Ejercicio".
23. El Sistema finaliza el intento de respuesta y redirige a la pantalla de Resultado en la parte derecha de la pantalla mostrando la lista de preguntas del ejercicio con sus respuestas.

**Postcondición:** El Estudiante es capaz de guardar el ejercicio con éxito y ver el menú de Resultado con el mensaje de Guardado. El porcentaje de la barra de progreso de la lección se actualiza. El Profesor recibe un mail con el mensaje "Un estudiante ha contestado una pregunta" y la referencia a la respuesta generada por el Estudiante.

Visualizar Resultado en Ejercicio

Luego de finalizar un ejercicio el estudiante tendrá la posibilidad de visualizar la respuestas a cada pregunta en Blockly Games según el contexto en el que haya completado el ejercicio. Es importante destacar que si bien podrá visualizarlas, no podrá modificarlas.

Una vez esté disponible la corrección del profesor, el estudiante podrá ver los comentarios y el puntaje recibido en cada pregunta al visualizar la respuesta.

**Nombre:** Visualizar Resultado de Pregunta en Ejercicio

**Descripción:** El estudiante visualiza el resultado de una pregunta tipo BlocklyQuestion dentro de una lección.

**Actores:** Estudiante, Sistema

**Precondición:** El Estudiante debe haber finalizado un ejercicio con una pregunta de tipo BlocklyQuestion.

**Curso de Eventos:**
1. El Estudiante accede al sistema con su usuario y contraseña.
2. El Sistema valida la información del usuario y redirige al menú Página Principal.
3. El Estudiante hace clic en el menú "Mis Cursos".
4. El Sistema redirige al menú de cursos del estudiante y muestra una lista con los cursos en los que se encuentra inscripto.
5. El Estudiante hace clic en el nombre del curso deseado.
6. El Sistema redirige al menú del curso seleccionado con las opciones disponibles.
7. El Estudiante hace clic en la opción Ejercicios.
8. El Sistema redirige al menú de ejercicios y muestra la lista de ejercicios disponibles en el curso seleccionado.
9. El Estudiante hace clic en el ejercicio a ser ejecutado.
10. El Sistema redirige al menú del ejercicio y muestra la lista de intentos realizados por el estudiante.
11. El Estudiante hace clic en el botón Mostrar del intento del que desea ver el resultado.
12. El Sistema redirige a la pantalla de Resultado mostrando la lista de preguntas del ejercicio con sus respuestas.
13. El Estudiante hace clic en el botón "Haga clic aquí" para ir al juego de la pregunta

BlocklyQuestion.
14. El Sistema abre el juego de Blockly Games en otra pestaña o ventana y hace foco en ella.
15. El Estudiante hace clic en el botón Volver al ejercicio de Studium.
16. El Sistema cierra la ventana o la pestaña con el juego de Blockly Games y hace foco en la ventana del menú de Resultado del ejercicio que se está visualizando.

**Postcondición:** El Estudiante es capaz de visualizar el resultado de una pregunta de tipo BlocklyQuestion en Blockly Games y retornar al ejercicio de Studium exitosamente.

Visualizar Resultado en Lección

Si un estudiante responde a un ejercicio dentro del contexto de una lección, sólo podrá visualizar la respuesta del ejercicio accediendo a través de la lección. Una vez que la corrección del ejercicio esté disponible se podrán visualizar los comentarios y el puntaje recibidos, mostrando la respuesta de la misma manera.

**Nombre:** Visualizar Resultado en Lección

**Descripción:** El estudiante visualiza el resultado de una pregunta tipo BlocklyQuestion dentro de una lección.

**Actores:** Estudiante, Sistema

**Precondición:** El Estudiante debe haber finalizado un ejercicio con una pregunta de tipo BlocklyQuestion dentro de una lección.

**Curso de Eventos:**
1. El Estudiante accede al sistema con su usuario y contraseña.
2. El Sistema valida la información del usuario y redirige al menú Página Principal.
3. El Estudiante hace clic en el menú “Mis Cursos”.
4. El Sistema redirige al menú de cursos del estudiante y muestra una lista con los cursos en los que se encuentra inscripto.
5. El Estudiante hace clic en el nombre del curso deseado.
6. El Sistema redirige al menú del curso seleccionado con las opciones disponibles.
7. El Estudiante hace clic en la opción Lecciones.
8. El Sistema redirige al menú de lecciones y muestra la lista de lecciones disponibles en el curso seleccionado.
9. El Estudiante hace clic en el nombre de la lección a ser ejecutada.
10. El Sistema redirige a la vista de la lección y muestra la lista de ítems de la lección seleccionada en la izquierda de la ventana y el contenido del primer ítem en la derecha de la ventana.
11. El Estudiante hace clic en el ítem que contiene el ejercicio.
12. El Sistema muestra el ejercicio en la parte derecha de la ventana con la lista de intentos realizados por el estudiante.
13. El Estudiante hace clic en el botón Mostrar del intento del que desea ver el resultado.
14. El Sistema muestra la pantalla de Resultado en la parte derecha de la venta, mostrando la lista de preguntas del ejercicio con sus respuestas.
15. El Estudiante hace clic en el botón “Haga clic aquí” para ir al juego de la pregunta

BlocklyQuestion.
16. El Sistema abre el juego de Blockly Games en otra pestaña o ventana y hace foco en ella.
17. El Estudiante hace clic en el botón Volver al ejercicio de Studium.
18. El Sistema cierra la ventana o la pestaña con el juego de Blockly Games y hace foco en la ventana del menú de Resultado del ejercicio que se está visualizando.

**Postcondición:** El Estudiante es capaz de visualizar el resultado de una pregunta de tipo BlocklyQuestion en Blockly Games y retornar a la lección de Studium exitosamente.

## 4.3 Diseño: Interacción y Diagramas de Secuencia

En esta sección se presentan los diagramas de secuencia que se desarrollan a partir de la ejecución de las diferentes acciones de los usuarios en Studium.

Para demostrar la interacción de las funcionalidades desarrolladas a partir de la implementación de este sistema, se plantean 3 grandes procesos:

- Crear una pregunta de Blockly Games.
- Contestar una pregunta de Blockly Games.
- Corregir una pregunta de Blockly Games.

Adicionalmente, mediante el análisis de los diagramas de estos procesos se busca brindar un contexto para el análisis más detallado de la lógica contenida en los diferentes componentes que forman parte de Studium.

Es importante mencionar que los diagramas de flujos presentados en esta sección fueron simplificados para permitir una mejor legibilidad. Para esto se hizo foco en los procesos más importantes de cada uno. Al considerar esta decisión se tuvo en cuenta la gran complejidad que presenta el código de Chamilo y el hecho que no es el objetivo de este trabajo analizar en detalle la funcionalidad de Chamilo.

### 4.3.1 Crear Pregunta de Blockly Games

En el diagrama de la Figura 27, podemos observar cómo el sistema genera el formulario de creación de una pregunta a partir de la acción de clic en el botón de “Juego de Blockly Games” dentro del menú de administrar ejercicio.

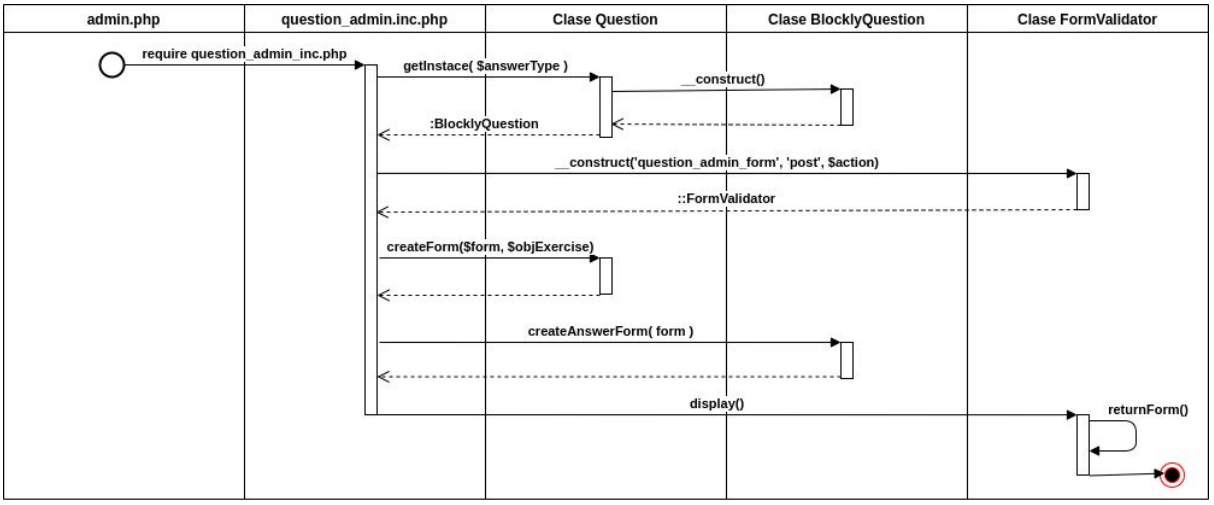


*Figura 27. Diagrama de Flujo - Creación de Formulario de Agregar Pregunta.*

El botón inicia la acción de agregar una nueva pregunta de Blockly Games a un ejercicio. El script **admin.php** se encarga de iniciar el proceso de crear el formulario para agregar la pregunta. Para hacer esto incluye el script **question_admin.inc.php**. La lógica incluida se utiliza tanto al crear como modificar una pregunta.

El script **question_admin.inc.php** comienza usando el método **getInstance()** de la clase **Question** para instanciar una pregunta**.** Este método devuelve una instancia de la subclase de **Question** correspondiente al tipo de pregunta pasado por parámetro, siendo en este caso una pregunta de Blockly Games.

El formulario a ser utilizado para la creación de la pregunta es instanciado por la clase **FormValidator** utilizando los métodos de instancia de la pregunta creada previamente. Esta clase extiende la clase de PEAR **HTML_QuickForm** e implementa varios métodos útiles para manejar este tipo de formularios dentro de Chamilo. El formulario es creado pasando como parámetro el nombre del mismo, el método utilizado y la acción que se ejecutará al procesarlo.

El formulario luego es modificado para agregar los campos necesarios para la creación de la nueva pregunta. El script **question_admin.inc.php** utiliza el método **createForm()** implementado en la clase **Question** para agregar al formulario todos los campos generales de una pregunta de Chamilo. Se utiliza el método **createAnswerForm()** de la clase **BlocklyQuestion** para añadir los campos propios del tipo de pregunta de Blockly Games y modificar algunos de los campos agregados por la clase **Question**.

Finalmente, utilizando el método **display()**, el script muestra el formulario en pantalla reemplazando el contenido del menú de administración de ejercicio desde donde se inició el proceso.

El usuario puede completar el formulario con la información de la nueva pregunta de Blockly Games y enviar el mismo iniciando el proceso que se muestra en la Figura 28.

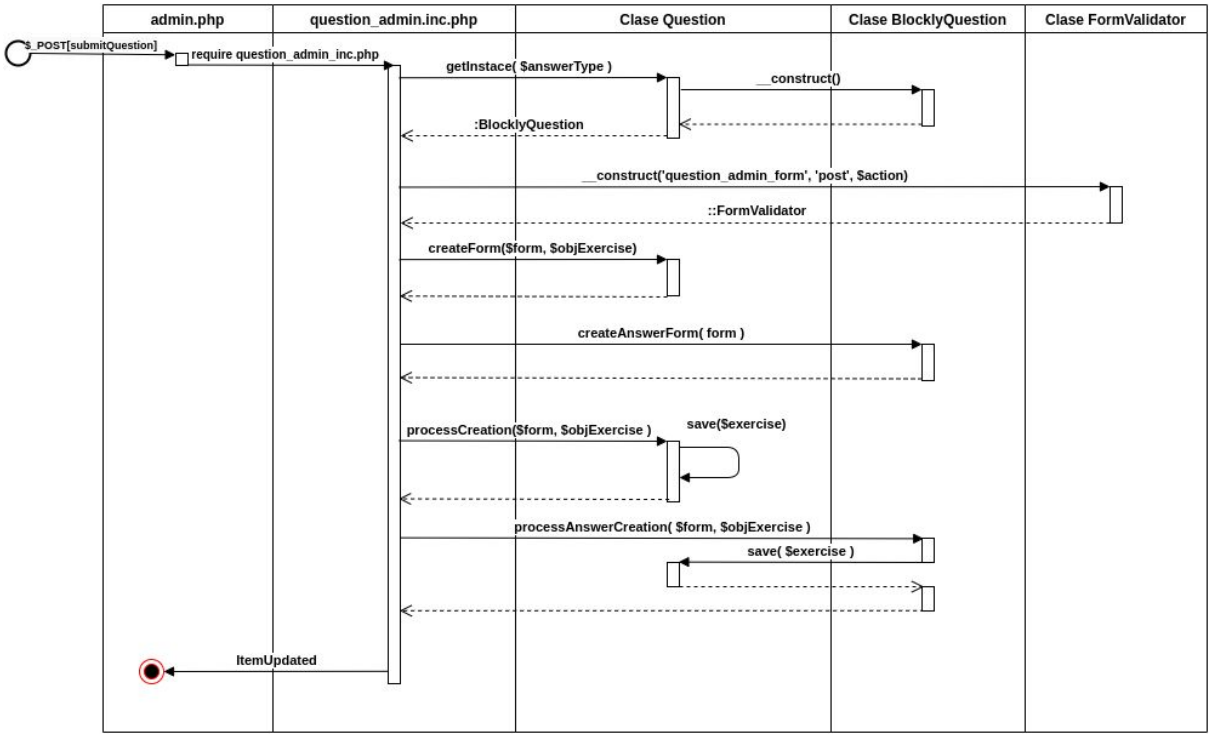


*Figura 28. Diagrama de Flujo - Procesar Formulario de Agregar Pregunta.*

Al enviar el formulario de creación de pregunta, el script **admin.php** recibe el mensaje POST e incluye nuevamente el script **question_admin.inc.php** para que procese el pedido. El script incluido instancia una pregunta de tipo de Blockly Games y un formulario de creación para este tipo de pregunta. A diferencia del proceso anterior, en esta ocasión los campos de la instancia del formulario son inicializados con los valores recibidos a través del requerimiento POST.

Utilizando la instancia de la pregunta creada previamente, el script recupera los datos de los campos generales del formulario utilizando el método **processCreation()**. Este método almacena esos datos en la instancia de la pregunta utilizada para invocarlo.

Luego, usando el método **processAnswerCreation()**, se recuperan en la misma instancia de pregunta los datos contenidos en los campos propios del tipo de pregunta de Blockly Games del formulario. Para guardar la pregunta en la base de datos y actualizar la lista de preguntas del ejercicio, se utiliza el método **save()** implementado en la clase **Question()**.

Finalmente **question_admin.inc.php** ejecuta un script que redirige al menú de administración del ejercicio modificado mostrando el mensaje de “Elemento Añadido”.

### 4.3.2 Contestar Pregunta de Blockly Games

El botón de “Iniciar Ejercicio” dentro del menú de visualización de un ejercicio, permite a un estudiante comenzar un intento de respuesta del mismo. Para generar el formulario de respuesta de la primera pregunta del ejercicio, se sigue el proceso mostrado en la Figura 29.

El script **exercise_submit** comienza generando una instancia temporal de la clase Exercise. Esta instancia se usará para obtener toda la información del ejercicio de la base de datos, incluida su descripción y su lista de preguntas.

Una vez obtenida esta información, el script muestra la primera pregunta utilizando el método **showQuestion()** de la clase **ExerciseLib**. Este método obtiene la información de la pregunta a responder y la almacena en una instancia de la misma usando el método **read()** de la clase **Question**.

Luego el método instancia el formulario a utilizarse para responder la pregunta. Para esto se utiliza el método **__construct** de la clase **FormValidator**, que es invocado sin especificar el método ni la acción a ejecutarse al enviar el formulario.

El formulario obtenido es modificado por la clase **ExerciseLib** que agrega los campos necesarios para la respuesta según el tipo de pregunta de la instancia obtenida anteriormente usando la clase **Question**. Finalmente, esta clase procede a mostrar el formulario en pantalla en formato html usando **return Form()** de la clase **FormValidator**. Usando este formulario el usuario puede comenzar a responder la pregunta.

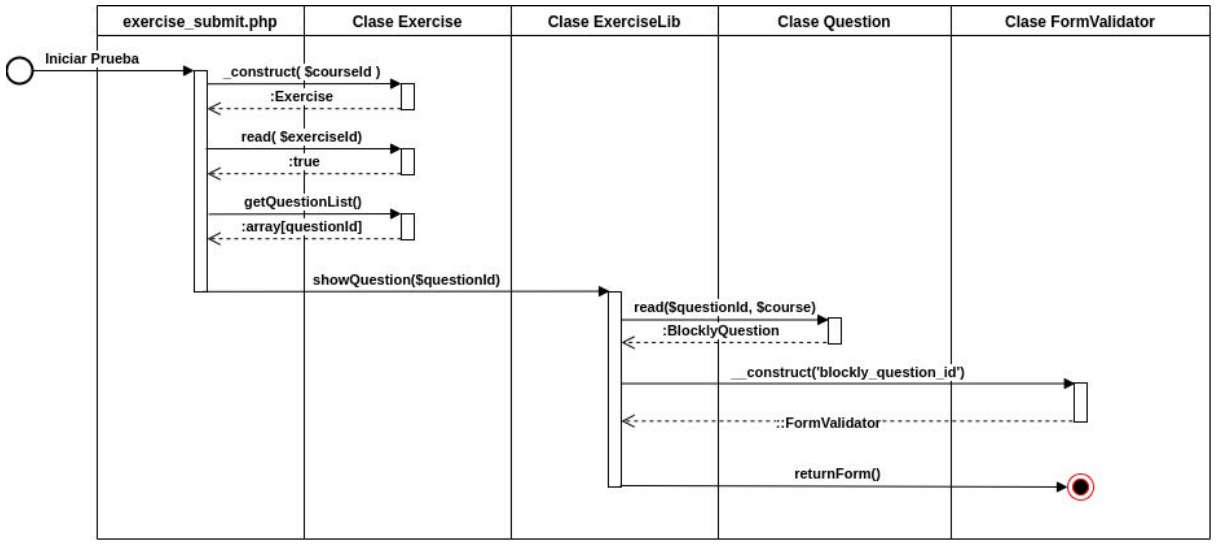


*Figura 29. Diagrama de Flujo - Creación de Formulario de Contestar Pregunta.*

El botón “Haz clic aquí para ir a el juego” de la interfaz de la pregunta de Blockly Games inicializa el proceso que lleva a ejecutar el juego relacionado a la misma, como muestra la Figura 30. El script **exercise_submit.php** implementa un evento de JQuery para este botón que crea un intento de respuesta en la base de datos utilizando un request de Ajax para el método **save_exercise_by_now** del script **exercise.ajax.php**. Luego redirige al juego de Blockly Games.

Para este ejemplo, se utilizó el juego “Laberinto” de Blockly Games, por lo que se redirige a su pantalla principal **maze.html.** El juego se cargará en el lenguaje y el nivel indicado en la url usada para la redirección. En este punto el usuario puede interactuar con el juego y generar su programa en bloques.

Al finalizar la ejecución del juego, el usuario hará clic en el botón “Volver al ejercicio de Studium”. Este botón está ligado a la función **link()** implementada en el espacio de nombres **BlocklyStorage**, que es la encargada de guardar el estado del juego en la base de datos de Studium.

La función **link()** obtiene el código del editor del código del juego, utilizando el método **getCode()** del espacio de nombres **BlocklyInterface**. Luego la función invoca **getSoapWSSaveBlocklyAttemptBody()** para generar el contenido en formato xml que se utilizará para acceder al servicio web de Studium. Una vez obtenidos estos datos, serán utilizados para invocar a la función **soapRequest()**. Esta función genera un pedido SOAP para la ejecución del método **WSSaveBlocklyAttempt** que es atendido por el servicio web **WSBlocklyGames** de Studium. El servicio web obtiene la información enviada en el pedido SOAP y actualiza el intento de respuesta de la pregunta, almacenado en la base de datos, con el código implementado durante la ejecución del juego de Blockly Games. Para modificar la base de datos se utiliza el método **saveQuestionAttempt()** de la clase **Event**. Una vez realizada esta operación se devuelve la respuesta al pedido SOAP. Al recibir la respuesta del servicio web de Studium se emite una mensaje para informar al usuario que la respuesta a la pregunta fue guardada con éxito. Para esto se utiliza el método **alert()** implementado en el espacio de nombres **BlocklyStorage**. Este mensaje es interceptado por la función **storageAlert()** del espacio de nombres **BlocklyDialogs** quien le da formato a la ventana donde se muestra el mensaje y redirecciona a la pregunta dentro del ejercicio de Studium cuando el usuario cierra esa ventana.

Al cargar nuevamente la pregunta del ejercicio desde **exercise_submit.php**, ésta mostrará un mensaje indicando que la pregunta ya tiene una respuesta guardada debajo del botón que inició este proceso. Para terminar el ejercicio, el usuario deberá presionar el botón “Terminar Ejercicio” que ejecutará la función de JQuery **save_now()** que notifica a **exercise_submit.php** que se finalizó la ejecución y redirige a la pantalla de resultados.

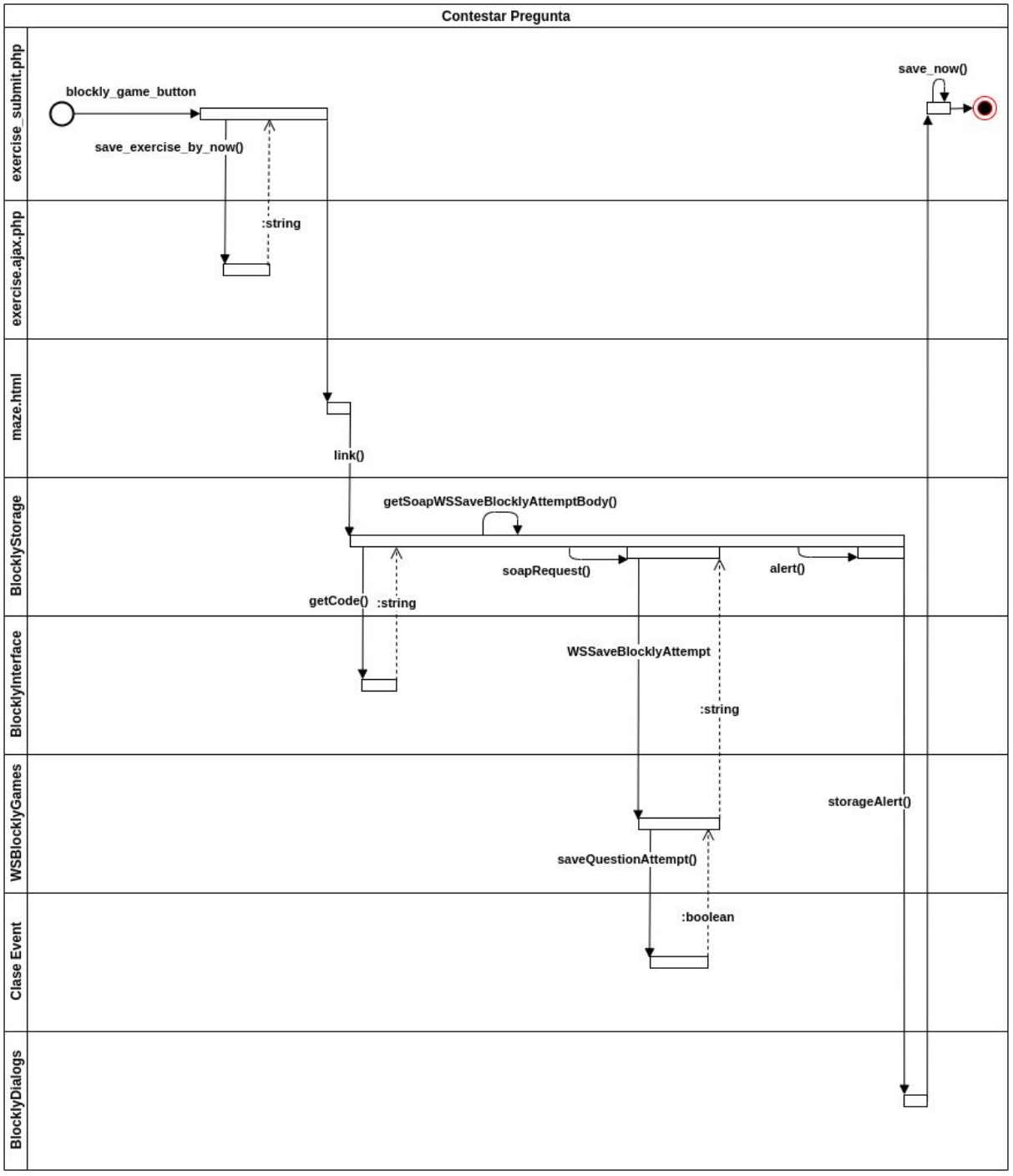


*Figura 30. Diagrama de Flujo - Contestar Pregunta.*

### 4.3.3 Corregir Pregunta de Blockly Games

Desde el menú de “Puntuación de los alumnos” de un ejercicio es posible acceder a la opción de corregir un intento de respuesta del mismo. Al hacer clic en el botón “Calificar” se ejecuta el script **exercise_show.php** que muestra los resultados del intento seleccionado comenzando el proceso que se detalla en la Figura 31.

Para obtener toda la información de configuración del ejercicio que permite definir cómo se mostrarán los resultados se utiliza el método **get_exercise_track_exercise_info()** de la clase **ExerciseLib**. Luego el script recupera todos los identificadores (ids) de las

preguntas del ejercicio, junto con las respuestas relacionadas a este intento, de la base de datos. Para cada uno de los ids de pregunta recuperados se utiliza el método **read()** para obtener el puntaje máximo y su tipo de pregunta.

Luego se utiliza la función **manage_answer()** implementada en la clase **Exercise** para generar el código html que representa la respuesta de esa pregunta. Este método genera en primer lugar el html que muestra la pregunta. Luego, según el tipo de pregunta, se obtiene el html que muestra la respuesta. En este ejemplo se utiliza el método **display_blockly_answer** propio de las preguntas de Juegos de Blockly Games.

Este método se encuentra implementado en la clase **ExerciseShowFunctions** y devuelve el html con el link que redirecciona al juego de Blockly Games con su respuesta. Para obtener la url al juego utiliza la función **getGameUrl()** de la clase **BlocklyQuestion**.

Luego de obtener el html de todas las preguntas y sus respuestas, el script **exercise_show.php** muestra en pantalla la página generada con los resultados.

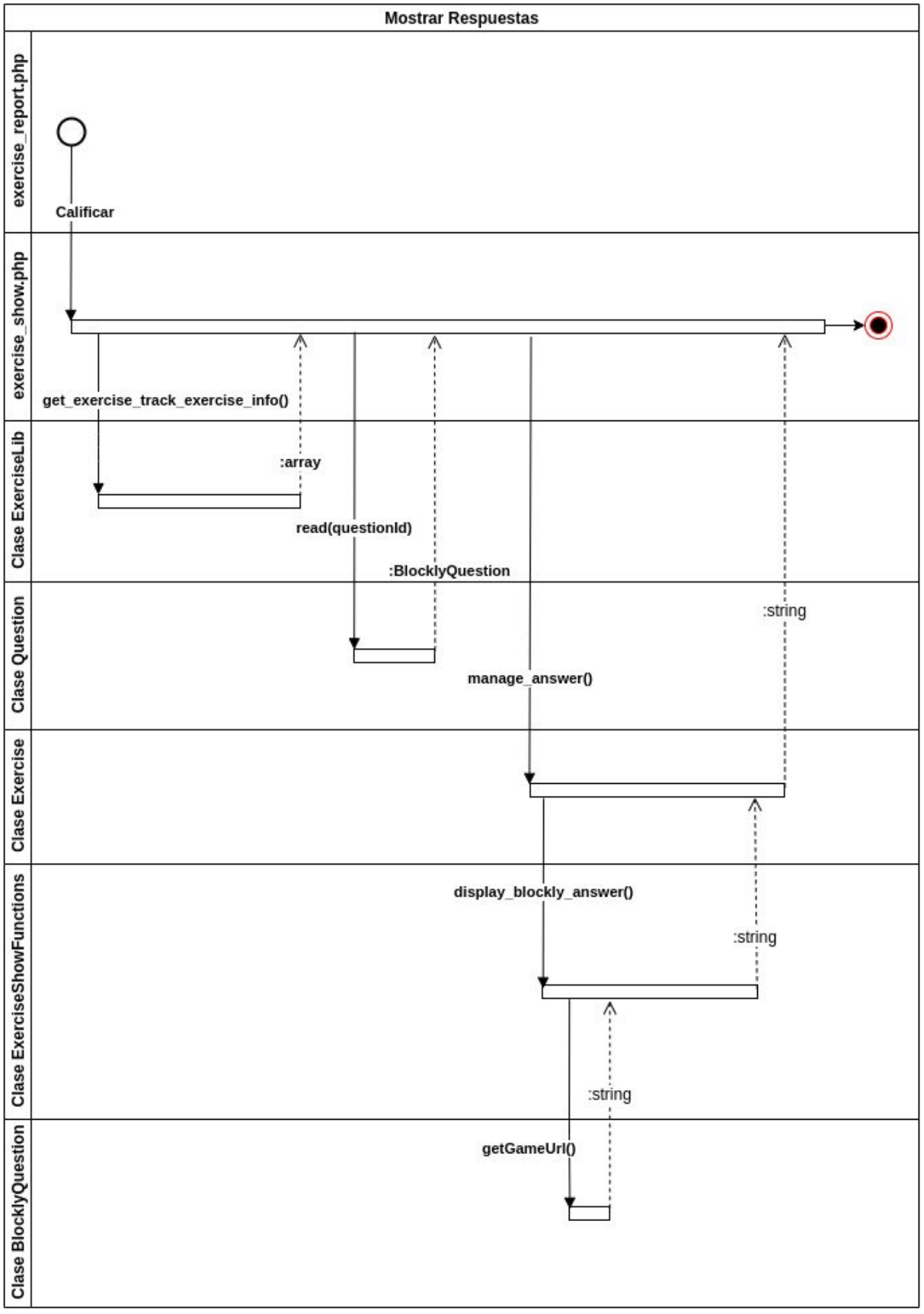


*Figura 31. Diagrama de Flujo - Mostrar Respuestas.*

Para acceder al juego de Blockly Games y ver la respuesta creada por el estudiante, desde la pantalla de resultados se debe presionar el link “Haga clic aquí para ir al juego” que se encuentra en el campo de respuesta de la pregunta de Juego de Blockly Games, iniciando así el proceso que puede verse en la Figura 32. Este link abrirá el juego de Blockly Games en otra pestaña o ventana, según la configuración del navegador donde se esté ejecutando Studium, y hará foco en la respuesta.

Cuando el usuario termine de ver y ejecutar el código generado como respuesta para ese juego, puede volver a la pantalla de resultados. Para hacer esto deberá hacer clic en el botón “Volver al ejercicio de Studium” que ejecutará la función **link()** del espacio de nombre **BlocklyStorage**. Esta función volverá el foco del navegador a la pantalla de resultados y cerrará la ventana o pestaña del juego de Blockly Games.

Cada pregunta de tipo Blockly Games también incluirá el botón de “Corregir y Puntuar” en la pantalla de resultados. Al presionar este botón se ejecuta la función **showfck()** que muestra los campos de entrada de comentarios y de puntaje para esa pregunta. Una vez completados estos campos para todas las preguntas, el usuario debe hacer clic en el botón “Corregir este ejercicio” para guardar la corrección. El botón ejecuta la función **getFCK()** que obtiene todos los datos de la corrección y los pasa como parámetro al script **exercise_report.php**. Este script se encarga de actualizar el intento de respuesta en la base de datos y muestra la pantalla de puntuacion de los alumnos finalizando el proceso.

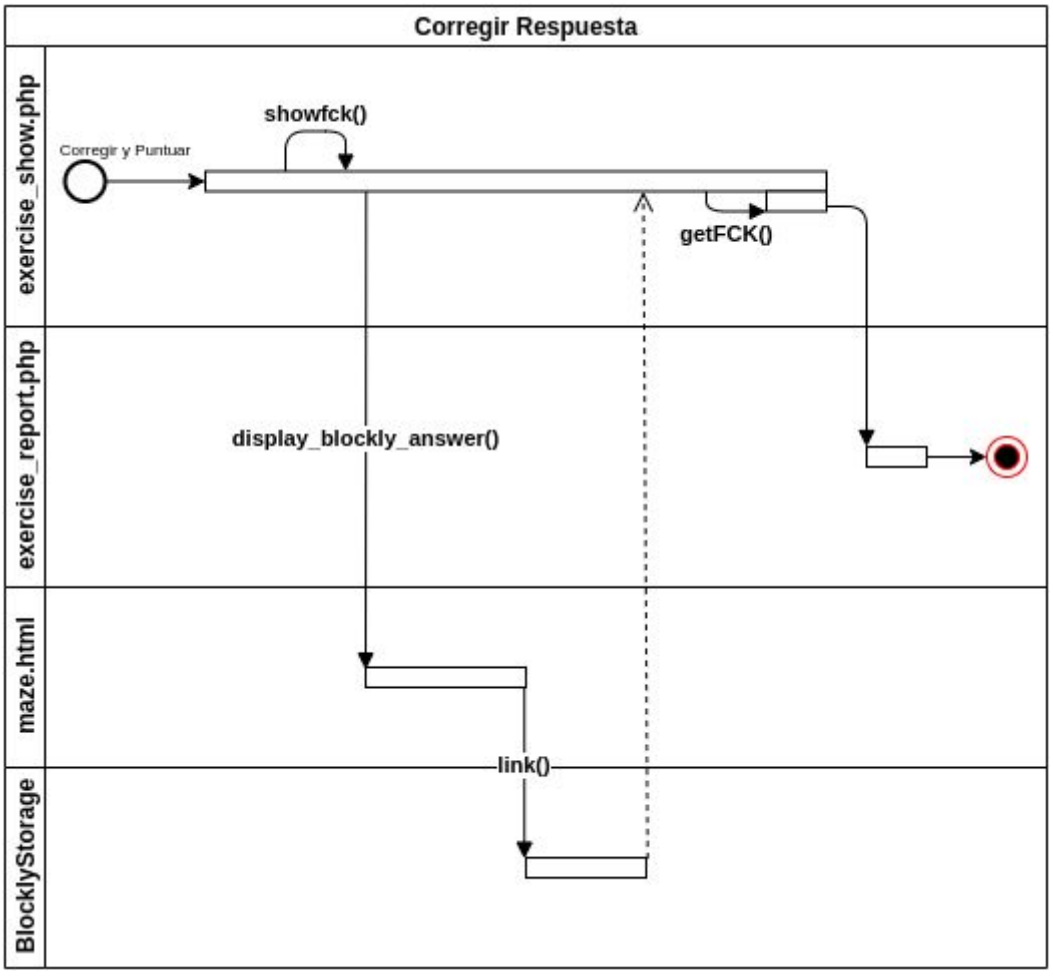


*Figura 32. Diagrama de Flujo - Corregir Respuesta.*

## 4.4 Implementación

En esta sección se hace una revisión de la arquitectura de la implementación de Studium. La información presentada brinda una breve descripción de la lógica desarrollada y luego ofrece un detalle de cada componente que forma parte de la implementación. Para cada uno se incluye una explicación de su utilización, su ubicación dentro de la estructura de archivos del sistema y un detalle de sus métodos más importantes.

Los componentes incluidos en esta sección son aquellos que contienen la lógica implementada para el desarrollo de Studium. No se analiza la lógica de las aplicaciones Chamilo o Blockly Games, que no haya sido modificada para el desarrollo de Studium.

La información brindada a continuación puede utilizarse como punto de inicio para analizar más profundamente el código de Studium disponible en:

- https://github.com/neptuneschild/Studium

### 4.4.1 Modificación de la Lógica de Chamilo

Para realizar la integración de las herramientas Chamilo y Blockly Games se decidió utilizar la funcionalidad de pregunta ofrecida por Chamilo. Con esto en mente se desarrolló un nuevo tipo de pregunta que se relaciona a los diferentes juegos de Blockly Games.

Este nuevo tipo de pregunta permite acceder a un juego de Blockly Games y almacena el código generado, como respuesta, en la base de datos de Chamilo. Cada pregunta estará relacionada a un juego y nivel específico en idioma español. Adicionalmente, el nuevo tipo de pregunta ofrece toda la funcionalidad propia de una pregunta de Chamilo.

Debido a su sencilla lógica se utilizó como modelo el tipo de pregunta OpenQuestion de Chamilo para crear el nuevo tipo de pregunta de Juego de Blockly Games.

A continuación se analizan todos los componentes, dentro de Chamilo, utilizados para la desarrollo de esta lógica.

#### Clase Blocky Question

La clase BlocklyQuestion es utilizada para instanciar el tipo de pregunta de Juego de Blockly Games. Al igual que todos los tipos de pregunta de Chamilo, BlocklyQuestion extiende a la clase **Question**. Su implementación se encuentra en el archivo **blocklyQuestion.class.php** que está ubicado en el directorio **~/chamilo/main/exercise** de Studium. Seguidamente se detallan los métodos más importantes de esta clase.

El método **return_header()** se utiliza para mostrar a los administradores del curso, el título de la pregunta y su estado de corrección al momento de realizar la corrección del ejercicio.

El método **createAnswersForm()** agrega al formulario, pasado por parámetro, los campos a utilizarse en la creación de una pregunta de este tipo. Entre los campos agregados al formulario se encuentran:

- **blockly_selected_game**: indica el juego seleccionado de Blockly Games que será evaluado en la pregunta.

- **blockly_selected_game_description**: contendrá la descripción del juego seleccionado en **blockly_selected_game**.
- **blockly_selected_level**: indica el nivel del juego seleccionado.
- **weighting**: indica el puntaje máximo con el que puede calificarse la pregunta.

El campo **blockly_selected_level** se muestra solo cuando se selecciona un juego con niveles disponibles en el campo **blockly_selected_game**. Para lograr este comportamiento se agrega un script en JQuery al formulario. Este script oculta o muestra la sección **blockly_level** según el juego seleccionado. Adicionalmente este script es el encargado de actualizar el campo **blockly_selected_game_description** según se modifique el juego seleccionado.

El método **getBlocklyDefaultGamesList()** es el encargado de retornar el arreglo que contiene los nombres de todos los juegos disponibles en Blockly Games.

El método **getBlocklyDefaultGamesDescription()** obtiene las descripciones de los juegos disponibles de Blockly Games según el lenguaje de la aplicación.

El método **processAnswersCreation()** procesa los datos introducidos en el formulario de creación de la pregunta. Los datos son almacenados en la base de datos de Chamilo utilizando el método **save()** implementado en la clase **Question**. Para guardar el tipo de juego y el nivel seleccionado de Blockly Games en la base de datos se utiliza el campo **extra** de la pregunta.

El método **getGameURL()** genera la url a un juego específico de Blockly Games utilizando el contenido del campo **extra** de una pregunta que se recibe por parámetro. Este campo es dividido para obtener los dos valores almacenados en el mismo. El primero de ellos es el tipo de juego de Blockly Games y el segundo el nivel del mismo. Para generar la url se utiliza el valor de la constante **blockly_url**, que contiene la ruta de la herramienta Blockly Games, con la porción de la url correspondiente al juego y nivel específico que se desea ejecutar.

### Clase Question

La clase Question define los campos genéricos utilizados por todos los tipos de pregunta de Chamilo. Además implementa parte de la funcionalidad necesaria para manipular las preguntas. Su implementación se encuentra en el archivo **question.class.php** localizado en el directorio **~/Chamilo/main/exercise** de Studium.

La constante **questionTypes** de la clase Question mapea todos los tipos de pregunta disponibles en Chamilo con los archivos donde se encuentra su implementación. La clase **BlocklyQuestion** fue agregada para que fuera reconocida por el sistema.

El método **__construct()** de esta clase también fue modificado para que el nuevo tipo de pregunta se agregue a la lista de preguntas con devolución. Esto permite a los administradores del curso agregar comentarios relacionados a las respuestas de estas preguntas durante la corrección de los ejercicios.

Asimismo, para mostrar el mensaje de Revisado/No Revisado y el puntaje de las preguntas tipo **BlocklyQuestion**, durante la corrección de los ejercicios, se modificó el método **return_header()**.

Finalmente se creó el método **selectExtra()** para acceder al atributo **extra** de una instancia de la clase Question.

### Clase Exercise

La clase **Exercise** permite instanciar ejercicios de Chamilo, su implementación se encuentra en el archivo **exercise.class.php** localizado en el directorio **~/Chamilo/main/exercise** de Studium.

El método **manage_answer()** de esta clase, se utiliza para gestionar las respuestas a una pregunta, y fue modificado para agregar la clase **BlocklyQuestion** a la lista de preguntas que solo tiene una posible respuesta. De esta manera, cuando se desea visualizar una pregunta de este tipo, se genera el código html para mostrar una sola respuesta asociada a la misma.

Dentro del mismo método se agregó el nuevo tipo de pregunta a la lógica que obtiene los valores de la respuesta y el puntaje de la pregunta para su visualización.

También se agregó al método **manage_answer()** una llamada a **display_blockly_answer()** de la clase **ExerciseShowFunctions**, que se ejecuta cuando se requiere mostrar una respuesta para el tipo de pregunta **BlocklyQuestion**.

Otra modificación relevante del método **manage_answer()** fue el guardado de las respuestas para las preguntas de tipo **BlocklyQuestion**.

A su vez, el método **sendNotificationForOpenQuestions()** fue modificado para incluir la respuesta relacionada a una pregunta de tipo **BlocklyQuestion** en la notificación que se envía al responder una pregunta de este tipo.

### Clase ExerciseShowFunctions

La clase **ExerciseShowFunctions** contiene la lógica utilizada por Chamilo para mostrar, en formato html, las respuestas a los diferentes tipos de preguntas. Está implementada en el archivo **exercise_show_functions.lib.php** ubicado en el directorio **~/chamilo/main/inc/lib** de Studium.

Dentro de la clase **ExerciseShowFunctions** está implementado el método **display_blockly_answer()** que procesa la respuesta a una pregunta de tipo **BlocklyQuestion** y la devuelve en formato html.

### Clase ExerciseLib

La clase **ExerciseLib** contiene parte de la lógica que permite mostrar una pregunta y sus respuestas. Está implementada en el archivo **exercise.lib.php** localizado en el directorio **~/chamilo/main/inc/lib** de Studium.

El método **showQuestion()** de la clase **ExerciseLib** fue modificado para agregar la lógica que permite mostrar el formulario de respuesta para una pregunta de tipo **BlocklyQuestion**. El formulario generado contiene un botón que permite navegar al juego de Blockly Games relacionado a la pregunta y el campo oculto **choiche[]** que contendrá la respuesta obtenida en la ejecución del juego.

El método **displayQuestionListByAttempt()** también fue modificado para considerar el nuevo tipo de pregunta al mostrar los resultados de un ejercicio. Este método valida si la respuesta a la pregunta fue corregida, y actualiza la cuenta de preguntas pendientes de corrección en caso de ser necesario.

### Librería Api

La librería **Api** contiene la lógica de uso genérico que se incluye por defecto en todos los componentes funcionales de Chamilo. Se encuentra dentro del archivo **api.lib.php** ubicado en **~/chamilo/inc/lib** de Studium.

La lógica de Chamilo usualmente usa una constante numérica para identificar los tipos de preguntas durante la ejecución del sistema. Debido a esto la constante **$BLOCKLY_QUESTION** fue creada dentro de esta librería con valor 100. Se eligió el valor 100 para evitar conflictos con futuras versiones de Chamilo que podrían agregar nuevos tipos de preguntas.

Adicionalmente, Chamilo valida los tipos de preguntas que pueden ser creados al generar una nueva pregunta en un ejercicio. Para garantizar que el nuevo tipo de pregunta no sea rechazado por esta validación se incluyó el nuevo tipo de pregunta dentro de la constante **QUESTION_TYPES**.

### Script exercise_show.php

El script **exercise_show.php**, localizado en el directorio **~/chamilo/main/exercise**, es el punto de entrada para mostrar a estudiantes y profesores, los resultados de un ejercicio.

Este script se modificó para incluir en la visualización de resultados de los ejercicios, los resultados del nuevo tipo de pregunta. De esta forma, para cada pregunta tipo **BlocklyQuestion** de un ejercicio, se ejecuta el método **manage_answer()** de la clase **Exercise** y se agregan los campos de entrada para comentarios y puntaje al formulario de corrección de las mismas.

### Script exercise_submit.php

El script **exercise_submit.php**, localizado en el directorio **~/chamilo/main/exercise** de Studium, contiene parte de la lógica necesaria para ejecutar un ejercicio y mostrar sus preguntas.

Se modificó el código JQuery de este script para agregar un evento que se dispara al hacer clic en el botón **blockly_game_button** de una pregunta tipo **BlocklyQuestion**. Se valida si es la primera vez que el usuario intenta responder esta pregunta en este intento de respuesta y, si es así, crea el intento asociado a la misma en la base de datos. Para esto genera un requerimiento de Ajax para el método **save_exercise_by_now** del script **exercise.ajax.php**.

Adicionalmente se agregan a la url, que se usará para redirigir al juego de Blockly Games, los datos que son usados por Blockly Games para identificar el intento de respuesta. Estos datos serán usados luego por Blockly Games para guardar el estado del juego y redirigir de vuelta al intento en Studium. Una vez modificada la url, el evento termina su ejecución utilizando la url para redirigir a la pantalla del juego de Blockly Games.

Servicio Web WSBlocklyGames

El servicio web SOAP, **WSBlocklyGames**, fue creado para manejar la comunicación entre la aplicación Blockly Games y Chamilo, y está ubicado en el archivo **blockly-games.soap.php** del directorio **~/chamilo/main/webservices** de Studium.

Este servicio define el método **WSSaveBlocklyAttempt** que actualiza el intento de respuesta de una pregunta en la base de datos con los datos pasados en el requerimiento SOAP. Para esto, obtiene todos los datos del requerimiento y los utiliza para invocar el método **saveQuestionAttempt()** de la clase **Event.**

Constantes de Lenguaje

Los archivos **trad4all.inc.php** contienen las constantes que se utilizarán para almacenar los textos relacionados el nuevo tipo de pregunta en cada idioma. Estos textos son accedidos a través del método **getLang(),** implementado por Chamilo.

El nuevo tipo de pregunta fue configurado para los idiomas Español e Inglés. Los respectivos archivos para estos lenguajes se encuentran en los directorios **~/chamilo/lang/spanish** y **~/chamilo/lang/english** de Studium.

### 4.4.2 Modificación de Lógica - Blockly Games

Para la integración de Blockly Games con Chamilo, se ubicó el directorio raíz de Blockly Games dentro del directorio de instalación de Chamilo en Studium, **~/chamilo/main/blockly-games**. La herramienta fue compilada dentro de este directorio para que pueda ser accedida fácilmente desde Chamilo.

Adicionalmente se consideró que Blockly Games debería poder reconocer que la ejecución de sus juegos están asociados a una pregunta de Chamilo. A su vez, al finalizar la ejecución de un juego, éste debía poder guardar su estado como respuesta a estas preguntas y permitir al usuario continuar con la ejecución normal del ejercicio de Chamilo.

Para lograr este comportamiento se modificó la lógica de los diferentes componentes de Blockly Games como se detalla a continuación.

Espacio de Nombres BlocklyStorage

El espacio de nombres **BlocklyStorage** define la lógica que permite recuperar y guardar el estado de los juegos de Blockly Games. Éste se implementa en el archivo **storage.js** localizado en el directorio **~/chamilo/main/blockly-games/appengine/common** de Studium.

Esta lógica fue modificada para poder guardar el estado de un juego en la base de datos de Chamilo. Adicionalmente también se modificó para recuperar el estado de un juego de la url del mismo.

La función **link()** fue modificada para que se ejecute al presionar el botón “Volver al ejercicio de Studium” de la interfaz de juego de Blockly Games. Ésta válida si el juego está siendo accedido por un estudiante, en cuyo caso envía un requerimiento SOAP que almacena el estado del juego actual en la base de datos de Chamilo y luego redirige la pantalla a la ejecución del ejercicio de Chamilo desde donde se accedió al juego. Caso

contrario se interpreta que un profesor está accediendo al juego para su corrección, por lo que se borra el localStorage, se cierra la ventana actual y se redirige a la página de Studium sin hacer ninguna otra acción previa.

La función **retrieveXml()** es la encargada de restaurar el estado del juego de Blockly Games y fue modificada para recibir como parámetro el código con el estado de un juego en formato XML.

La función **soapRequest()** se creó para generar y enviar el requerimiento SOAP para el servicio web **WSBlocklyGames** de Chamilo. Esta recibe como parámetro el cuerpo del requerimiento SOAP y la función a utilizarse para procesar la respuesta del mismo.

La función **getSoapWSSaveBlocklyAttemptBody()** fue creada para obtener el cuerpo del requerimiento SOAP en formato xml con el nombre del método a invocar y los parámetros necesarios para su ejecución en el servicio web.

Por otro lado, **handleSoapWSSaveBlocklyAttemptResponse()** se creó para validar el estado de la respuesta del requerimiento SOAP y proceder acorde al resultado. En caso de obtener una respuesta exitosa se utiliza el método **BlocklyStorage.alert()** para emitir el mensaje de “El juego fue guardado con éxito”. Caso contrario se informa al usuario de que hubo un error.

### Espacio de Nombres BlocklyInterface

El espacio de nombres **BlocklyInterface** contiene funciones de uso común para todos los juegos de Blockly Games. Éste se define en el archivo **lib-interface.js** localizado en el directorio **~/chamilo/main/blockly-games/appengine/js** de Studium.

Debido a que Blockly Games no permite modificar el editor de bloques luego de ingresar código en el editor de texto de la ejecución de un juego, fue necesario modificar la lógica de este espacio de nombres.

El método **getCode()** fue modificado para que se obtenga el código del editor de bloques en el caso de que ambos editores están disponibles durante la ejecución de un juego.

El método **setCode()** se modificó para asegurarse que ante la presencia de ambos editores durante la ejecución de un juego se actualice el código del editor de bloques.

### Espacio de Nombres BlocklyDialogs

**BlocklyDialogs** es un espacio de nombres que contiene los métodos para mostrar los mensajes y alertas comunes de los juegos de Blockly Games. Éste está definido en el archivo **lib-dialogs.js** localizado en la ruta **~/chamilo/main/blockly-games/appengine/js** de Studium.

El método **goBackToStudium()** fue creado para limpiar el contenido del localStorage y redirigir la pantalla al ejercicio de Studium.

El método **storageAlert()** modifica la alerta emitida por el método **alert()** del espacio de nombres **BlocklyStorage** dándole formato y definiendo el método a ejecutarse al cerrar la ventana. Éste fue modificado para que ejecute **goBackToStudium()** al cerrar la ventana de alerta.

El método **congratulations()** muestra la alerta al ejecutar el código de un juego con éxito, finalizando el mismo. Éste se modificó para que el usuario no sea redirigido al próximo nivel del juego al cerrar la alerta.

Espacio de Nombres Pond

**Pond** es un espacio de nombres que contiene métodos utilizados por los juegos **Tutor del estanque** y **Estanque**. Éste se encuentra definido en el archivo **pond.js** localizado en el directorio **~/chamilo/main/blockly-games/appengine/pond/js** de Studium.

El método **docsButtonClick()** fue modificado para asegurar que la documentación para estos juegos siempre sea cargada en idioma inglés independientemente del idioma de la plataforma. Esto fue necesario dado que esta documentación sólo está disponible en dicho idioma.

Espacio de Nombres Pond.Duck

El espacio de nombres **Pond.Duck** contiene todos los métodos propios del juego **Estanque**. Éste se define en el archivo **duck.js** ubicado en el directorio **~/chamilo/main/blockly-games/appengine/pond/duck/js** de Studium.

El método **init()** se modificó para que se recupere el estado del juego en caso de que se envíe información como parámetro en el hash de la url.

Espacio de Nombres Puzzle

El espacio de nombres **Puzzle** contiene parte de la lógica relacionada al juego **Rompecabezas**. Éste se encuentra definido en el archivo **puzzle.js** ubicado en el directorio **~/chamilo/main/blockly-games/appengine/puzzle/js** de Studium.

El método **init()** de este espacio de nombres fue modificado para que se actualice el estado del juego si la url del mismo contiene el código del mismo.

El método **checkAnswers()** válida si los bloques del rompecabezas fueron acomodados correctamente y muestra una alerta indicando el resultado. Éste fue modificado para que el botón de “Aceptar” de la ventana de alerta no redirige la pantalla de index de Blockly Games cuando el juego fue completado exitosamente.

Archivos HTML

No todos los juegos de Blockly Games cuentan con la opción de guardar el estado de los juegos por defecto. Para modificar esto tuvo que incluirse el archivo **storage.js** en los templates html de estos juegos. Adicionalmente los juegos que ya importaban este archivo tuvieron que ser modificados dado que éste no podía ser encontrado luego de que Blockly Games fue incluido en Chamilo.

Se modificaron los archivos **puzzle.html**, **maze.html**, **bird.html**, **turtle.html**, **movie.html**, **music.html**, **pond-tutor.html** y **pond-duck.html**, ubicados en el directorio **~/chamilo/main/blockly-games/appengine** de Studium, agregando la siguiente línea de código dentro del cuerpo html:

***<script src="common/storage.js"></script>***

Esto permite que los métodos contenidos en el espacio de nombres **BlocklyStorage** puedan ser utilizados en la ejecución de estos juegos.

### 4.4.3 Visualización y Estilo Chamilo

Studium fue personalizado reemplazando los iconos propios del sistema Chamilo. Adicionalmente fue necesario crear iconos para el tipo de pregunta **BlocklyQuestion**.

En los demás aspectos del sistema, Studium utiliza el estilo de la página provisto por defecto en Chamilo.

A continuación se brinda más detalle referente a los iconos creados durante el desarrollo.

#### Iconos Studium

Para reemplazan aquellos provistos por defecto en Chamilo fue necesario crear varios iconos e imágenes.

El archivo **header-logo.png** representa el logo de Studium, que puede verse en a Figura 33. Éste tiene un tamaño de 245 x 68 píxeles. Se encuentra localizado en los directorios **~/chamilo/app/Resources/public/css/themes/chamilo/images** y **~/chamilo/web/css/themes/chamilo/images** de Studium.


*Figura 33. Captura de Imagen - Icono header-logo.*

El icono **platform,** que se ve en la Figura 34, se utiliza en varios contextos dentro de Studium. El mismo se encuentra en diferentes archivos y directorios según su funcionalidad y tamaño:

- Archivo **favicon.ico**, se utiliza para mostrar el icono en el título de la página. Es de 24x24 píxeles y se localiza en los siguientes directorios de Studium:
    - **~/chamilo/app/Resources/public/css/themes/chamilo/images**
    - **~/chamilo/app/Resources/public/css/themes/medical/images**
    - **~/chamilo/web/css/themes/chamilo/images**
    - **~/chamilo/web/css/themes/medical/images**
    - **~/chamilo**
- Archivo **platform.png**, se muestra en la cabecera de los menús de administración de la página. Cuenta con tamaño de 32x32 y 48x48 píxeles y se localiza en **~/chamilo/main/img/icons/** dentro de los directorios correspondiente a su tamaño en pixeles.
- Archivo **apple-touch-icon.png**, se usa como icono de la opción apple-touch de Chamilo. Es de 57x57 pixeles y se localiza en **~/chamilo/**.

*Figura 34. Captura de Imagen - Icono platform.*

El archivo **session_default.png** contiene la imagen que se utiliza por defecto como presentación de los cursos creados dentro de Studium, como puede verse en la Figura 35. Ésta es de 400 x 224 píxeles y se localiza en el directorio **~/chamilo/main/img** de Studium. Adicionalmente el archivo **session_default_small.png** contiene una versión más pequeña de esta imagen, 85 x 48 píxeles, que se localiza en el mismo directorio.




*Figura 35. Captura de Imagen - Icono session_default.*

### Iconos BlocklyQuestion

Para la representación de la clase BlocklyQuestion dentro de Studium fue necesario crear dos iconos. Estos iconos son utilizados en los diferentes menús de creación de preguntas así como también en la visualización de los ejercicios. Se crearon representaciones de ambos iconos en 16, 22, 32, 48, 64 y 128 pixeles.

El archivo **blockly_question.png** contiene el icono que representa el tipo de pregunta BlocklyQuestion cómo puede verse en la Figura 36. Éste se encuentra ubicado en el directorio **~/chamilo/main/img/icons** dentro de las carpetas que representan sus diferentes versiones según su tamaño en pixeles en Studium.

*Figura 36. Captura de Imagen - Icono BlocklyQuestion.*

El archivo **blockly_question_na.png**, que puede verse en la Figura 37, representa el nuevo tipo de pregunta cuando este esta bloqueado y no está disponible para el usuario. Las representaciones de éste se almacena en los mismos directorio que el icono **blockly_question.png** según su tamaño en pixeles.

*Figura 37. Captura de Imagen - Icono BlocklyQuestion NA.*

### 4.4.4 Visualización y Estilo Blockly Games

La versión de Blockly Games utilizada en el desarrollo de Studium conserva el estilo e iconos originales de la herramienta.

Sin embargo, debido a que ciertas funcionalidades de la herramienta se delegan al tipo de pregunta **BlocklyQuestion** de Chamilo, las plantillas de visualización de Blockly Games tuvieron que ser modificados para restringir las funcionalidades ofrecidas a los usuarios.

Entre las funcionalidades restringidas se incluyen:

- Cambio de niveles del juego ejecutado.
- Cambio de lenguaje de la herramienta.
- Navegación hacia la pantalla de index de la herramienta.

Adicionalmente las plantillas de algunos juegos tuvieron que ser modificadas para agregar la opción de guardar el estado del juego y volver a Studium.

A continuación se detallan los cambios realizados a los espacio de nombres de Blockly Games que definen la interfaz de la herramienta.

#### Espacio de Nombres BlocklyGames.soy

El espacio de nombre **BlocklyGames.soy** contiene la definición de las plantillas Composite que son utilizadas comúnmente por todos los juegos de Blockly Games. Éste se encuentra en el archivo **template.soy** localizado en el directorio **~/chamilo/main/blockly-games/appengine** de Studium.

El template **headerBar**, que se utiliza para mostrar la cabecera de cada juego, fue configurado para ocultar la siguiente información:

- Menú de niveles del juego.
- Select con las opciones para cambiar lenguaje de Blockly Games.

El botón **linkButton** dentro de este template es el utilizado por el usuario para guardar el estado del juego de Blockly Games y volver a Studium. Éste fue modificado para mostrar el mensaje definido por la constante **Games.linkToolButton**.

El template **titleSpan** fue modificado para prevenir la navegación del usuario a la página de index al cliquear el título de la herramienta durante la ejecución de un juego.

El template **doneDialog** define la alerta que se muestra cada vez que un juego se ejecuta con éxito. Éste fue modificado para quitar el mensaje que sugiere pasar al próximo nivel del juego.

Espacio de Nombres Pond.Duck.soy

**Pond.Duck.soy** es un espacio de nombres que define algunas de los plantillas que son utilizados por Blockly Games para visualizar el juego **Estanque**. Éste se define en el archivo **template.soy** localizado en el directorio **~/chamilo/main/blockly-games/appengine/pond/duck** de Studium.

El template **.start** dentro de este espacio de nombres fue modificado para incluir el botón “Volver al ejercicio de Studium”.

Adicionalmente se agregó a este espacio de nombres las llamadas a los templates **BlocklyGames.soy.doneDialog** y **BlocklyGames.soy.storageDialog.** Esto permite incluir los diálogos que se muestran al finalizar el juego correctamente y al guardar el juego en Studium durante la ejecución del juego **Estanque**.

Espacio de Nombres Puzzle.soy

El espacio de nombres **Puzzle.soy** contiene las plantillas que determinan la visualización del juego **Rompecabezas.** Éste se define en el archivo **template.soy** ubicado en el directorio **~/chamilo/main/blockly-games/appengine/puzzle** de Studium.

El template .**start** fue modificado para incluir el botón de guardado de estado “Volver al ejercicio de Studium” a este juego permitiendo así ejecutar la función **link()** del espacio de nombres **BlocklyStorage**.

Espacio de Nombres Games

El espacio de nombres **Games** define una serie de constantes con el contenido de los mensajes a ser mostrados en las diferentes secciones de Blockly Games. Estas constantes definen los mensajes en cada idioma disponible en la aplicación.

El archivo **es.json** contiene estas constantes para el idioma español y se localiza en **~/chamilo/main/blockly-games/json** de Studium.

La constante **Games.linkToolButton** fue añadida a este archivo con el mensaje “Volver al ejercicio de Studium” que se utiliza en la interfaz de la ejecución de todos los juegos.

## 4.5 Interfaz de Usuario

Esta sección muestra las interfaces principales de Studium donde se puede apreciar la integración de Chamilo con Blockly Games. Se eligieron dos de ellas de manera tal de visualizar el trabajo realizado. Las mismas se refieren a las interfaces para agregar y ejecutar un ejercicio de Blockly Games dentro de Studium

Al agregar un ejercicio en una lección como puede verse en la Figura 38, Studium muestra la opción de Juego de Blockly Games entre los tipos de pregunta disponibles en un

ejercicio. Una vez agregada una pregunta de este tipo, siguiendo el procedimiento normal de Chamilo, la misma puede verse en la lista de preguntas del ejercicio identificada por el icono del tipo de pregunta.

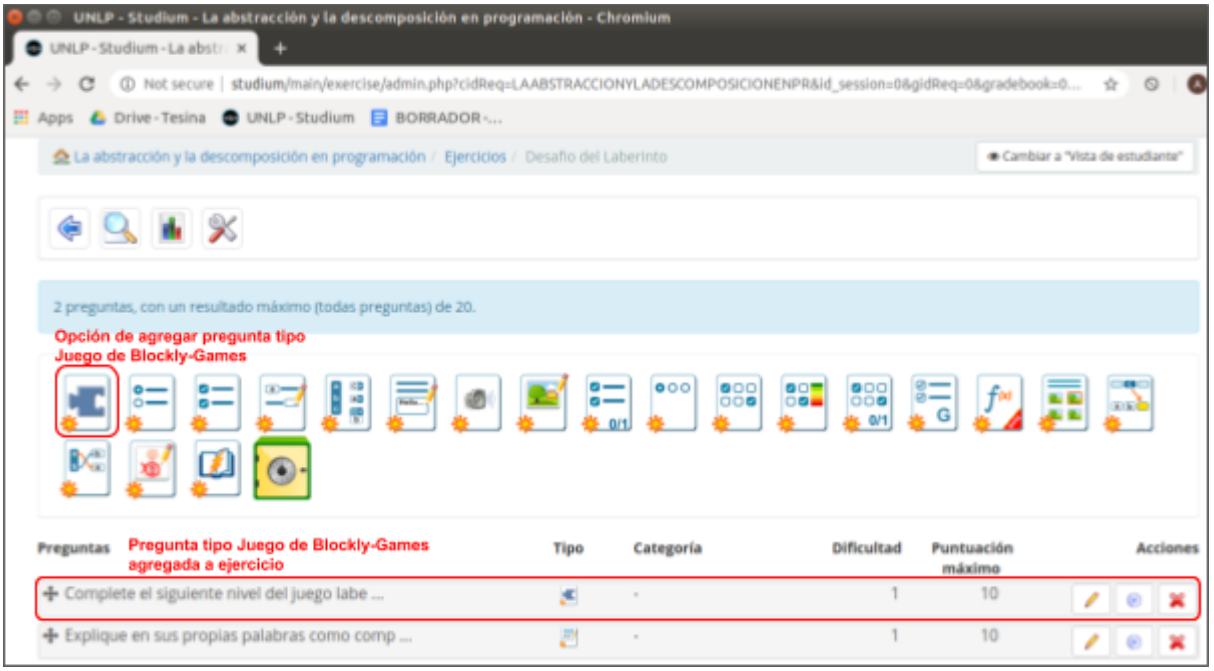


*Figura 38. Captura Navegador - Menú Administración de Ejercicio Studium.*

La Figura 39 muestra la interfaz de navegación de una lección que contiene un ejercicio relacionado a Blockly Games. Al ejecutar una pregunta de tipo Juego de Blockly Games dentro de la misma se puede ver el juego relacionado embebido en la pantalla de la lección permitiendo acceder al mismo tiempo al menú de navegación la lección y a la interfaz del juego.

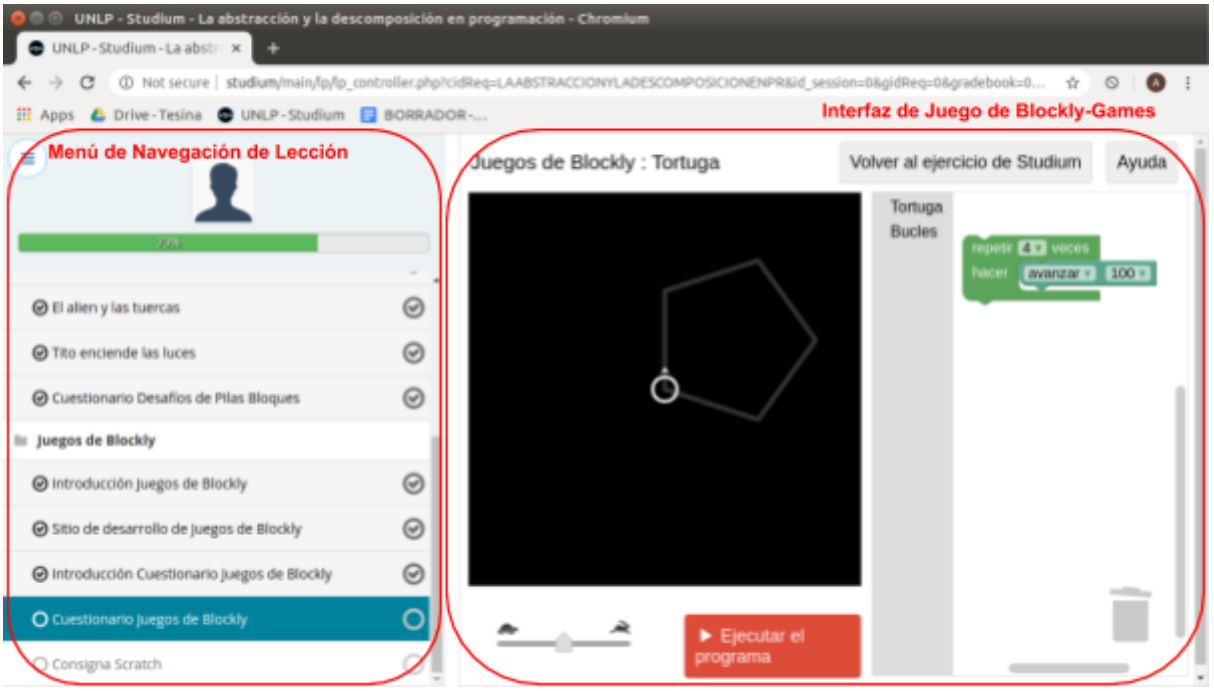


*Figura 39. Captura Navegador - Ejecución Juego de Blockly Games en Lección.*

# Capítulo 5. Evaluación de Studium

En este capítulo se realiza una breve introducción al concepto de usabilidad y se detalla el proceso de evaluación de la usabilidad de Studium.

## 5.1 Usabilidad

Si bien la usabilidad tiene varias definiciones, que conflictúan entre sí, para el propósito de este trabajo se considerará una definición general que busca englobar y conectar todas las otras definiciones. Ésta fue emitida en 1998 por la Organización de Estándares Internacional y define a la usabilidad como: “La medida en que un producto puede ser usado por usuarios específicos para alcanzar objetivos específicos con efectividad, eficiencia y satisfacción en un contexto de uso específico”. (Borsci, Federici, Malizia, De Filippis, 2019)

La usabilidad de cualquier herramienta o sistema tiene que ser considerada en términos de en qué contexto es usada, y cómo se adapta a ese contexto. Cuando se habla de sistema informático, teniendo en cuenta este concepto, es imposible determinar la usabilidad de un sistema sin definir primero a qué usuarios está dirigido, las tareas que éstos llevarán a cabo en el sistema y las características del ambiente físico, organizacional o social en donde el sistema será usado.

Si bien hay muchos métodos para estudiar la usabilidad de un sistema, para este trabajo se propone utilizar las pruebas de usuarios (Brooke, 2011)(Nielsen, 2012). Este método es uno de los más básicos y efectivos, y está dividido en 3 componentes:

- Obtener algunos usuarios representativos del sistema.
- Pedirle a estos usuarios que hagan alguna tareas representativas en el sistema.
- Observar qué hacen los usuarios, donde tienen éxito y donde tienen dificultades con la interfaz y obtener la opinión de ellos mismos sobre estos puntos.

## 5.2 Método Utilizado

Para realizar la evaluación de Studium se decidió obtener la opinión de los usuarios que formaron parte de la misma utilizando dos cuestionarios basados en la escala de Likert.

La escala de Likert es una herramienta de medición que permite medir actitudes y conocer el grado de conformidad del encuestado respecto a diferentes declaraciones. Este cuestionario toma la forma de una lista de declaraciones (ítems) relacionadas al objeto del estudio. Los encuestados deberán mostrar su nivel de acuerdo con cada declaración en una escala métrica de 5 o 7 puntos, de “completamente en desacuerdo” hasta “completamente de acuerdo”. La combinación de todas las respuestas revelan la actitud de los participantes hacia el objeto de estudio.(Joshi, Kale, Chandel, Pal, 2015)

Entre los métodos estandarizados para medir la usabilidad de un sistema que utilizan esta escala podemos mencionar Software Usability Measurement Inventory[21] (SUMI) y System Usability Scale(SUS)[22], entre otros. Si bien estos métodos son efectivos y

[21] Artículo del Método: http://sumi.uxp.ie/about/whatis.html, último acceso 16/03/2020.
[22] Artículo del Método: https://www.usabilitest.com/system-usability-scale, último acceso 16/03/2020.

ampliamente utilizados, no fueron seleccionados para el análisis de Studium debido a que requieren un muestreo de usuarios mayor al que dispone al momento de cierre de este informe para esta evaluación, aunque la evaluación propuesta se basa en dichos métodos.

La evaluación del sistema fue dividida según los perfiles de los usuarios a realizarla. Se incluyó en la misma solo la funcionalidad desarrollada por Studium y no aquella funcionalidad provista por defecto por las implementaciones de Chamilo y Blockly Games de manera tal de poder enfocar solamente la integración de ambas herramientas. Como guía para los usuarios se crearon 4 videos instructivos, subidos al canal de Studium en YouTube:

- https://www.youtube.com/channel/UCXuXIyhvWTkTfUVSxkUybmA?view_as=subscriber

El objetivo de estos videos es introducir a los usuarios a las funcionalidades que serán evaluadas, especialmente para aquellos usuarios que no tengan suficientes conocimiento en la navegación de Chamilo. Los videos subidos abordan las siguientes tareas:

- **Crear un ejercicio con pregunta de Juego de Blockly Games**: este video, orientado a profesores, explica cómo crear un ejercicio, en un curso existente, y agregar a éste una pregunta de tipo Juego de Blockly Games.
- **Agregar un ejercicio a Lección**: este video, también orientado a docentes, muestra cómo agregar un ejercicio existente a una lección de Studium.
- **Corregir un ejercicio con pregunta de Juego de Blockly Games**: este video muestra el proceso que se debe seguir para realizar la corrección de una respuesta a una pregunta de tipo Juego de Blockly Games.
- **Iniciar resolución de Lección de Studium**: este video, orientado a estudiantes, muestra cómo ejecutar una lección de Studium en un curso al que ya se encuentran inscriptos.

Los docentes invitados a formar parte de la evaluación tendrán acceso a los videos antes de utilizar Studium.

Se creó una curso en Studium basado en una secuencia didáctica trabajada en el módulo “Programación en Bloques” dictado en el marco de la propuesta “Especialización Docente en Didáctica de las Ciencias de la Computación”[23]. En dicho curso, se definió una lección y un ejercicio con al menos una pregunta de tipo Juego de Blockly Games. Se crearon 8 usuarios con rol de docentes y estudiantes matriculados en este curso, denominado “La abstracción y la descomposición en programación”. A los docentes, se les asignará la siguiente consigna: crear un ejercicio que contenga al menos una pregunta de tipo Juegos de Blockly Games y corregir un ejercicio similar previamente creado. Inmediatamente terminadas las tareas propuestas, los profesores deben contestar el cuestionario mostrado en la Tabla 5 que contiene las opciones de puntuación según Likert para cada una, siendo 1: “Totalmente en desacuerdo” y 5: “Totalmente de acuerdo”.

[23] Documento de la propuesta: http://sedici.unlp.edu.ar/handle/10915/73627, último acceso 18/03/2020.

| | Cuestionario Perfil Profesor | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 1 | Encontré el icono de "Juego de Blockly Games" fácilmente después de crear el nuevo ejercicio. | | | | | |
| 2 | La pantalla de preparar pregunta de ejercicio de Blockly Games me pareció intuitiva y fácil de usar. | | | | | |
| 3 | Pude seleccionar el nivel del ejercicio de Blockly Games sin inconvenientes. | | | | | |
| 4 | La interfaz de resultado del ejercicio a calificar fue muy fácil de entender. | | | | | |
| 5 | Me sentí muy seguro/a abriendo el juego de Blockly Games. | | | | | |
| 6 | Entendí fácilmente como volver a la pantalla de resultado desde el juego de Blockly Games. | | | | | |
| 7 | Pude ingresar los comentarios y el puntaje de la corrección sin ningún problema. | | | | | |
| 8 | Me pareció sencillo guardar las correcciones realizadas al nuevo ejercicio. | | | | | |

*Tabla 5. Cuestionario de Evaluación de Profesores.*

También se propuso una evaluación con estudiantes en donde se les propondrá visualizar el video “Iniciar resolución de Lección de Studium” y luego la siguiente consigna: buscar el curso “La abstracción y la descomposición en programación” y realizar la lección “Repeticiones y repeticiones anidadas” donde se incluye, entre otras cosas, ejercicios con preguntas de tipo Juego de Blockly Games. Esta lección puede verse en la Figura 40.

Al finalizar la consigna dada se les entregará el cuestionario con la escala de Likert que se muestra en la Tabla 6.

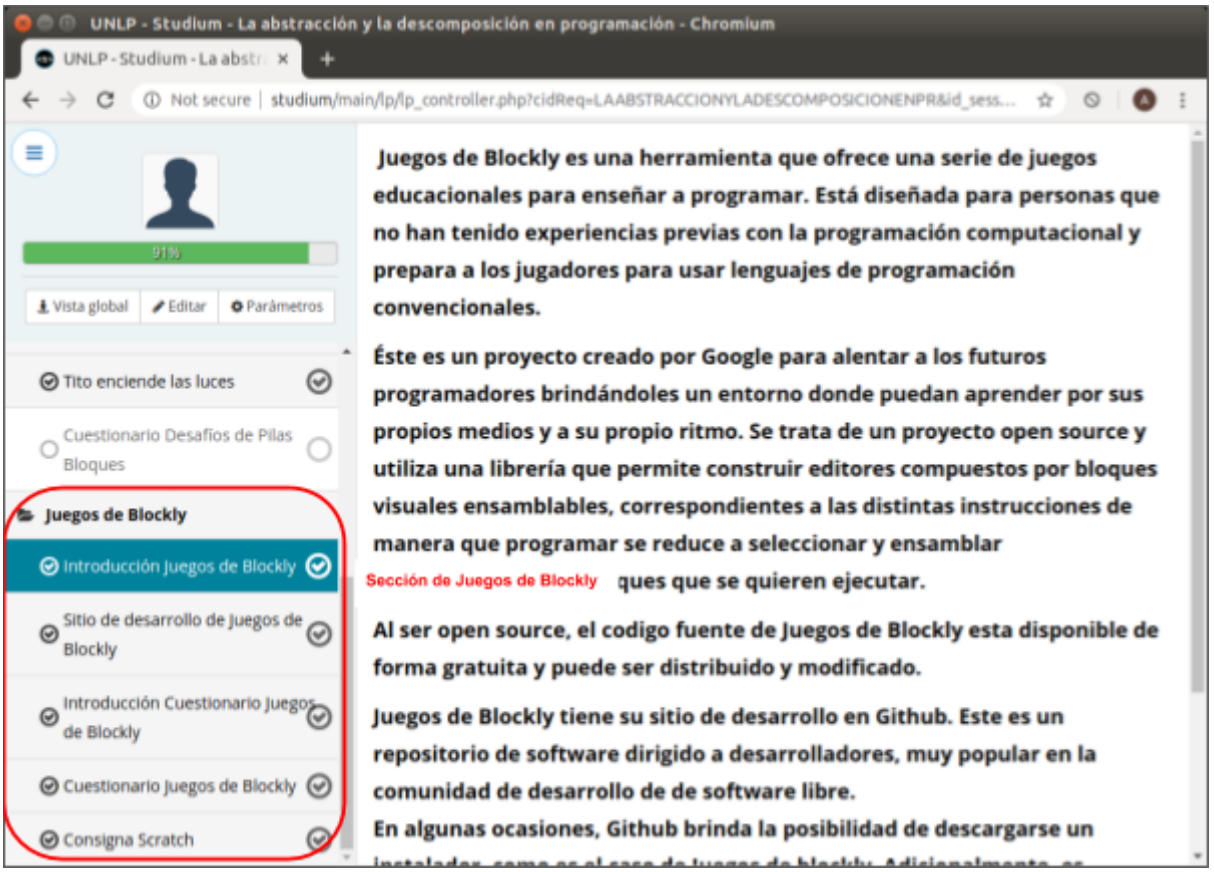


*Figura 40. Captura Navegador - Evaluación de Studium para Estudiantes.*

| | Cuestionario Perfil Estudiante | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 1 | Me sentí muy segura/o ejecutando los juegos Juegos de Blockly. | | | | | |
| 2 | Entendí fácilmente cómo volver al ejercicio de Studium desde los juegos de Blockly Games. | | | | | |
| 3 | Me resultó sencillo avanzar a otra pregunta en el ejercicio "Cuestionario Juegos de Blockly". | | | | | |
| 4 | Fue muy fácil terminar el ejercicio "Cuestionario Juegos de Blockly". | | | | | |
| 5 | Pude ver las respuestas ingresadas en el Cuestionario Juegos de Blockly sin inconvenientes. | | | | | |

*Tabla 6. Cuestionario de Evaluación de Estudiantes.*

Una vez obtenidos los resultados de ambos grupos de usuarios se realizará una evaluación de las opiniones de los mismos basadas en los cuestionarios de las Tablas 5 y 6.

## 5.3 Resultados Obtenidos

El proceso de evaluación estuvo previsto hacerse con docentes y estudiantes que participan del proyecto “Extensión en Vínculo con Escuelas Secundarias”. Las actividades de este proyecto comienzan habitualmente entre mediados de febrero y principios de

marzo. Esta evaluación estaba prevista para la segunda quincena del mes de marzo, luego de iniciar el contacto con las escuelas participantes del proyecto.

Debido a las recomendaciones emitidas por el Ministerio de Salud en el marco de la emergencia, respecto a la proliferación del Coronavirus COVID -19, y las resoluciones adoptadas por la UNLP al momento de cerrar este informe, fue necesario tomar ciertas consideraciones respecto a la evaluación de Studium propuesta.

La Resolución de la UNLP 667-2020[24] decreta, entre otras cosas, interrumpir las realización de instancias áulicas presenciales de enseñanza, mínimamente hasta el 31 de marzo, y eximir a los estudiantes de la carga presencial. Estas medidas imposibilitan avanzar con la evaluación de Studium antes de la fecha estipulada en la resolución, dado que no sería posible, ni responsable, invitar a los usuarios necesarios para realizar la evaluación presencial. Considerando esta información, se decidió no llevar a cabo la evaluación de Studium según estaba planeada.

Adicionalmente, se tuvo en cuenta en esta decisión la situación actual del autor de este trabajo quien, por motivos laborales y de desarrollo profesional no podrá demorar su entrega y defensa más allá de la fecha del 01/05/2020.

Se plantea la realización de la evaluación de Studium, detallada en este capítulo, como trabajo futuro de esta tesina.

[24] Sitio de la resolución: https://www.info.unlp.edu.ar/wp-content/uploads/2020/03/Res.667-2020.pdf, último acceso.

# Capítulo 6. Conclusiones y trabajos futuros

## 6.1 Conclusiones

En el marco de este trabajo se estudiaron y analizaron las herramientas de enseñanza de programación basada en bloques popularmente adoptadas para la enseñanza de programación en las escuelas de nuestro país. Se seleccionaron 2 herramientas de este tipo, de código libre, las cuales son usadas regularmente por proyectos de extensión de la Facultad de Informática en actividades relacionadas con el acercamiento de programación a las aulas de la escuela secundaria. Ambas herramientas fueron analizadas teniendo en cuenta la estructura de su código, el lenguaje de programación en el que están implementadas y la forma en que manejan sus plantillas, teniendo en cuenta que se busca integrarlas a un LMS/CMS. De este análisis resultó elegida la herramienta Blockly Games para utilizarla en la integración propuesta en este trabajo.

Se analizaron, en este trabajo, las características de los sistemas LMS/CMS, inicialmente se hizo un relevamiento de 14 sistemas existentes, de los cuales 3 de ellos fueron seleccionados como candidatos para ser usados como base en el desarrollo de Studium. Los 11 restantes fueron descartados debido a no ajustarse al criterio de selección planteado para el sistema a desarrollar en este trabajo. Los candidatos fueron instalados localmente y analizados teniendo en cuenta la funcionalidad ofrecidas y la posibilidad de integración con Blockly Games. Finalmente se seleccionó el sistema Chamilo para ser utilizado como base del desarrollo de Studium.

El aporte principal de este trabajo fue el desarrollo de Studium, una extensión del sistema LMS/CMS Chamilo, que integra una herramienta de enseñanza de programación basada en bloques.

A partir del análisis del código fuente de Chamilo se determinó cómo se integraría Blockly Games al mismo, definiéndose las tecnologías a utilizarse, los casos de uso y los diagrama de flujo relacionados a la lógica a ser implementada. Teniendo esta información en cuenta se procedió al desarrollo de Studium.

Para la evaluación de Studium se diseñaron test de usabilidad basados en la escala de Likert, con docentes y estudiantes del proyecto de extensión de la Facultad de Informática “Extensión en vínculo con escuelas secundarias”.

Para obtener las opiniones desarrolladas durante la evaluación por los usuarios, se elaboró un cuestionario específico con una escala de Likert según el perfil de usuario. Si bien esta evaluación estuvo preparada para ser llevada a cabo al comienzo del ciclo lectivo, la misma tuvo que ser cancelada por las razones públicamente conocidas, provocadas por la pandemia del CONVID-19. A pesar de esto se considera que se pudieron cumplir los objetivos propuestos basados en la evaluación realizada durante el desarrollo.

## 6.2 Trabajos Futuros

Entre los posibles trabajos futuros que podrían realizarse podemos mencionar:

- **Extender la funcionalidad de Studium para obtener métricas sobre la resolución de cada juego de Blockly Games**: algunas de las métricas que podrían

ser de utilidad para los profesores podrían ser, por ejemplo la cantidad de intentos de ejecución realizados antes de resolver un juego exitosamente o determinar qué juegos fueron más difíciles de resolver para los y las estudiantes de un curso, entre otras.

- **Implementar una galería en Studium para que los jugadores puedan compartir sus creaciones en los juegos de ”Tortuga”, “Película” y “Música” de Blockly Games con otros estudiantes inscriptos en el curso**: mediante una galería los estudiantes podrían aprender de las creaciones de otros y estimulando así su creatividad.
- **Traducir la documentación ofrecida por Blockly Games para los juegos “Tutor de Estanque” y “Estanque” a idioma español**: la documentación ofrecida pretende ayudar con los tipos de bloques más complejos presentados por estos juegos. Si bien ésta se encuentra disponible en Studium, la misma solo está en idioma inglés.
- **Integrar otras herramientas de enseñanza de programación en las aulas a Studium**: durante este trabajo solo una herramienta fue integrada en Studium. Tomando como modelo esta integración, otras herramientas de las mismas características podrían ser agregadas al sistema haciendo éste más atractivo para su uso en las aulas.
- **Realizar la evaluación de Studium en las aulas**: si bien Studium fue evaluado durante el desarrollo, por los motivos explicados en el capítulo 5 de este trabajo el mismo no pudo ser presentado en las aulas como estaba inicialmente planeado. Esta evaluación debería ser el punto de partida para cualquier trabajo futuro que utilice Studium como base.

## Bibliografía


- **ATutor, (2020), ATutor Details.** https://atutor.github.io/atutor/index.html

- **Arena, D.A., Schwartz, D.L. (2013). Experience and explanation: using videogames to prepare students for formal instruction in statistics. Journal of Science Education and Technology, 1–11.**

- **Brooke, J. (2011), SUS - A quick and dirty usability scale.** http://www.tbistafftraining.info/smartphones/documents/b5_during_the_trial_usability_scale_v1_09aug11.pdf

- **Borsci, S., Federici, S., Malizia, A., De Filippis, M.L. (2019), Shaking the usability tree: why usability is not a dead end, and a constructive way forward** https://www.tandfonline.com/doi/pdf/10.1080/0144929X.2018.1541255?needAccess=true

- **Chamilo, (2020), Chamilo - Chamilo.org** https://chamilo.org/es/chamilo/

- **Chamilo, (2019), Guia de Instalación de Chamilo 1.11** https://11.chamilo.org/documentation/installation_guide_es_ES.html#2._Installation_of_Chamilo_LMS

- **Coble, R. (2016), Learning and Course Management Systems (LMS/CMS).** https://cft.vanderbilt.edu/learning-and-course-management-systems/

- **Coble, R. (2016), Learning and Course Management Systems (LMS/CMS).** https://cft.vanderbilt.edu/learning-and-course-management-systems/

- **Combefis, S., Beresnevicius, G., Dagiene, V. (2016), Learning Programming through Games and Contests: Overview, Characterisation and Discussion** https://ioinformatics.org/journal/v10_2016_39_60.pdf

- **Dickey, M.D. (2005). Engaging by design: how engagement strategies in popular computer and video games can inform instructional design. Educational Technology Research and Development, 53(2), 67–83.**

- **Google, (2019), Blockly Games.** https://github.com/google/blockly-games

- **Google, (2019), app.yaml Reference** https://cloud.google.com/appengine/docs/standard/python/config/appref

- **Grover, S., Pea, R., Cooper, S. (2015), Designing for deeper learning in a blended computer science course for middle school students. https://www.tandfonline.com/doi/abs/10.1080/08993408.2015.1033142**

- **Grover, S., Basu, S. (2017), Measuring Student Learning in Introductory Block-Based Programming: Examining Misconceptions of Loops, Variables, and Boolean Logic. https://www.researchgate.net/profile/Shuchi_Grover/publication/314296105_Measuring_Student_Learning_in_Introductory_Block-Based_Programming_Examining_Misconceptions_of_Loops_Variables_and_Boolean_Logic/links/5afafcb8458515c00b6cb5f0/Measuring-Student-Learning-in-Introductory-Block-Based-Programming-Examining-Misconceptions-of-Loops-Variables-and-Boolean-Logic.pdf**

- **Joshi, A., Kale, S., Chandel, S., Pal, D.K. (2015), Likert Scale: Explored and Explained. https://eclass.aspete.gr/modules/document/file.php/EPPAIK269/5a7cc366dd963113c6923ac4a73c3286ab22.pdf**

- **Nielsen, J. (2012), Usability 101: Introduction to Usability. https://www.nngroup.com/articles/usability-101-introduction-to-usability/?lm=design-everyday-things-revised&pt=book**

- **Prensky, M. (2003). Digital game-based learning. Computers in Entertainment (CIE), 1(1), 21.**

- **Queiruga C., Banchoff Tzancoff C., Venosa P, Gómez S., Morandi G. (2018). Ciencias de la Computación y escuelas ¿una didáctica específica?. En CACIC 2018 (Congreso Argentino de Ciencias de la Computación). UNICEN, Tandil, Argentina. Octubre de 2018. Libro de Actas XXIV CACIC 2018. pp 1040-1049. ISBN 978-950-658-472-6.**

- **Shute, V.J. (2011). Stealth assessment in computer-based games to support learning. In: S. Tobias and D. Fletcher (Eds.).Computer Games and Instruction. Information Age: Charlotte N. C., 503–524.**

- **Symfony (2020), Twig the flexible, fast, and secure template engine for PHP. https://twig.symfony.com/**

- **The Computer Language Co. Inc., (2019), Web server Definition from PC Magazine Encyclopedia. https://www.pcmag.com/encyclopedia/term/54342/web-server**

- **The Apache Software Foundation, (2019), About the Apache HTTP Server Project.**
  **http://httpd.apache.org/ABOUT_APACHE.html**

- **The Apache Software Foundation, (2019), Apache Virtual Host Documentation**
  **https://httpd.apache.org/docs/2.4/vhosts/**

- **Weintrop, D., Wlensky, U. (2018), How block-based, text-based, and hybrid block/text modalities shape novice programming practices.**

  **https://www.sciencedirect.com/science/article/pii/S2212868917300314**

- **Watson, W.R., Lee, S., Reigeluth, C.M. (2006), Learning Management Systems: An overview and roadmap of the systemic application of computers to education.**
  **https://www.researchgate.net/publication/288750144_Learning_management_systems_An_overview_and_roadmap_of_the_systemic_application_of_computers_to_education_122**

- **Watson, W.R., Lee, S., Reigeluth, C.M.(2006), Learning Management Systems: An overview and roadmap of the systemic application of computers to education.**
  **https://www.researchgate.net/publication/288750144_Learning_management_systems_An_overview_and_roadmap_of_the_systemic_application_of_computers_to_education_122**

# Anexo A: Instalación Local de Blockly Games

A continuación se detallan los pasos realizados para instalar una copia local de Blockly Games en un sistema Linux.

## Prerrequisitos

La implementación de Blockly Games se basa principalmente en código python y javascript. El código provisto para su instalación requiere que sea compilado utilizando el comando make.

Considerando estos requisitos se verificó que tanto el intérprete de python como el comando make estuvieran disponibles en el sistema. El sistema donde se realizó la instalación cuenta con las siguientes versiones de los requisitos:

- Python 2.7.12
- GNU Make 4.1

## Descarga de binarios

BlocklyGames ofrece dos posibilidades de instalación. Una de ellas es la versión completa que está pensada para instalar en un servidor web. Está cuenta todas las funcionalidades de BlocklyGames y podemos compilarla en cualquiera de los idiomas disponibles en la herramienta.

Por otro lado contamos con Blockly Games offline. Esta es una versión acotada pensada para instalaciones con conexión a internet limitada. No cuenta con la funcionalidad de guardado de estado de los juegos y puede ser compilada solo en un idioma.

Se consideró que, teniendo en cuenta que Studium debería poder mostrar información de la performance de cada estudiante, es fundamental poder guardar el estado de la ejecución de cada juego. Debido a esto se eligió analizar la opción de la versión completa de Blockly Games.

Se descargo el repositorio completo de BlocklyGames para su instalación local desde https://github.com/google/blockly-games.

Una vez descargado se descomprimio el archivo para obtener como resultado el directorio **blockly-games-master** que usaremos para la instalación.

## Configuración Virtual Host Apache

Para el análisis local de la herramienta BlocklyGames se decidió compilar el mismo dentro de un VirtualHost de apache. De esta forma podemos acceder fácilmente al mismo desde nuestro navegador y ejecutar cualquier test fácilmente.

Inicialmente se movió la carpeta **blockly-games-master** al path default de Apache, ubicado en el sistema de instalación en **/var/www/html**. Está luego fue renombrada a **blockly-local** para una mayor organización del sistema.

El próximo paso en la configuración del Virtual Host fue la creación del archivo de configuración del mismo, **blockly-local.conf**. Este archivo se creó en copia del archivo de configuración default provisto por Apache localizado en el directorio **/etc/apache2/sites-available** y se colocó en el mismo directorio.

Una vez creado el archivo su contenido fue modificado para que haga referencia al directorio **blockly-local** creado previamente y definir el nombre de nuestro servidor y su alias como “blockly-local”.

Adicionalmente, como puede verse en la Figura 41, se definieron algunos permisos para el mantenimiento del Virtual Host de Blockly Games y las rutas donde se guardan los logs de ser necesarios.

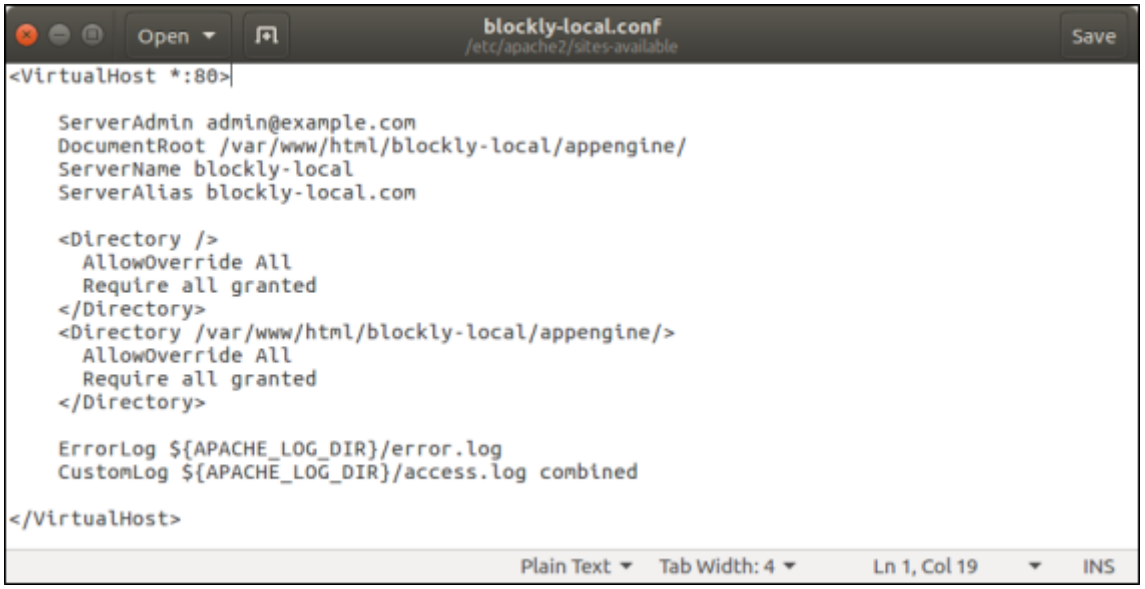


*Figura 41. Captura Texto - Contenido Archivo de Configuración VH BlocklyGames Local.*

Como ultimo paso se expuso la configuración del Virtual Host al servidor de Apache para poder acceder al mismo desde nuestro navegador. Para hacer esto habilitamos el archivo de configuración y se agregó el dominio blockly-local con la ip del localhost a la lista de hosts del sistema. Finalmente se reseteo el servidor para reflejar estos cambios. Estas acciones pueden verse en las Figuras 42 y 43.

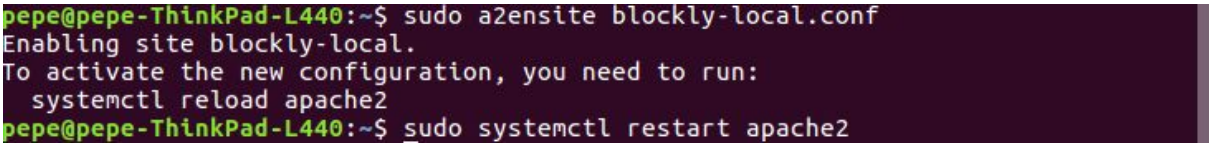


*Figura 42. Captura Linea de Comandos - Habilitación VH BlocklyGames Local.*

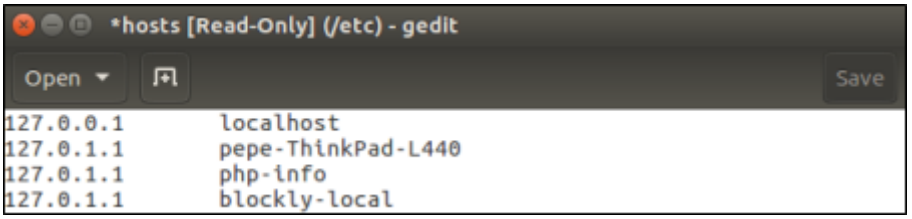


*Figura 43. Captura Texto - Modificación Archivo /etc/hosts para Configuración VH BlocklyGames Local.*

## Compilado del código

Una vez descargado se compilaron las dependencias del mismo utilizando make. Blockly Games requiere que se realice la compilación de las dependencias de terceros antes de comenzar a compilar las diferentes partes del código como se muestra en la Figura 44. Al correr este comando por primera vez se creará el directorio **third-party-downloads** dentro de nuestro directorio raíz y se actualizará con todas las librerías necesarias para la ejecución de Blockly Games.

```
pepe@pepe-ThinkPad-L440:/var/www/html/blockly-local$ sudo make deps
```

*Figura 44. Captura Linea de Comandos - Compilación de dependencias BlocklyGames Local.*

Para continuar con la compilación Blockly Games ofrece varias opciones. Entre ellas encontramos:

- Compilar todo el código en idioma inglés.
- Compilar todo el código en todos los idiomas disponibles.
- Seleccionar qué partes del código compilar en idioma inglés.
- Seleccionar qué partes del código compilar en todos los idiomas disponibles.

Para poder acceder a la plataforma con idioma español nos vimos forzados a realizar la compilación en todo los idiomas disponibles. Considerando que hay +50 idiomas disponibles la compilación de todo el código lleva un tiempo considerable. Por esta razón, para realizar el análisis de la herramienta, se decidió compilar solo la página de inicio y el juego Laberinto como se muestra en las Figura 45.

```
pepe@pepe-ThinkPad-L440:/var/www/html/blockly-local$ sudo make index; sudo make maze;
```

*Figura 45. Captura Linea de Comandos - Compilación de página de inicio y juego Laberinto BlocklyGames Local.*

Al terminar la compilación se cambió el owner a **www-data:www-data** de todos los directorios y archivos del directorio blockly-local.

Finalmente podemos validar la instalación, como se muestra en la Figura 46, accediendo a la página de inicio de blockly-local desde nuestro navegador sin inconvenientes.


*Figura 46. Captura Navegador - Página de Inicio BlocklyGames local.*

## Errores de Compatibilidad

BlocklyGames fue diseñado para ejecutarse en el servidor App Engine de Google. Por esta razón el mismo presenta ciertos problemas de compatibilidad al realizar la instalación en un servidor Apache.

El primero de estos problemas es que la funcionalidad que permite guardar el estado de los juegos solo funciona en App Engine. Esto es así debido a que la implementación está pensada para guardar la información usando el servicio de almacenamiento en la nube provisto por Google y no en una base de datos local. Este problema fue afrontado durante la implementación de Studium como se muestra más adelante en este documento.

El segundo problema encontrado está relacionado con la forma que tiene Blockly Games de redireccionar a cada uno de los juegos. Al intentar abrir el juego de **Laberinto**, previamente compilado, se muestra el contenido del directorio del mismo y no el juego en sí, como se puede apreciar en la Figura 47.

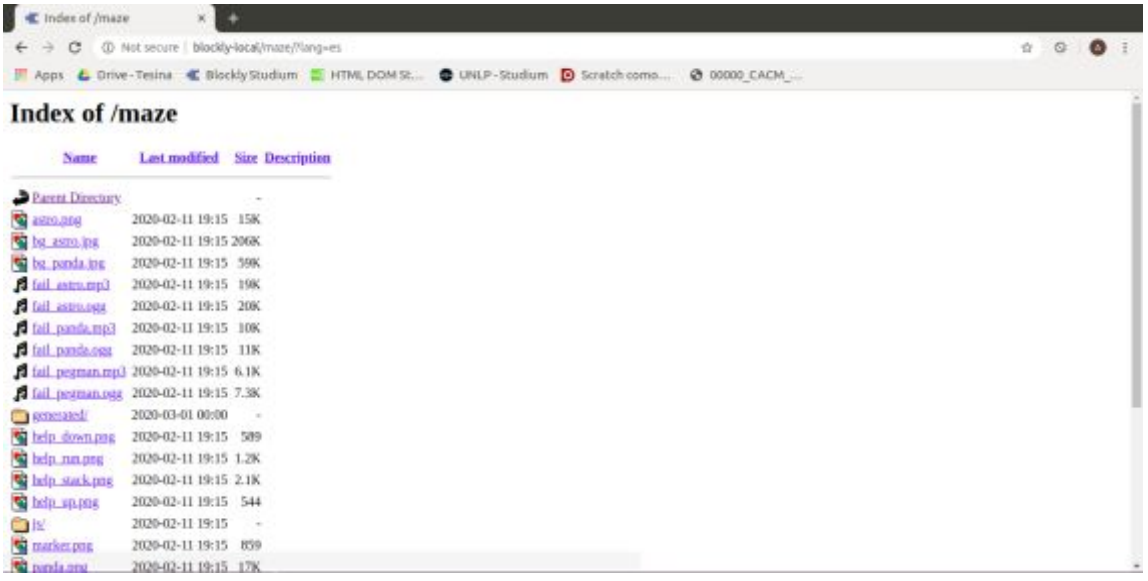


*Figura 47. Captura Navegador - Error compatibilidad juego Laberinto BlocklyGames local.*

Después de investigar algunos de los incidentes registrados en el repositorio de Blockly Games se constató que algunos de ellos tenían cierta similitud con el problema encontrado en la instalación local. Para obtener más información relacionada a este se decidió crear el incidente #119, con referencia al issue #18, en el repositorio de BlocklyGames. En este incidente se describió el proceso de instalación realizado y el problema como se muestra en la Figura 48.

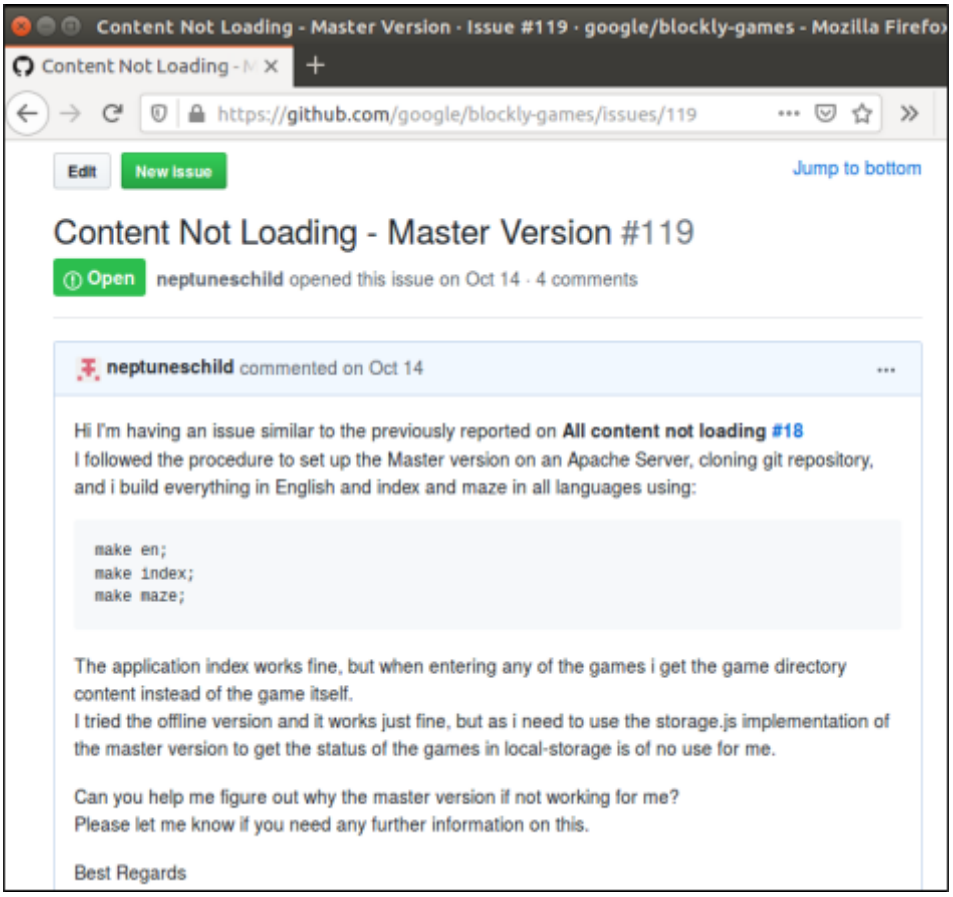


*Figura 48. Captura Navegador - Incidente compatibilidad juego Rompecabezas Repositorio BlocklyGames.*

Como respuesta a este incidente se obtuvo una explicación más detallada de porqué ocurre el problema y dos posibles soluciones al mismo. El problema ocurre debido a que las redirecciones de la aplicación están configuradas en el archivo app.yaml para un servidor App Engine. Este archivo es donde normalmente se guarda la configuración de la aplicación en un servidor App Engine. Entre otras funciones, específica que rutas URL se corresponden con los controladores de solicitudes y con los archivos estáticos. (Google, 2019). Al instalarse la aplicación en un servidor Apache se propuso dos formas de hacer que el servidor realice correctamente las redirecciones:

- Traducir las directivas en el archivo app. yaml para que correspondan a una configuración de Apache.
- Modificar el archivo js/lib-games.js de forma tal que se defina la constante BlocklyGames.IS_HTML=true; y luego compilar nuevamente el código.

Debido a su menor complejidad se optó por la segunda opción propuesta como se muestra en la Figura 49. Luego se re compilo el código de index y el juego Laberinto de la misma forma en que se hizo en la sección anterior: “Compilado del código”.

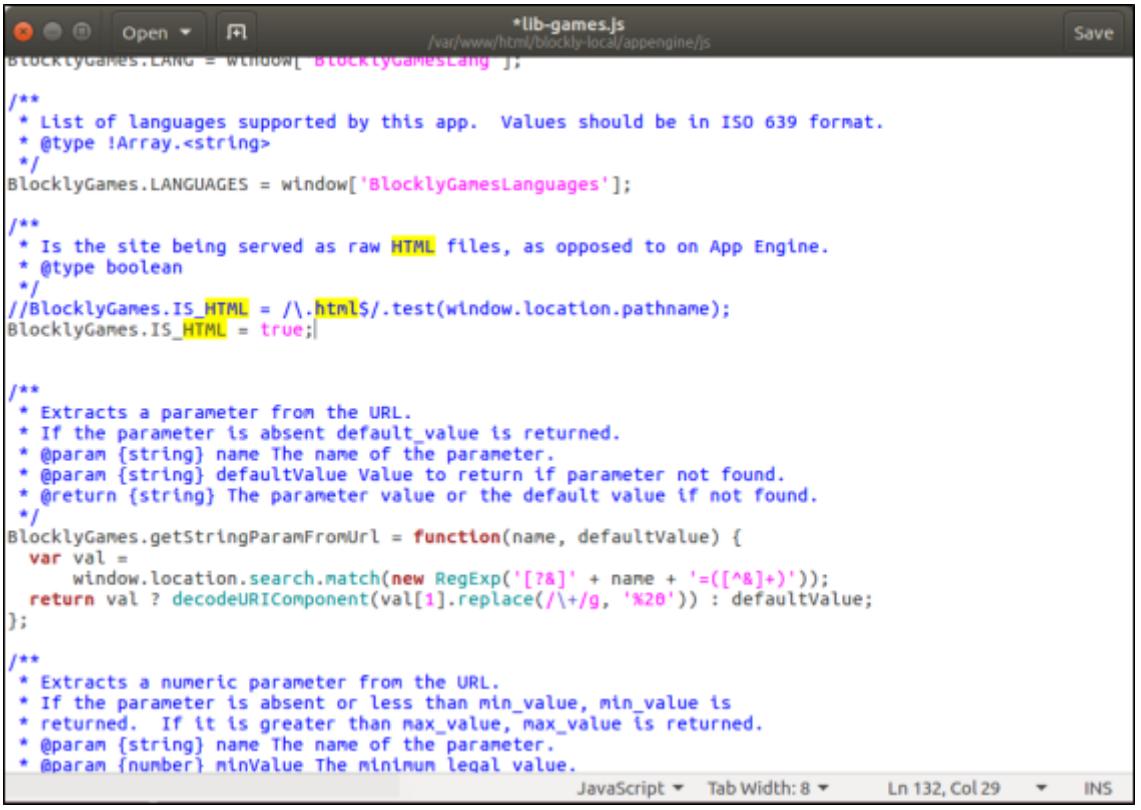


*Figura 49. Captura Texto - Modificación archivo lib-games.js BlocklyGames Local.*

Después de aplicar estos cambios podemos acceder al juego Rompecabezas sin mayores inconvenientes como se muestra en la Figura 50.

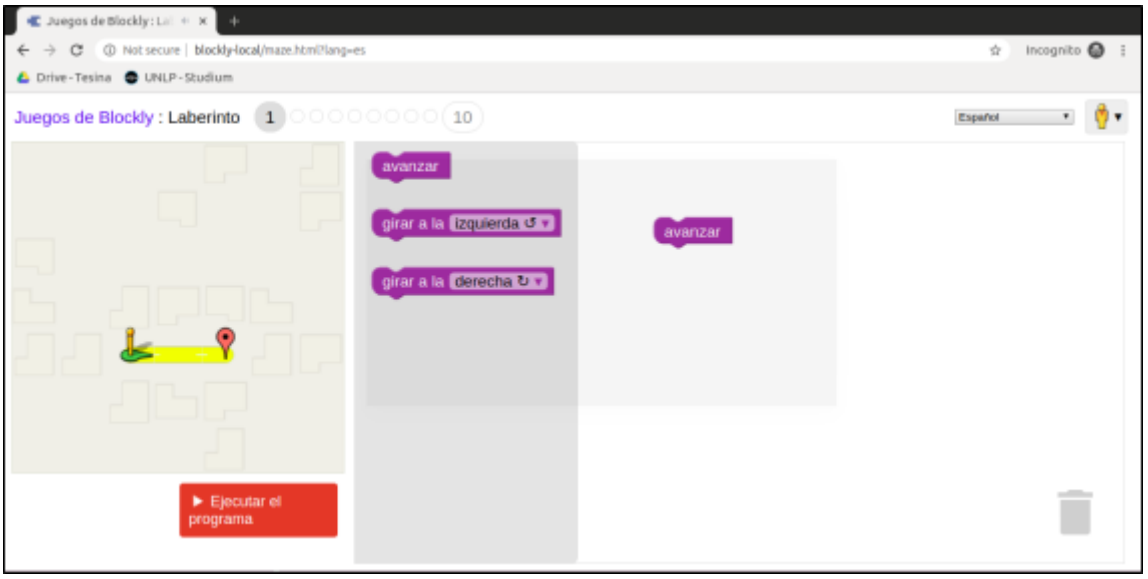


*Figura 50. Captura Navegador - Juego Rompecabezas BlocklyGames local.*

# Anexo B: Instalación Local de Chamilo

A continuación se detallan los pasos realizados para instalar una copia local de Chamilo en un sistema Linux.

## Prerrequisitos

Considerando los pre-requisitos de Chamilo, el ambiente donde se realizó las instalación local contó con las siguientes versiones de estos:

- Apache 2.4.18
- MySQL 5.7.28
- PHP 7.2.24

También fueron requeridos durante la instalación los siguientes paquetes de php:

- php7.2-mbstring
- php7.2-xml
- php7.2-mysql
- php7.2-dev
- php7.2-zip
- php7.2-gd
- php7.2-intl
- php7.2-curl
- php7.2-apcu
- php7.2-ldap
- libzip-dev

Como requisito también se realizó la instalación de zip, una extensión para crear, modificar y leer archivos zip. Está se descarga desde https://pecl.php.net/package/zip y se instala usando el comando mostrado en la Figura 51.


*Figura 51. Captura Linea de Comandos - Instalación librería zip.*

## Descarga de binarios

Chamilo ofrece dos versiones según la versión de PHP disponible en el servidor donde será instalado, ya sea PHP 5.6 o PHP 7.
Ambas versiones pueden ser descargadas desde https://chamilo.org/es/descargar/.

Para esta instalación se eligió la versión Chamilo LMS 1.11.10 para PHP 7 en formato TAR.GZ.

## Configuración base de datos

Para la instalación fue necesario contar con un nombre de usuario y contraseña que permitiera administrar la base de datos que usa la aplicación. Se consideró que en la guía

de instalación disponible en su sitio web se recomienda definir un usuario con acceso sólo a una base de datos específica.

Como se muestra en la Figura 52, como primer paso se creó una base de datos para Chamilo para poder asignar permisos al nuevo usuario.

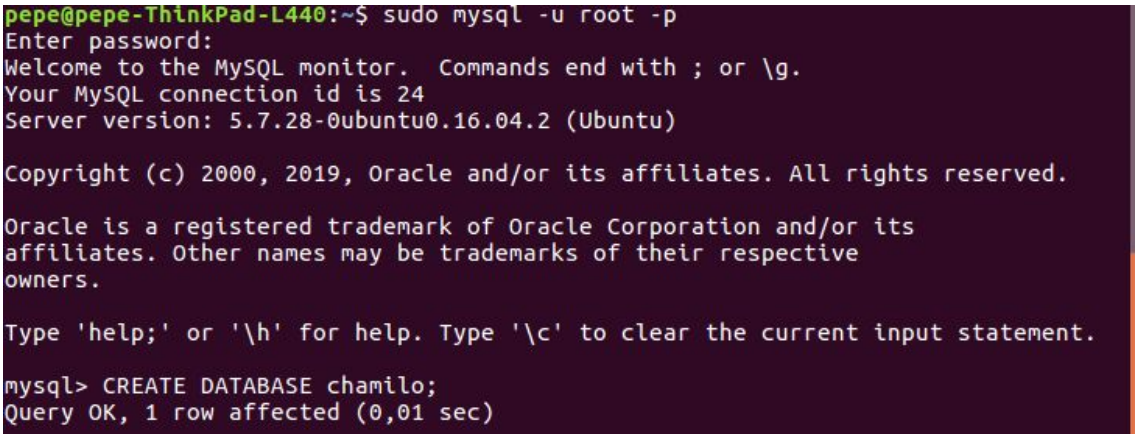


*Figura 52. Captura Linea de Comandos - Creación base de dato Chamilo.*

Luego se creó el usuario 'chamilouser'@'localhost' con una password genérica como puede verse en la Figura 53.

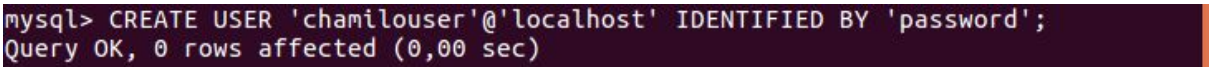


*Figura 53. Captura Linea de Comandos - Creación usuario MySQL chamilouser.*

Finalmente, como se muestra en la Figura 54, se asignó el usuario 'chamilouser'@'localhost' a la base de datos chamilo y se actualizaron los permisos de MySQL.

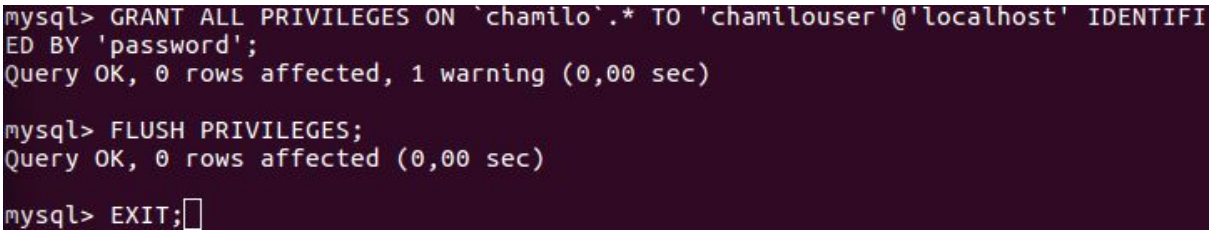


*Figura 54. Captura Linea de Comandos - Configuración permisos MySQL usuario chamilouser.*

## Configuración Virtual Host Apache

Para la instalación de la copia local de Chamilo se decidió configurar un Host Virtual de Apache.

En la Figura 55 pueden apreciarse los pasos necesarios para la creación del directorio de instalación de Chamilo en Apache. Como primer paso se procedió a mover la carpeta chamilo-1.11.10, obtenida al descomprimir la versión descargada de Chamilo, al path por defecto de Apache en el sistema de instalación, **~/var/www/html**. Luego se cambió el propietario de todos los archivos de chamilo-1.11.10 a www-data:www-data, garantizando

así que el servidor tendrá accesos a estos, y se asignaron los permisos necesarios para su ejecución y lectura.

```
pepe@pepe-ThinkPad-L440:/var/www/html$ sudo mv ~/Downloads/chamilo-1.11.10 .
pepe@pepe-ThinkPad-L440:/var/www/html$ ls -al
total 32
drwxr-xr-x  5 root     root      4096 nov 29 19:34 .
drwxr-xr-x  3 root     root      4096 nov 21 14:07 ..
drwxr-xr-x 12 pepe     pepe      4096 may  9  2019 chamilo-1.11.10
-rwxr-xr-x  1 root     root     11321 nov 21 14:07 index.html
drwxr-xr-x  2 www-data www-data  4096 nov 22 18:29 php
drwxr-xr-x  3 www-data www-data  4096 nov 22 19:58 studium
pepe@pepe-ThinkPad-L440:/var/www/html$ sudo chown -R www-data:www-data chamilo-1
.11.10/
pepe@pepe-ThinkPad-L440:/var/www/html$ sudo chmod -R 755 chamilo-1.11.10/
pepe@pepe-ThinkPad-L440:/var/www/html$ ls -al
total 32
drwxr-xr-x  5 root     root      4096 nov 29 19:34 .
drwxr-xr-x  3 root     root      4096 nov 21 14:07 ..
drwxr-xr-x 12 www-data www-data  4096 may  9  2019 chamilo-1.11.10
-rwxr-xr-x  1 root     root     11321 nov 21 14:07 index.html
drwxr-xr-x  2 www-data www-data  4096 nov 22 18:29 php
drwxr-xr-x  3 www-data www-data  4096 nov 22 19:58 studium
```

*Figura 55. Captura Linea de Comandos - Configuración Root Folder Chamilo Local VH Apache.*

Para la creación del archivo **chamilo-local.conf**, que contiene la configuración del Host Virtual de Apache, se creó en copia del archivo de configuración por defecto provisto por Apache y se localizó en el directorio **~/etc/apache2/sites-available/**.

Una vez creado el archivo su contenido fue modificado para que haga referencia a la carpeta chamilo-1.11.10, creada previamente, y definir el nombre de nuestro servidor y su alias como “chamilo-local” tal y como se muestra en la Figura 56.

Open *chamilo-local.conf /etc/apache2/sites-available Save

```
<VirtualHost *:80>

    ServerAdmin admin@example.com
    DocumentRoot /var/www/html/chamilo-1.11.10/
    ServerName chamilo-local
    ServerAlias chamilo-local.com

    <Directory />
      AllowOverride All
      Require all granted
    </Directory>
    <Directory /var/www/html/chamilo-1.11.10/>
      AllowOverride All
      Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>
```

*Figura 56. Captura Texto - Contenido Archivo de Configuración VH Chamilo Local.*

Como último paso se habilitó la configuración de **chamilo-local** en el servidor de Apache. Como se muestra en la Figura 57, esto se realizó agregando el archivo de configuración a la lista de sitios habilitados del servidor y reiniciando el servidor para aplicar los cambios.

```
pepe@pepe-ThinkPad-L440:~$ sudo a2ensite chamilo-local.conf
Enabling site chamilo-local.
To activate the new configuration, you need to run:
  systemctl reload apache2
pepe@pepe-ThinkPad-L440:~$ sudo systemctl restart apache2
```

*Figura 57. Captura Linea de Comandos - Habilitación VH Chamilo Local.*

Finalmente se agregó el dominio **chamilo-local** con la dirección IP del servidor local (localhost) a la lista de hosts del sistema localizada en **~/etc/hosts**. Este último paso permitió al navegador mapear la uri **chamilo-local** con el servidor local.

## Configuración Redirecciones

Chamilo LMS requiere que el servidor web redireccione las solicitudes recibidas, por lo tanto fue necesario realizar algunas configuraciones adicionales en el servidor y en el virtual host. En Apache existen dos opciones para realizar esta tarea: habilitando el módulo Rewrite y permitiendo reemplazos (a través de .htaccess) o agregando una sección de configuración específica al Virtual Host.

La primera opción es la utilizada por defecto y para su fácil implementación Chamilo provee el archivo .htaccess con todas las reglas de redirección utilizadas por la aplicación. Sin embargo, en las pruebas realizadas, esta opción demostró no funcionar correctamente dentro de un Virtual Host de Apache dado que el servidor tiene problemas para leer correctamente las reglas del archivo .htaccess.

Considerando este inconveniente se decidió utilizar la segunda opción disponible. Ésta fue copiar el contenido del archivo .htaccess al archivo de configuración del Virtual Host, chamilo-local.conf. Luego se eliminó el archivo .htaccess de manera tal que solo se tenga en cuenta la configuración del Virtual Host.

Adicionalmente se detectó un error en algunas de las reglas provistas por Chamilo. Algunas de las reglas omitían el carácter “/” al comienzo del path en su configuración. Esto causó que no se ejecutarán las mismas y no se pudiera acceder al recurso que apuntaban.

Para solucionar este problema se modificaron las reglas agregando el carácter “/” al comienzo de cada una de ellas que lo omitían. Finalmente se habilitó el módulo de rewrite de Apache y se reinicio el servidor para reflejar los cambios realizados.

## Instalación Guiada

Una vez terminadas las configuraciones anteriores fue posible acceder a la copia local de Chamilo desde un navegador utilizando la url “http://chamilo-local”.

Al acceder a Chamilo por primera vez, como se muestra en la Figura 58, se tiene acceso a la página de bienvenida de Chamilo desde donde se puede acceder a la instalación guiada de la plataforma.


*Figura 58. Captura Navegador - Pagina bienvenida chamilo-local.*

El primer paso de la instalación guiada permite elegir el idioma de la instalación y tener un vistazo de las diferentes etapas de la misma. Teniendo en cuenta que el entorno a desarrollar utilizara idioma Español se eligió la opción de instalación de Español Latino.

Luego el sistema procede a realizar el análisis de requerimientos de Chamilo. Este análisis muestra una lista de librerías y características que el servidor debe proporcionar para poder utilizar Chamilo con todas sus funcionalidades. Junto a los nombre de cada uno de estos requerimientos se indica, resaltando en colores, el estado del requerimiento:

- Verde: el requerimiento es cumplido por nuestro servidor.
- Rojo: el requerimiento no está disponible en el servidor y es mandatorio para la ejecución de una funcionalidad crítica de Chamilo.
- Naranja: el requerimiento no está disponible en el servidor y no afecta la funcionalidad crítica de Chamilo.

Antes de proceder con la instalación se validó que todos los requerimientos mandatorios estuvieran marcados como cumplidos como puede verse en la Figura 59.

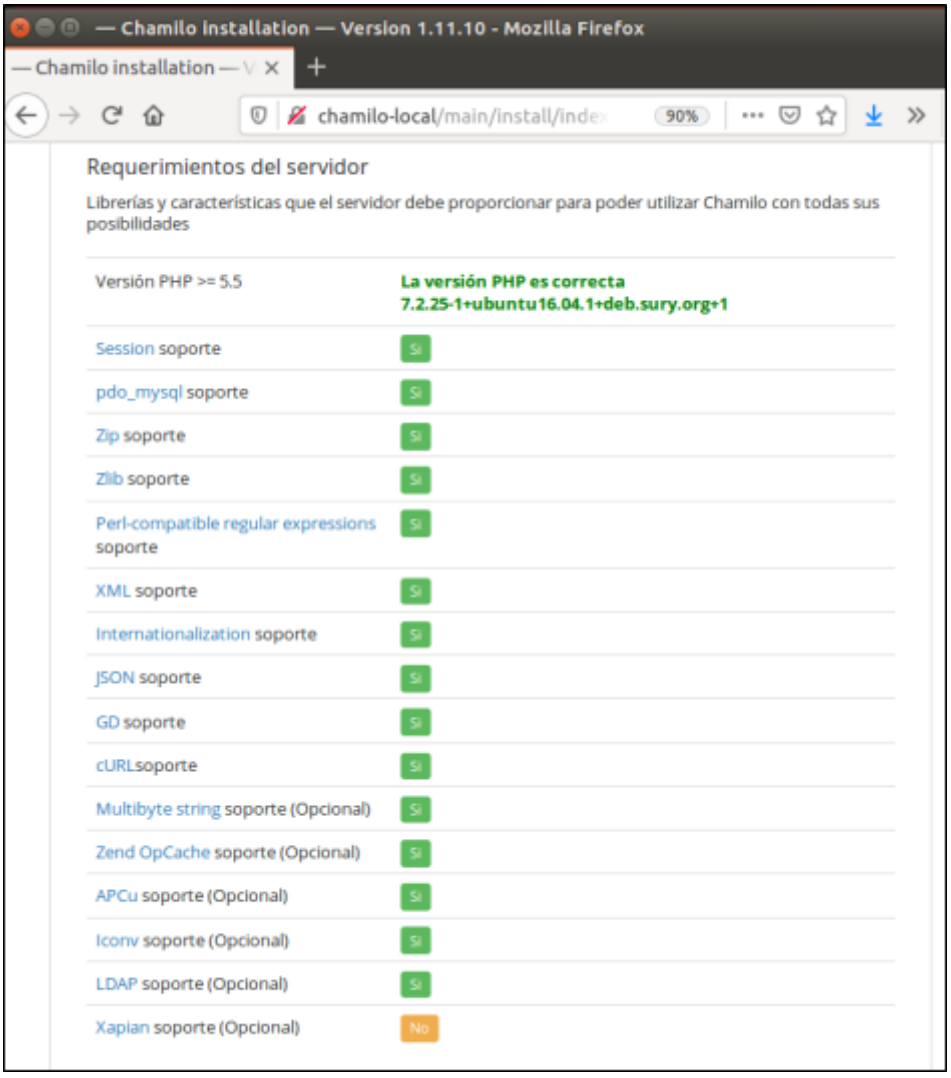


*Figura 59. Captura Navegador - Requerimientos del servidor de instalacion chamilo-local.*

A continuación de los requerimientos del servidor se pueden observar una serie de configuraciones recomendadas para el servidor en dos columnas. Tal como muestra la Figura 60, la primer columna contiene el valor recomendado de cada parámetro. La segunda columna contiene el valor actual de dicho parámetro en el archivo **php.ini** del servidor. Este archivo de configuración normalmente se encuentra en **~/etc/php/7.2/apache2/php.ini** en los sistemas operativos Linux. Si bien no es mandatorio respetar estos valores, es recomendable para que Chamilo se ejecute de forma óptima.

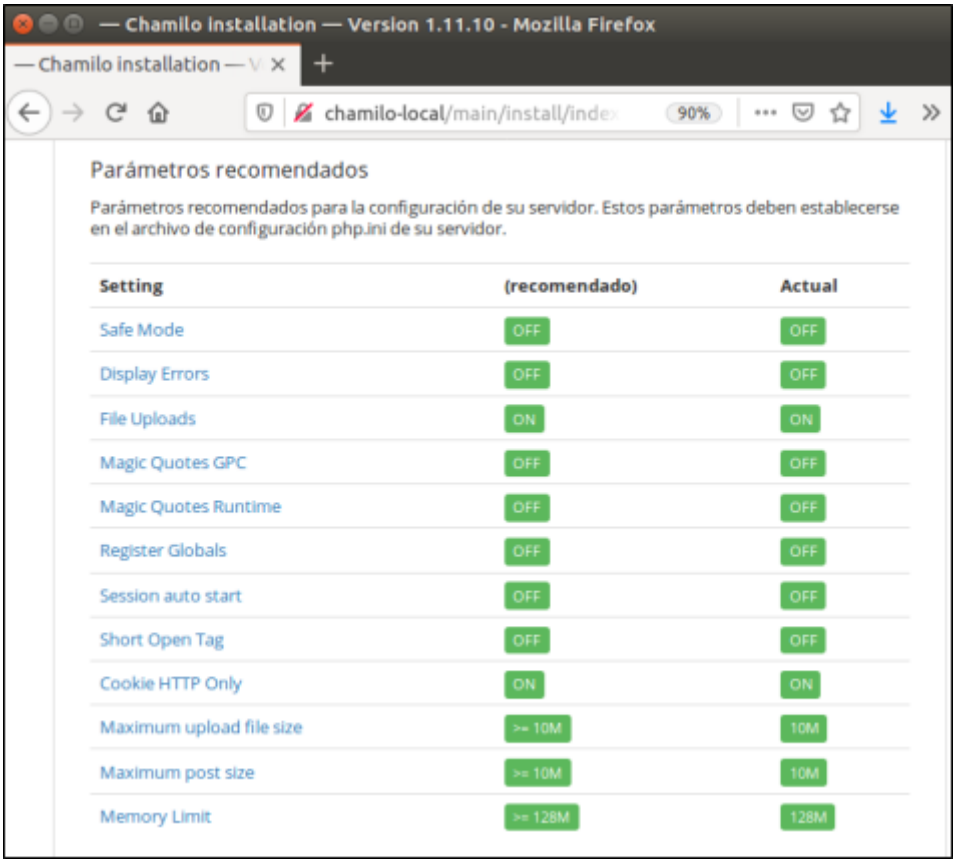


*Figura 60. Captura Navegador - Parámetros recomendados de instalacion chamilo-local.*

Finalmente puede verse, en la Figura 61, un tercer reporte que muestra los permisos que deben tener ciertos directorios y archivos de la carpeta de instalación de Chamilo.

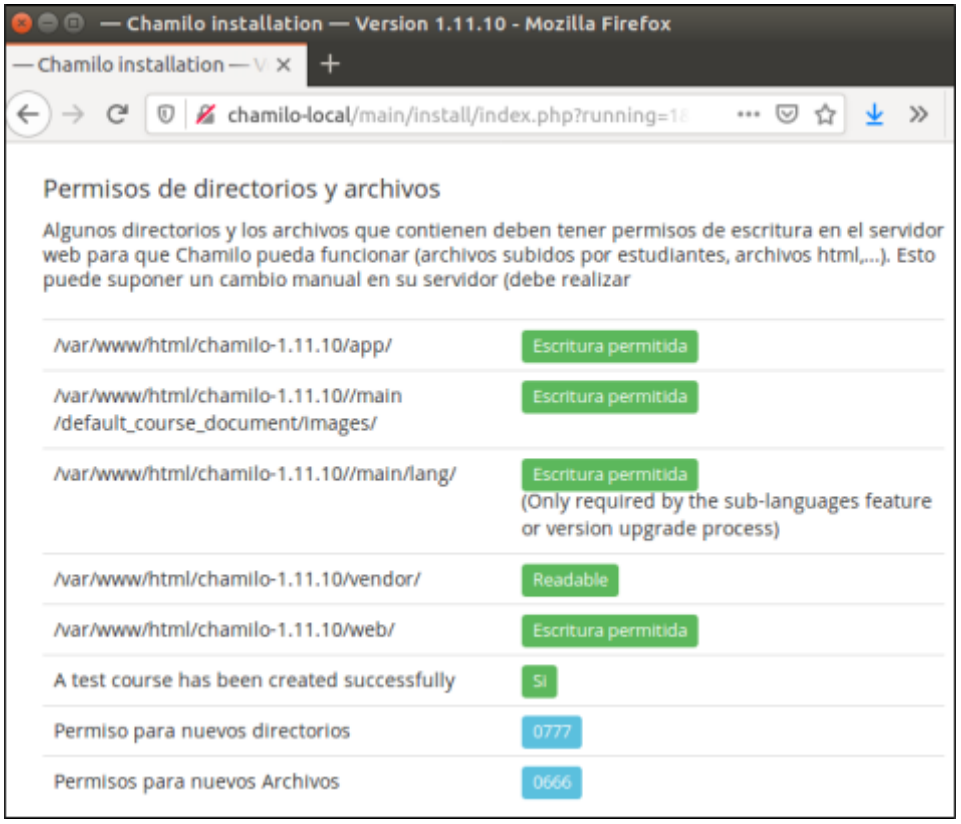


*Figura 61. Captura Navegador - Permisos de directorios y archivos de instalacion chamilo-local.*

Una vez validados todos los requerimientos anteriores se procedió a avanzar con la instalación.

El paso 4 de la instalación pide definir los valores relacionados al motor de base de datos a ser utilizado por Chamilo. En este formulario se utilizó el usuario y la base de datos creados previamente siguiendo los pasos de la sección Configuración Base de Datos. Una vez completados todos los datos se validó que la conexión funciona correctamente con el botón “Comprobar la conexión a la base de datos” y se avanzó al próximo paso de la instalación.

Al finalizar el último paso se notifica que la instalación fue exitosa, como se muestra en la Figura 62, y se permite acceder a la plataforma instalada.

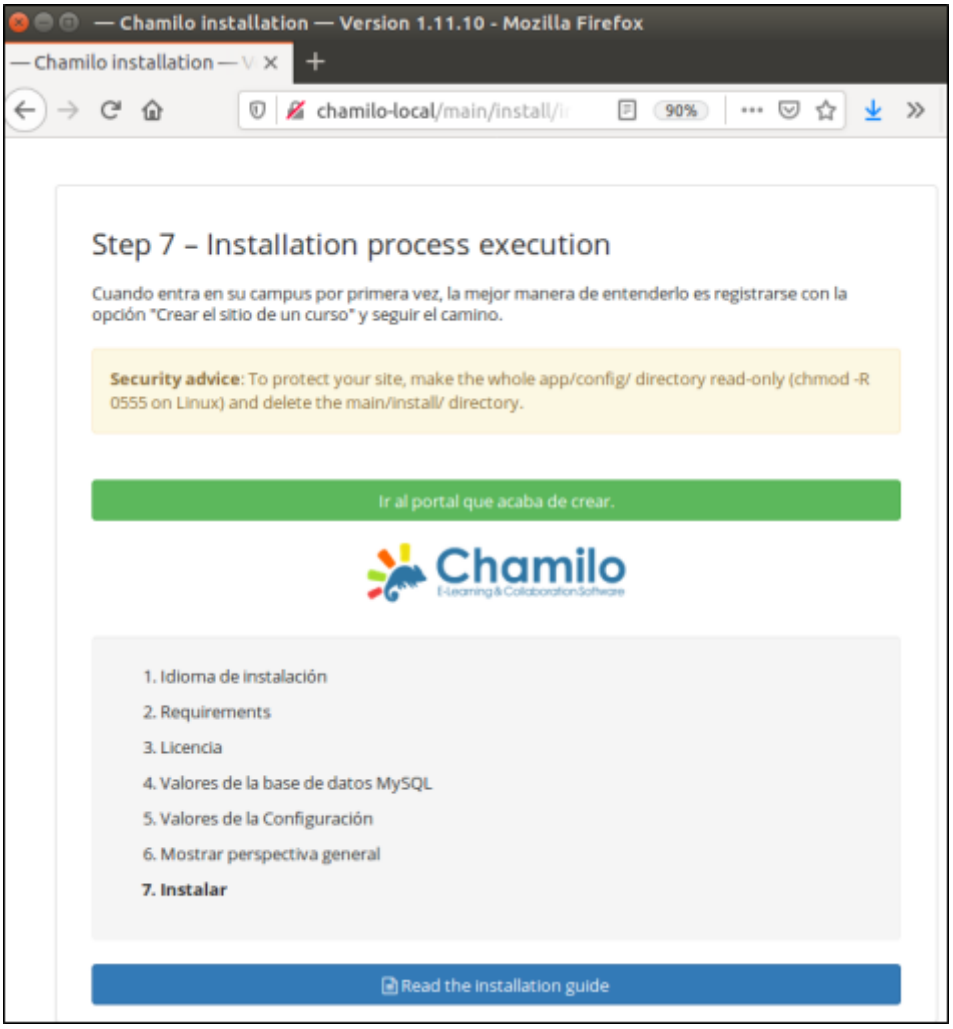


*Figura 62. Captura Navegador - Mensaje Instalación exitosa Chamilo.*