DISEÑO DE INFRAESTRUCTURA PARA LA IMPLEMENTACIÓN DE UN JUEZ ONLINE PARA LOS ENTRENAMIENTOS DEL GRUPO DE MARATONES DE PROGRAMACIÓN DE LA UNIVERSIDAD ECCI

MIGUEL ÁNGEL CASTELLANOS IBÁÑEZ

JHON FREDY PLAZAS HURTADO

UNIVERSIDAD ECCI

Facultad de Ingeniería de Sistemas

Bogotá D.C.

2019

Miguel Ángel Castellanos Ibáñez – Jhon Fredy Plazas Hurtado
miguela.castellanosi@ecci.edu.co – jhonf.plazash@ecci.edu.co

DISEÑO DE INFRAESTRUCTURA PARA LA IMPLEMENTACIÓN DE UN JUEZ ONLINE PARA LOS ENTRENAMIENTOS DEL GRUPO DE MARATONES DE PROGRAMACIÓN DE LA UNIVERSIDAD ECCI

Presentado por:

MIGUEL ÁNGEL CASTELLANOS IBÁÑEZ

JHON FREDY PLAZAS HURTADO

Presentado a:

Msc Ing. WILLIAM FRASSER ACEVEDO

Director de Proyecto

Msc Ing. OSCAR ALBERTO ZAMBRANO OSPINA

Asesor Metodológico

UNIVERSIDAD ECCI

Programa tecnología en desarrollo informático

Proyecto de Grado

Bogotá

2019

Agradecimientos

Son varias las personas que han contribuido al proceso y conclusión de este trabajo. En primer lugar, queremos agradecer al Msc Ing. William Frasser Acevedo, quien fue el primero que creyó en este proyecto, nos apoyó de manera personal y académica y alentó para que concluyera esta investigación.

Igualmente, al Msc Ing. Oscar Alberto Zambrano Ospina y a la Msc Ing. Ana Yaqueline Chavarro Peña por su colaboración, apoyo, amabilidad y acompañamiento brindado a lo largo de este proceso.

A nuestras familias por su apoyo y sacrificio brindado ya que nos dieron fuerzas para lograr obtener una de nuestras primeras metas deseadas.

Y finalmente a todas las personas y compañeros, en especial al Tnlgo. Diego Fernando Rodríguez Castañeda que desinteresadamente brindo sus conocimientos en pro del presente proyecto.

Tabla de Contenido

Abreviaturas

ACM

Association for Computing Machinery

ICPC

International Collegiate Programming Contest

ACSL

American Computer Science League

ASF

Apache Software Foundation

BOCA

Online Contest Administrator

BSD

Berkeley Software Distribution

FSF

Free Software Foundation

GNU

GNU's Not Unix

GPL

General Public License

IPSC

Internet Problem Solving Contest

MIT

Instituto Tecnológico de Massachusetts

NASSP

Asociación Nacional de Directores de Escuelas Secundarias

OMPC

Organización de Maratones de Programación

OSI

Open Source Initiative

SLPC

Stanford Local Programming Contest

Lista de Figuras

Lista de Tablas

Lista de Anexos

Glosario

Archivos parches

Son archivos que contienen cambios que se aplican a un software, para corregir errores, agregarle funcionalidad, actualizarlo, etc.

Código fuente

Es un conjunto de líneas de texto con los pasos que debe seguir la computadora para ejecutar un programa

Copyleft

Es un método general para convertir un programa u otro tipo de trabajo en software libre y exigir que todas las versiones del mismo, modificadas o ampliadas, también lo sean.

Google Code Jam

Es una competencia de programación patrocinada y administrada por Google

MIT License

Es una licencia de software libre la cual garantiza al usuario final los derechos de copiar, modificar, fusionar, distribuir libremente cualquier software (sin Copyleft).

Open Source Initiative

Es una organización dedicada a la promoción del código abierto

Open Source

Es un modelo de desarrollo de software basado en la colaboración abierta. Se enfoca más en los beneficios prácticos (acceso al código fuente) que en cuestiones éticas o de libertad que tanto se destacan en el software libre.

Plataformas E-Learning

Es un espacio virtual de aprendizaje orientado a facilitar la experiencia de capacitación a distancia, tanto para empresas como para instituciones educativas.

Abstract

The present project determines the infrastructure of software and hardware necessary for the implementation of an Open Source judge, which allows the development of programming competencies and contributes to potentiate the performance of the ECCI university's marathon of programming group.

Through this document, the problem is exhibited, the study of the different Online Open Source judges is carried out, the deployment of the software and hardware infrastructure, the manuals of the administrator, judge, team and the public, besides a guide that provides the installation process of the tool to choose, to finally carry out a pilot test.

Miguel Ángel Castellanos Ibáñez – Jhon Fredy Plazas Hurtado
miguela.castellanosi@ecci.edu.co – jhonf.plazash@ecci.edu.co

Resumen

El presente proyecto determina la infraestructura de software y hardware necesarios para la implementación de un juez Open Source, que permite el desarrollo de competencias de programación y contribuye en potencializar el desempeño del grupo de maratones de programación de la universidad ECCI.

A través del presente documento, se exhibe la problemática, se realiza el estudio de los diferentes jueces en línea Open Source, se lleva a cabo el despliegue de la infraestructura de software y hardware, se elaboran los manuales del administrador, juez, equipo, público y una guía que proporciona el proceso de instalación de la herramienta a escoger y finalmente los resultados de la prueba piloto.

Miguel Ángel Castellanos Ibáñez – Jhon Fredy Plazas Hurtado
miguela.castellanosi@ecci.edu.co – jhonf.plazash@ecci.edu.co

Introducción

Las maratones de programación son competencias en el que grupos de estudiantes de diferentes carreras universitarias se desafían para demostrar sus conocimientos, tácticas y habilidades para el desarrollo de algoritmos. Estas competencias disponen de un juez en línea que lee, compila y compara los algoritmos enviados, para luego retornar cinco posibles resultados *correct*, *wrong-answer*, *compilation error*, *runtime error*, y finalmente *time limit* que se da en el caso de que una solución haya sobrepasado un tiempo fijado, lo cual es ideal ya que obliga a los equipos a desarrollar algoritmos más eficientes para dar solución a un problema.

Ahora bien, el modo de estudio de esta área requiere de práctica y dedicación, más si se quiere destacar tanto en el área laboral como social, por eso en la actualidad el uso de jueces en línea está adquiriendo una mayor transcendencia, no tan solo enfocándose en el ámbito competitivo, sino también en el académico, logrando así celebrar concursos de programación sobre estas plataformas.

En este trabajo de grado, se realizó un estudio que permitió seleccionar el mejor juez en línea Open Source, posibilitando la elaboración de manuales de usuario y de infraestructura de software y hardware necesarios para la implementación de esta plataforma. Finalmente se da a conocer los resultados de la prueba piloto que se llevó a cabo en la universidad ECCI, permitiendo así demostrar el pleno funcionamiento de la plataforma.

Capítulo 1. Título de la Investigación

Diseño de infraestructura para la implementación de un juez online orientado al grupo de maratones de programación de la universidad ECCI.

Miguel Ángel Castellanos Ibáñez – Jhon Fredy Plazas Hurtado
miguela.castellanosi@ecci.edu.co – jhonf.plazash@ecci.edu.co

Capítulo 2. Problema de Investigación

2.1Descripción del Problema

Las maratones de programación son competencias en el cual equipos conformados por estudiantes (1-3 personas) de diferentes carreras universitarias, participan y se enfrentan a problemas de algoritmia de alto nivel, con el fin de resolver el mayor número de ejercicios en el menor tiempo posible, en un rango de 5 horas de competencia. En la actualidad existen diversas competencias auspiciadas a nivel local por ligas y asociaciones como: RPC (Red de programación competitiva), nacionales por ACIS (Asociación Colombiana de Ingenieros de Sistemas), REDIS (Red Nacional de Programas de Ingeniería de Sistemas) y CCPL (Colombian Collegiate Programming League) y finalmente las competencias regionales y mundiales que son auspiciadas por ACM/ICPC (Association for Computer Machinery/International Collegiate Programming Contest) y patrocinadas por la empresa IBM.

En Colombia ACM/ICPC realiza maratones de programación los cuales son llamados circuitos, y es una manera de prepararse para la maratón regional, en la cual participan otros equipos de universidades próximas geográficamente. Hay más de 30 regiones en todo el mundo. En cada una de estas regiones se agrupan diferentes países y los mejores clasificados en cada competencia regional participaran en la final mundial que es celebra anualmente en diferentes partes del mundo.

Igualmente les permite a los estudiantes más sobresalientes del país, mostrar su talento para la resolución de algoritmos lo cual les da la posibilidad a los participantes en mostrar su talento ante grandes empresas del sector tecnológico como Google, Microsoft, IBM, entre

otros, quienes frecuentemente convocan a los participantes más destacados con el fin de formar parte de su equipo de trabajo. Un ejemplo de ello es el caso del equipo UNBounded de la Universidad Nacional quien se destacó en la Maratón Regional y logro pasar a la Maratón Mundial de programación versión 41 del año 2017 y que compitió contra universidades como el MIT (Massachusetts Institute of Technology) y con países como Rusia y China, potencias en programación(“Equipo de la U.N., a Maratón Mundial de Programación,” n.d.).

La enseñanza de la algoritmia ha presentado un reto para los docentes en todos los niveles educativos, en conjunto con el desarrollo del pensamiento abstracto necesarios para creación de programas eficientes. El grupo de maratones de programación, no cuenta con un recurso que complemente el desarrollo de habilidades de lógica en programación y algoritmia, que permita reforzar el aprendizaje, y que conceda a los docentes del área de programación y a fines, utilizar este servicio para evaluar los conocimientos provistos en clases y que vaya de la mano con la administración de una maratón de programación.

2.1 Formulación del Problema

¿Qué infraestructura tecnológica debe implementarse para potencializar el desempeño del grupo de maratones de programación de la universidad ECCI, propiciando así, un espacio de retroalimentación previo y posterior a una maratón?

Capítulo 3. Objetivos de la Investigación

3.1 Objetivo General

Determinar una infraestructura de software y hardware, para la implementación de un juez online que permita el desarrollo de competencias y simulacros tipo maratón, para ser utilizado en los entrenamientos del grupo de maratones de programación de la Universidad ECCI.

3.2 Objetivos Específicos

- Realizar un estudio que permita seleccionar una herramienta computacional de código abierto que funcione como juez de maratones de programación, con el fin de determinar cuál se puede adaptar a las necesidades tecnológicas de la Universidad ECCI.
- Realizar el diseño de la infraestructura que permitan administrar y gestionar los recursos tecnológicos que requiere el desarrollo de entrenamientos o maratones de programación.
- Elaborar el manual de instalación del juez en línea, así como los manuales de uso para el competidor, administrador, juez y público en general.
- Implementar una prueba piloto sobre la herramienta computacional seleccionada haciendo uso de una infraestructura de hardware similar a la de un maratón de programación con el fin de comprobar su funcionalidad.

Capítulo 4. Justificación y Delimitación de la Investigación

4.1 Justificación

Con la aplicación de un software en cuestión, los estudiantes pertenecientes al grupo de maratones de programación ya no tendrán que recurrir a plataformas de otras instituciones, ya que les permitirá conectarse al software para interactuar, hacer competencias de programación y compartir conocimiento con sus propios compañeros.

Con el presente trabajo no sólo se obtendrá más participación en el grupo de maratones de programación, sino que los estudiantes podrán aplicar sus conocimientos matemáticos, de algoritmia y programación. De igual forma el programa de Ingeniería de Sistemas, así como la Universidad ECCI se potencializará, proyectándolo así a que sea reconocida por el uso de un juez en línea que pocas Universidades disponen.

Empresas reconocidas como Google, Microsoft, Yahoo entre otros, se enfocan en atraer estudiantes que vayan a maratones de programación, debido a que genera un desarrollo de lógica, e implementación de algoritmos matemáticos que ningún otro programador puede igualar sin haber participado en este proceso de aprendizaje, lo cual abre las puertas para que los estudiantes obtengan trabajos muy valorados y con grandes ingresos (Quora, Forbes.). La universidad javeriana es un claro ejemplo de éxito con la implementación de una herramienta que permite calificar algoritmos, ya que facilitó en su momento, se destacara sobre otras universidades convirtiéndola en una de las mejores en este campo.

Este aplicativo no solo estará a disposición de la Facultad de Ingeniería de Sistemas de la Universidad ECCI, si no que podrá facilitarse a cualquiera de los estudiantes para que puedan

utilizar las herramientas que el aplicativo ofrece, generando en un futuro un buen nivel cognitivo en el desarrollo de análisis algorítmico y de resolución de problemas.

4.2 Delimitación

Para realizar el presente proyecto de grado, se tiene previsto un lapso de seis meses, en el cual se realizará un estudio para determinar el mejor juez online Open Source y de esta forma diseñar la mejor infraestructura de software y hardware para la adaptación de esta plataforma.

Solo se escogerá una topología de red por lo cual, dependiendo de la elegida, se llevará a cabo el manual de instalación y la simulación pertinente. Este proyecto de grado solo se encargará de realizar el proceso de investigación y prueba, mas no la plena implantación de la herramienta.

Capítulo 5. Marco de Referencia de la Investigación

5.1 Marco Teórico

5.1.1 Plataformas de entrenamiento

Una plataforma de entrenamiento es un recurso reciente para la educación, que le apuesta a un aprendizaje especializado y de calidad, convirtiéndose en una experiencia virtual de crecimiento profesional tanto para las empresas como para los empleados. (empresa, n.d.). Al ser una herramienta virtual, puede ser accedida en cualquier lugar del mundo, desde cualquier dispositivo que tenga acceso a internet.

Se puede desglosar diferentes tipos de herramientas que abarcan temas específicos, uno de ellos son de tipo contests. Estos son muy reconocidos en el ámbito de las maratones de programación ya que aquí es donde los participantes se inscriben para realizar competencias. Dependiendo del tipo de concurso escogido, se tendrán diferentes normas, evaluadores, e incluso puede cambiar la cantidad de personas para conformar un grupo.

Estos concursos se pueden clasificar en real-Time Contests y Online Contests, aunque los dos poseen reglas predefinidas, los Real-Time Contest suelen ser más estrictos ya que requieren la presencia física del equipo de maratonistas en un lugar ya predefinido (Becerra, 2012), mientras los Online Contests, solo requiere de tener una conexión a internet y un usuario en un navegador para participar.

Algunos de los concursos de programación que requieren a los competidores en un sitio predefinido son:

- ACM/ICPC (Association for Computing Machinery/International Collegiate Programming Contest): es un concurso de programación algorítmica para estudiantes

universitarios. Los equipos son conformados por tres estudiantes, que representan a su universidad, trabajando en conjunto para resolver los problemas que se les proporcionan, fomentando la colaboración, la creatividad, la innovación y la capacidad de actuar bajo presión. A través del entrenamiento y la competencia, los equipos se desafían unos a otros para elevar la presión y la rapidez con el cual realizan los problemas. En pocas palabras, es el concurso de programación más antiguo, más grande y más prestigioso del mundo.("El Concurso Internacional de Programación Universitaria ICPC," n.d.)

- IPSC (Internet Problem Solving Contest): El IPSC es un concurso en línea de resolución de problemas para equipos de hasta tres personas. Varios problemas son publicados al comienzo de la competencia. Cada problema consiste en una descripción del mismo y algunos archivos de entrada. Para resolver un problema, se debe escribir un algoritmo que dé solución al ejercicio. Por lo general, esto significa que se debe escribir un programa que resuelva el problema, pero se puede entregar el resultado con puño y letra o de cualquier otra forma. Solo se evaluará la corrección de los datos de salida("Reglas IPSC 2018 - IPSC," n.d.)

- ACSL (American Computer Science League): ACSL organiza concursos de informática y concursos de programación informática para estudiantes de escuela primaria, secundaria y preparatoria. ACSL está en la lista de actividades aprobadas de la Asociación Nacional de Directores de Escuelas Secundarias (NASSP). ACSL es también un miembro institucional de la Asociación de Maestros de Ciencias de la Computación.("Concurso de computadora del concurso de programación ACSL," n.d.)

Dependiendo del concurso, existen herramientas computacionales que permiten desarrollarlos; conocidos como Judge contest y consiste en un sistema que recibe el código

fuente enviado por el maratonista como solución a un problema de la competencia, para compilarlo y ejecutarlo con los casos de prueba pertinentes. Entre los online Judge disponibles se encuentran:

- UVa Online Judge: es un juez automatizado en línea para problemas de programación organizado por la Universidad de Valladolid. Su archivo de problemas tiene más de 4300 problemas y el registro de usuarios está abierto a todos. Actualmente hay más de 100000 usuarios registrados. Un usuario puede enviar una solución en C ++ (C ++ 98), Pascal, Java, C ++ o Python.("UVa Online Judge," 2018)
- Timus Online Judge: Timus Online Judge es el mayor archivo ruso de problemas de programación con sistema de evaluación automática. Los problemas se recopilan principalmente en los concursos celebrados en la Universidad Federal de Ural, los Campeonatos de los Urales, los Concursos Subregionales Ural ACM/ICPC y los Campos de Entrenamiento de Petrozavodsk.("Timus Online Judge," n.d.)
- Google Code Jam: Code Jam llama a los programadores de todo el mundo a poner a prueba sus habilidades resolviendo múltiples rondas de acertijos algorítmicos. Las rondas en línea concluyen en la final mundial. Además de ser desafiante y divertido, los problemas de Code Jam pueden ayudar a desarrollar las habilidades de codificación y programación ("Acerca de Google Code Jam," n.d.). Es pertinente mencionar que las rondas que se realizan antes de la final mundial son del tipo Online Contest, pero una vez un grupo clasifica al mundial, los equipos son reunidos en un lugar en específico del mundo para realizar la ronda final. Google paga el viaje, los viáticos y el hospedaje.

5.1.2 Tipos de respuesta según el desempeño del código del participante:

El programa de juzgamiento recibe, compila el algoritmo enviado por el participante, para ingresarle casos de prueba y comparar las salidas obtenidas por las esperadas. Estos programas son capaces de enviar diferentes tipos de respuestas, que permiten conocer si el algoritmo implementado funciona o no. Las respuestas que pueden aparecer son:

- **Correct**: Se resolvió el problema, por lo tanto, el programa cumple con todas las especificaciones.

- **Presentation Error**: Los errores de presentación ocurren cuando el programa produce un formato de salida incorrecto.

- **Runtime Error/Run-Error**: Este error indica que el programa realiza una operación ilegal cuando se ejecuta en la entrada de los jueces. Algunas operaciones ilegales incluyen acceso no valido a referencias de memoria (fuera de un límite de matriz). También hay una serie de errores matemáticos comunes, tales como dividir por cero o desbordamiento.

- **Time Limit Exceded/Time Limit**: En un concurso, el juez tiene un límite de tiempo específico para cada problema. Cuando el programa no finaliza en ese límite de tiempo especificado, se obtiene este error. Es posible que se esté utilizando un algoritmo ineficiente, por ejemplo, tratando de encontrar la factorial de un gran número recursivamente, o tal vez el programa tiene un bucle infinito.

- **Incorrect Output/Wrong answer**: Se obtiene cuando el resultado del programa no coincide con la solución real. En general, (salida incorrecta) ocurre porque se ha entendido mal el problema, no se verificó las condiciones extremas o simplemente no se

tiene la experiencia suficiente para resolverlo. Los problemas a menudo contienen trucos que se pierden al no leer el enunciado con cuidado.

- **No Output:** El programa no produce una salida. Generalmente esto ocurre debido a una mala interpretación del formato de entrada o archivo. Por ejemplo, puede haber una confusión en el nombre del archivo de entrada, el juez está dando entrada desde "a.in", pero el programa está leyendo entrada de "b.in.".

Las herramientas computacionales que son mencionadas en la sección “5.1.1 Plataformas de entrenamiento” son una muestra del software existente para maratones de programación. Debido a la variedad de concursos de programación que se desarrollan en diferentes lugares del mundo, el idioma suele ser siempre en inglés, por lo que esto representa un obstáculo a la hora de utilizarlos (Campos, Ferreira, & Anido, 2010). Algunos de los ejemplos más notorios están en la universidad de Stanford con su grupo de estudio Stanford Local Program Contest (SLPC) donde se realizan regularmente maratones de preparación para sus estudiantes las cuales pueden ser accedidas también por cualquier persona externa.

Estas plataformas pueden ser utilizadas para entrenamiento o para la competencia, su uso está limitado a las condiciones de los administradores. En el caso de los juzgadores online, estos ya están por defecto, pero en algunos otros, se pueden personalizar para que tengan características específicas, pero siempre con el objetivo de evaluar los problemas computacionales.

En Colombia, las universidades que suelen participar en maratones de programación, tienden a usar algún juzgador online de otro país para realizar prácticas y calificar los

ejercicios realizados por ellos, por lo tanto, aun no se ha visto desarrollado e implementado esta herramienta en pro a necesidades específicas de los maratonistas.(Becerra, 2012).

Estas herramientas no solo ayudan a la creación, compresión, utilización de estructuras de datos y capacidad de razonamiento lógico, sino que permite trabajar en equipo, así como innovar y crear nuevos programas poniendo en práctica el desarrollo de algoritmos bajo presión por lo que promueve investigación por parte de los estudiantes de temas no cubiertos en clase y la capacidad de resolver estos problemas sin el instructor a cargo.(Becerra, 2012).

Al analizar estudios preliminares en los cuales se implanto algún tipo de software se pudo ver resultados muy favorables, la utilización de un ambiente virtual competitivo dentro de un salón de clase en cursos relacionados a la programación puede generar un alto nivel de motivación por parte de los estudiantes lo cual generalmente se traduce en un mejor desempeño académico (Moreno, Pineda, & Montoya, 2013). No solo se vio un mejor índice de aprendizaje, sino que esto llevo algunos grupos como el de la Universidad Francisco de Paula Santander- Cúcuta (Colombia) a participar en maratones nacionales, quedando en los primeros puestos y verificando el nivel de éxito que puede tener el uso de este software. Estos cambios se han hecho en respuesta del avance de las tecnologías informáticas para definir estudiantes centrados en un curso el cual provee interactividad sin restricciones de tiempo y de ubicación(Gómez et al., 2014).

El escoger un software libre acarrea muchos elementos con él, así como definiciones que deben quedar claros antes de continuar, uno de estos es el tema de las licencias, las cuales pueden dar información de las partes que se puede descargar, modificar y distribuir.

Algunas licencias mencionadas a continuación se originaron del término GNU (GNU's Not Unix), el cual es un proyecto creado en septiembre de 1983 como un sistema operativo libre que tiene un diseño tipo UNIX, sin embargo, el código que contiene es distinto al anterior. Su creador Richard Stallman específico la definición de "Software Libre" así que establece la Fundación para el software libre (Free Software Foundation - FSF), saliendo en el año 1989 al mercado, la primera versión de la Licencia Pública General (General Public License - GPL).

Ponerlo a disposición de dominio público, es la mejor forma de presentar un software como libre, brindándole la oportunidad a cualquier persona de descargar el código fuente y declararse propietario del mismo. Esto conlleva a que una persona "X" pueda vender a una persona "Y" el software de su "propiedad", por lo tanto, esta no tendrá la misma libertad que el software tenía inicialmente.

Según GNU, Operations System Copyleft es un método para hacer un programa o cualquier trabajo libre. Sin embargo, no evita que un usuario que descargue o modifique el código (programa) pueda ponerle precio. Así mismo se encarga de que cualquier persona que obtenga el software, con o sin cambios, tenga la libertar de cambiarlo o realizar más copias, garantizando la libertad a todos los usuarios.

Ahora debido a que el termino software libre tenía una connotación ideológica bastante fuerte, se creó el término "Código abierto", el cual en gran parte significa lo mismo "software libre", pero busca ser más atractivo hacia las compañías que apoyan este movimiento. Después de ello se crea el termino OSI (Open Source Initiative) el cual se desarrolló una

estrategia para proveer y respaldar las iniciativas empresariales utilizando un modelo de desarrollo nuevo y peculiar.

La OSI tiene los siguientes criterios para que un software sea llamado código abierto:

1. Redistribución gratuita: La licencia de un software no requerirá una regalía o alguna otra tarifa para obtenerla.

2. Código fuente: El programa debe permitir la distribución del código fuente, y el método para obtenerlo debe preferiblemente ser descargado del internet sin ningún costo.

3. Trabajos derivados: Debe permitir la modificación y realizar software derivados a este, los cuales se distribuyan bajo los mismos términos que la licencia del software original.

4. Integridad del código fuente del autor: La licencia puede restringir la distribución del software con modificaciones a no ser que esta permita distribuirla con algunos “archivos parches”, de igual forma puede requerir que los trabajos derivados a este sean mencionados con un número de versión diferente al software original.

5. No discriminación contra personas o grupos: La licencia no debe discriminar a ninguna persona o grupo de personas.

6. No discriminación contra campos de esfuerzo: No debe restringirse el uso del programa a ningún campo de estudio en específico, es decir, que no impida que el programa se utilice en un negocio o para cualquier investigación que se vaya a realizar.

7. Distribución de licencia: Los derechos que posea el programa deben aplicarse a todos aquellos que los redistribuyan sin tener la necesidad de que se agreguen una licencia opcional.

8. La licencia no debe ser específica para un producto: Si el programa se extrae de una distribución en específico, los términos de la licencia se distribuyen con él, es decir, que a quien se le haya redistribuido el programa se le otorga los mismos derechos del software original.

9. La licencia no debe restringir otro software: Define que los demás programas distribuidos en el software libre no deban ser necesariamente software de código abierto.

10. La licencia debe ser tecnología-neutral: Ninguna disposición de la licencia debe basarse en imponer preferencias a favor o en contra de una determinada tecnología.

A continuación, se menciona algunas de las licencias de software libre y de código abierto más importantes de la actualidad:

- **GNU GPL v2** (“gnu.org,” n.d.-a)**.** Lanzada en junio de 1991, es una versión previa de GPL de GNU, que posee Copyleft. Es una de las más utilizadas como licencia de código abierto. Entonces un licenciatario de software puede:

    - Copiar y distribuir el código fuente no modificado del programa
    - Modificar el código fuente del programa y distribuirlo

- Distribuir versiones compiladas del programa, tanto modificadas como no modificadas
- Todas las copias modificadas o no, deben llevar un aviso de copyright y la exclusión de la garantía
- Todas las copias modificas se deben distribuir bajo la GPL v2
- Todas las versiones compiladas del programa deben ir acompañadas del código fuente, o una vía la cual se pueda acceder.

Por lo tanto, esta licencia solo asegura que todas las versiones modificadas del código permanezcan libres y de código abierto.

- **GNU GPL v3**("gnu.org," n.d.-b) Lanzada el 29 de junio de 2007, se ofrece como una alternativa a la GPL v2, en lugar de un reemplazo. Debido a que en la GPL v2 se distribuía el código fuente de forma cifrada, los usuarios no podían obtener por completo el código, ya que necesitaban una llave para la desencriptación la cual poseía solo la persona o empresa "propietaria" del software, por lo tanto, en la GPL v3 se hizo obligatorio que cualquier persona que distribuyera el software con esta licencia, debía pasar la información adicional, así como las claves necesarias para modificar y ejecutar las copias del software. También permitió que la demanda por litigios de patentes de software fuera menos atractiva, esto debido a que, si una persona cobraba el uso de un software con licencia GPL v2, y otra, además, tenía la patente de esta, debía dejar de distribuirla por completo.

  Así mismo se solucionaron problemas de incompatibilidades con otras licencias de código abierto y libre, como el lenguaje legal ambiguo y aclaración de responsabilidades en torno a la distribución del código fuente.

Por lo tanto, GPL v3 asegura que las versiones modificadas del código permanezcan libres y de código abierto e intenta difundir el copyleftismo (Copyleft).

- **Lesser General Public License 2.0 (LGPLv2)**("GNU General Public License v2.0 only | Software Package Data Exchange (SPDX)," n.d.) Publicado en el año 1991 como una versión suavizada de la GPL, está pensado para librerías y establece que cualquier modificación a esta también debe ser licenciada como LGPL, pero no impone ningún tipo de licencia a aplicaciones que usen la librería. Este permite enlazar librerías o módulos privados, por lo cual no cumple de manera fiel el término de Copyleft. Una característica importante es que se puede convertir cualquier código LGPL a GPL, para que este se no sea utilizado de forma privativa (La Versión actual de LGPL es la 3).

- **Artistic License**("Artistic Licenses | Open Source Initiative," n.d.) Esta Licencia es utilizada para ciertos paquetes de código abierto o de software libre, se usa casi siempre en proyectos basados en el lenguaje de programación Perl y se caracteriza por no utilizar Copyleft. La versión 1.0 tenían muchas brechas jurídicas y era demasiado vaga, por lo que se creó una versión dos la cual fue aprobada como una licencia de software libre. Es compatible con otras licencias como GPL por su cláusula de relicenciación.

- **Apache License 2.0**("Apache License, Version 2.0," n.d.) Es una licencia de código abierto lanzada por la Apache Software Foundation (ASF), permite usar, modificar y distribuir libremente cualquier producto con esta licencia, con la única condición de

informar a los usuarios receptores que el software liberado contiene un código con la licencia de apache.

- **MIT License**(“The MIT License | Open Source Initiative,” n.d.) Es una licencia de software libre la cual garantiza al usuario final los derechos de copiar, modificar, fusionar, distribuir libremente cualquier software (sin Copyleft). No contiene cláusulas de publicidad, así como tampoco está sujeto a derechos de autor por lo que es apropiado para que los desarrolladores realicen cambios del software como mejor les parezca, sin la necesidad de liberar al público los cambios realizados al programa original. Es también conocido con el nombre X11.

- **BSD Licenses**(“BSD license definition,” n.d.) La licencia BSD (Berkeley Software Distribution) es una licencia desarrollada inicialmente en 1998 para sistemas BSD. Esta clase licencias suelen ser de carácter permisiva, es decir, se puede hacer uso del código abierto con mucha más libertad en relación a proyectos licenciados bajo GPL.

- **Q Public License v1.0** (“The Q Public License Version (QPL-1.0) | Open Source Initiative,” n.d.) La Q public license 1.0 es una licencia sin copyleft desarrollada por la compañía Trolltech en 1999. Dicha compañía diseño esta licencia a fin de regular la distribución y modificaciones del conocido software Qt, que es la herramienta para el desarrollo de interfaces gráficas de usuario generalmente usadas en sistemas UNIX, el entorno más conocido desarrollado con esta herramienta es el KDE usado en diferentes distribuciones Linux. Una de las características más destacadas de esta licencia es que permite la distribución de las modificaciones del software original siempre y cuando se publiquen como parches del mismo.

5.2 Marco Conceptual

Una maratón de programación es básicamente un evento en el que grupos de estudiantes o personas con conocimientos en algoritmia y programación compiten desarrollando soluciones a problemas de variados enfoques mediante el empleo de algoritmos, modelos matemáticos, estadísticos, razonamiento y lógica en resolución de problemas. Hay infinidad de competencias en todo el mundo y todas ellas emplean diferentes metodologías y Jueces en Línea para evaluar los problemas.

Los Jueces en Línea son una herramienta que permite la evaluación de problemas de programación aceptando código de diferentes lenguajes, ejecutando dicho código y comparando la respuesta que genera con la salida que el aplicativo tiene por respuesta. La mayoría de estos jueces en línea son de libre acceso y cuentan con servicios base como la evaluación de problemas a manera de entrenamiento, o, como jueces de maratones de programación. Los problemas de entrenamiento suelen estar categorizados por el nivel de dificultad o temática específica, estas plataformas son muy versátiles y útiles en el aprendizaje de algoritmia y resolución ágil de problemas. (“The ICPC International Collegiate Programming Contest,” n.d.)

Un ejemplo del empleo de empleo de los jueces en línea se presenta en la Maratón Nacional de Programación organizada por la Organización de Maratones de Programación Colombiana (OMPC) que es el consorcio conformado por la Asociación Colombiana de Ingenieros de Sistemas y la Red de Decanos y Directores de Ingeniería de Sistemas, a fin de promover y llevar a cabo competencias y eventos de programación en Colombia. Estos

también se encargan de seleccionar los equipos que destacaron en la Maratón Nacional y les da la oportunidad de competir en la Maratón Regional Latinoamericana ACM/ICPC.(“Maratón de Programación | ACIS,” n.d.)

5.3 Marco Legal

Para el desarrollo de este proyecto se va a utilizar software libre. El término Software libre se enfoca en las libertades que se le otorga al usuario. La organización Free Software Foundation (FSF) representa el software libre. Según la FSF para que determinado software sea considerado como software libre, debe cumplir con las siguientes libertades (“gnu.org,” 2018):

- **libertad 0:** La libertad de ejecutar el programa para cualquier propósito.
- **libertad 1:** La libertad de estudiar cómo funciona el programa, y cambiarlo para que haga lo que usted quiera. El acceso al código fuente es una condición necesaria para ello.
- **libertad 2:** La libertad de redistribuir copias para ayudar a su prójimo.
- **libertad 3:** La libertad de distribuir copias de sus versiones modificadas a terceros. Esto le permite ofrecer a toda la comunidad la oportunidad de beneficiarse de las modificaciones. El acceso al código fuente es una condición necesaria para ello.

Es muy importante tener en cuenta estas libertades debido a que se va a usar software libre. Adicional a las libertades expuestas, el concepto copyleft es bastante utilizado en el mundo del software libre. Básicamente, el copyleft es un método general para hacer un programa (u otro tipo de trabajo) libre, exigiendo que todas las versiones modificadas y extendidas del mismo sean también libres (“gnu.org,” 2018), por lo cual se debe tener claro que sin importar las modificaciones que se le haga al software, debe seguir siendo libre.

Para el desarrollo del proyecto también se debe tener en cuenta ciertas leyes del ámbito nacional.

- **Ley 603 de 2000:** Esta ley se refiere a la protección de los derechos de autor en Colombia. Recuerde: el software es un activo, además está protegido por el Derecho de Autor y la Ley 603 de 2000 obliga a las empresas a declarar si los problemas de software son o no legales. («ley 603 de 2000 en Colombia», 2000)
- **Ley 44 de 1993:** Especifica penas entre dos y cinco años de cárcel, así como el pago de indemnizaciones por daños y perjuicios a quienes comentan el delito de piratería de software. (*ley44_1993.pdf*, n.d.)

Existen algunos softwares propietarios similares a lo que se quiere acceder en este proyecto y no se debe pasar por alto, ya que si se intenta usar alguno de ellos sin el respectivo permiso de sus propietarios se estaría cometiendo un delito que se paga con cárcel.

- **Ley 1273 de 2009:** Por medio de la cual se modifica el Código Penal, se crea un nuevo bien jurídico tutelado - denominado "de la protección de la información y de los datos"- y se preservan integralmente los sistemas que utilicen las tecnologías de la información y las comunicaciones, entre otras disposiciones. («Ley_1273_2009.pdf», 2009)
- **Ley estatutaria 1581 de 2012:** Entró en vigencia la Ley 1581 del 17 de octubre 2012 de PROTECCIÓN DE DATOS PERSONALES, sancionada siguiendo los lineamientos establecidos por el Congreso de la República y la Sentencia C-748 de 2011 de la Corte Constitucional. («LEY_1581_DE_2012.pdf», 2012)

Es necesario que los usuarios de la aplicación ingresen datos personales al momento de crear las cuentas, por lo que se debe garantizar la protección de la información y de los datos de los usuarios.

5.4 Marco Histórico.

La ICPC fue creada en la universidad A&M de Texas por pioneros de “Alpha Chapter” de la “UPE (Upsilon Pi Epsilon) Computer Science Honor Society” en el año 1970. Comenzó como torneos internos de universidad, permitiendo que para el año 1977 se extendiera y se convirtieran en una competencia con varias rondas clasificatorias posibilitando así la participación de equipos de Estados Unidos y Canadá además de que en este año la ACM (Computer Science Conference) ayudo a la organización de la competición final. A partir del año 1989 se empezaron a incluir otras universidades a este tipo de competencias debido a que era un programa innovador, que le permitía a los competidores, aumentar la capacidad de resolución de problemas haciendo uso de algoritmos implementados en computadores, además que concedía oportunidades de trabajo a los mejores equipos en las diferentes áreas de la computación. Desde 1977 el principal patrocinador de esta competencia es IBM (International Business Machines Corporation) permitiendo así que pasaran de 840 equipos de 560 universidades en esta fecha, a 46381 estudiantes de 2948 universidades en 2016. Anualmente el número de equipos inscritos aumenta entre 10% y un 20%*(League 2019, n.d.)*.

Anexo 1. ICPC FACT SHEET

El grupo de maratones de programación de la universidad ECCI inicialmente fue creado por el profesor Simar Enrique Herrera Jiménez en el año 2014, como y cito textualmente “se creó como respuesta a procesos de acreditación, y por otra parte por la necesidad de tener participación en el circuito REDIS y en las competencias nacional, regional y mundial si se podía”.

Capítulo 6. Tipo de Investigación

A continuación, se presenta las actividades que se realizarán con su respectivo tipo de investigación.

Tabla 1. Tipos de Investigación del Proyecto.

Tipo de Investigación	Actividades
Documental	-Levantamiento de información (Juzgadores Open Source) -Modelo Entidad Relación
Correlacional	-Selección de la herramienta
Aplicada	-Diseño de la topología de red. -Diseño de casos de uso. -Diseño de diagramas de secuencia. -Diseño de diagrama de despliegue. -Instalación del aplicativo. -Desarrollo de Manuales de administrador, juez, equipos, instalación, espectadores y organizador un evento -Prueba piloto.

Fuente: Los Autores. 2019.

Capítulo 7. Diseño Metodológico

7.1 Levantamiento de información (Juzgadores Open Source)

El tipo de investigación que se acomoda a esta actividad es de carácter documental, debido a que se tiene que recuperar de fuentes externas, documentos que permitan conocer que juzgadores Open Source tienen la capacidad de realizar una maratón de programación.

Para definir el juez online que será la base para realizar el estudio de software y hardware del proyecto (En la Tabla 2. Características del juzgador DOMJudge, se puede observar cómo se resume en una tabla la información que será presentada más adelante, esto se realizó con todos los juzgadores preseleccionados), se tuvieron en cuenta algunos criterios que serán descritos a continuación:

- Juzgamiento de algoritmos implementados en C, C++, Java y Python: estos lenguajes fueron escogidos, debido a que son utilizados en las maratones de circuito, nacionales, internacionales, regionales y mundiales.
- Licencia: al momento de implementar el juez en línea, primero se deben tener claro cada una de las características legales que lo rodean, permitiendo conocer qué juzgadores permiten modificaciones, la distribución y el uso.
- Cumplir con reglamentación de la ACM/ICPC: ya que no solo se busca complementar el desarrollo de habilidades de lógica en programación y algoritmia, la selección de este criterio busca un juez que brinde al usuario, una simulación de las competencias de programación que actualmente están disponibles, promoviendo así el uso de algoritmos eficientes y permitiéndoles interactuar con un software que cumpla las normas predefinidas por una competencia, además de permitirles familiarizarse con el entorno web que se presente el día de la maratón.

- Actualizaciones: debido a que el juzgador debe implementar las reglas de ACM/ICPC, cualquier cambio que esta organización realice, debe verse reflejado en la herramienta, para mantener a los competidores preparados ante los maratones de programación. Añadiendo igualmente que los softwares no son perfectos, y estos presentan bugs que pueden afectar de alguna manera una competencia, por lo que las actualizaciones permiten mantener el sistema siempre al margen, manteniendo la integridad de sus funciones.
- Diferencias entre perfiles de usuario: es necesario establecer permisos e igualmente escoger una herramienta que permita diferenciar los usuarios, tales como; competidores, jueces y administradores de la herramienta, por lo que este punto es importante al momento de escoger un juez.

Tabla 2. Características del juzgador DOMJudge.

Nombre Juzgador	DOMJudge
Idioma	Ingles
Lenguaje de implementación	• PHP • C++ • C • Bash
Juzga algoritmos implementados en	• Java • Python 2/3 • C • C++ • Kotlin • Ruby • Ada • Fortran 95 • Lua • Scala • AWK • Bash shell • C# • Haskell • JavaScript • Pascal • Perl • POSIX shell • Prolog
Licencia	• GNU/GPL • MIT License • BSD License

Link del juzgador	https://www.domjudge.org/download
Años	2004 Registrado – Presente
Versión	Actual 7.0.1
Eventos importantes en los que se usaron	• ICPC Sub-regionals: ○ BAPC (since 2004) ○ GCPC (since 2010), ○ UKIEPC(since 2015) • ICPC Regionals: ○ NWERC (since 2007) ○ SWERC (since 2008) ○ SER USA (since 2010) ○ SOCAL (since 2013) ○ RMRC (since 2014) ○ South Pacific Regional • ICPC World Finals: ○ (since 2012, in 2018 as main system)
Ventajas	**Desventajas**
• Reglamentado por ACM-ICPC • Aplicativo Web • Interfaz intuitiva • Juzgamiento automático • Diferentes perfiles de usuario • Compatibilidad de ejecución en sistemas Linux	

Fuente: Los Autores. 2019.

Las demás tablas de las características de los juzgadores investigados, pueden ser encontrados en el siguiente anexo.

Anexo 2. Comparación Juzgadores

7.2 Selección de la herramienta

Una vez realizada la investigación de los juzgadores Open Source, se debe concretar que juzgador se deberá utilizar, esto se hará, haciendo uso de una lista de características que se definirán más adelante, por lo tanto, el tipo de investigación será de carácter correlacional.

Figura 1. Tabla comparativa de juzgadores

Juez	Juzga algoritmos implementados en: Java, C++, Python (2 Unidades)	Licencia para modificación (2 Unidades)	Cumple con reglamentación ACM-ICPC (2 Unidades)	Actualizaciones vigentes (2 Unidades)	Diferentes perfiles de usuario (2 Unidades)	Puntuación final
Injuslin		✓				2
Celiz Judge System		✓				2
Mooshak	✓	✓			✓	6
Code Cracker		✓			✓	4
Taskeval		✓			✓	4
DevalRun		✓				2
DOMJudge	✓	✓	✓	✓	✓	10
OKO Online Judge		✓				2
BOCA Online Contest	✓	✓	✓		✓	8

Fuente: Los Autores. 2019.

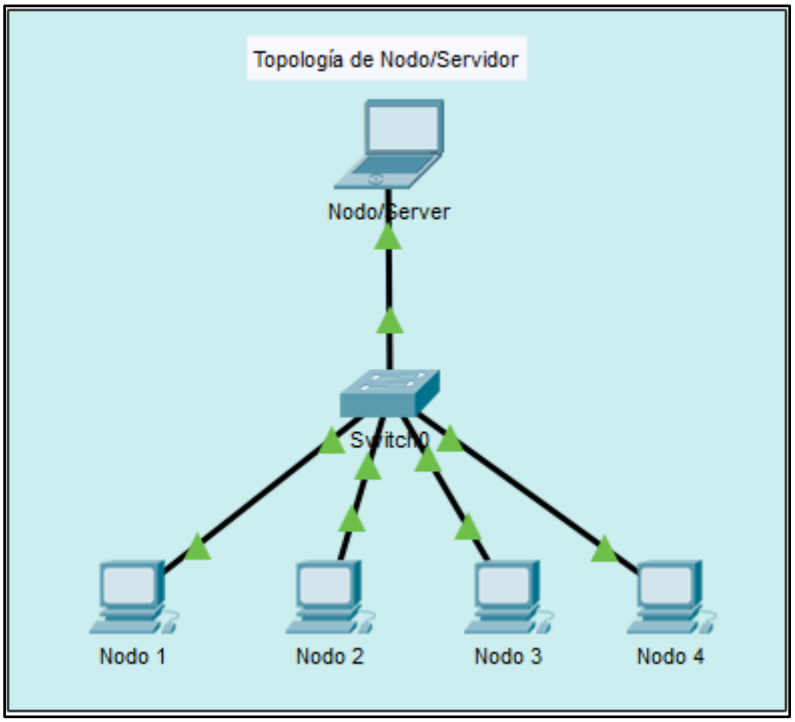
Como se puede observar en la “Figura 1. Tabla comparativa de juzgadores” y con los criterios anteriormente expuestos, el juzgador que mejor se adapta con las condiciones del proyecto, es DomJudge. En la tabla “Tabla 2. Características de DOMJudge”, se puede destacar que este juzgador cumple con creces el ítem “Juzgamiento de algoritmos implementados en C, C++, Java y Python” ya que no solo permite la evaluación de los lenguajes anteriormente mencionados, sino que además dispone de otros 15 lenguajes para juzgamiento. El juzgador permite la re-distribución y/o la modificación todo bajo los términos de la GNU (General Public License). Uno de los criterios en los cuales se diferenció DOMJudge sobre los demás, son las actualizaciones que se presentan, inclusive una vez realizado el proceso de instalación y configuración sobre la versión (6.0.2), se pudo observar que se actualizo a la versión (7.0.1). Cabe mencionar que la herramienta no exige nuevas actualizaciones para su funcionando, por lo que es un nuevo punto a favor, ya que si el usuario final no observa ningún problema con la versión que tiene en cuestión, perfectamente la puede seguir utilizando sin conllevar riesgo de fallos (si el sistema recibe una actualización de seguridad, se recomienda actualizar).

Se observa igualmente que el juez ha sido utilizado en sub-regionales, regionales y para el mundial de programación de la ICPC, por lo que satisface a grandes rasgos el ítem "reglamentación de la ACM/ICPC".

7.3 Diseño de la topología de red

Una vez se tenga conocimiento del juzgador que se empleará, se dispondrá a consultar la documentación para conocer los diferentes tipos de conexión que la herramienta permite, para luego realizar la topología de la red correspondiente (investigación aplicada).

Figura 2. Topología de Nodo/Servidor



Fuente: Los Autores. 2019.

Esta topología de red muestra la posibilidad de instalar el aplicativo en un nodo (computador), el cual permitirá únicamente la conexión de los dispositivos dentro de la red LAN en la que esté conectado el nodo/server. En este nodo se instalará tanto el DOMServer como el Judgehost. Se escogió este tipo de topología, debido a que es la más fácil de implementar en términos de tiempo, dinero y hardware.

Las demás topologías a las cuales se le realizaron investigación, pueden ser encontradas en el siguiente anexo.

Anexo 3. Modelado Software Hardware

7.4 Diseño de diagramas de casos de uso

Para el diseño de los casos de uso, se tendrá como base la documentación obtenida durante el levantamiento de la información. Esto permite visualizar el sistema desde la perspectiva del usuario posibilitando un mejor entendimiento para el mismo. La siguiente tabla muestra el diagrama de caso de uso para la creación de un contest.

Figura 3. Diagrama caso de uso Crear Contest

<table>
<tr><td colspan="2">Caso de Uso: Crear contest</td></tr>
<tr><td>Actor:</td><td>Administrador</td></tr>
<tr><td>Importancia:</td><td>Alta</td></tr>
<tr><td>Precondiciones:</td><td>- Logueo del administrador
- Agregar equipos
- Habilitar lenguajes de programación</td></tr>
<tr><td>Postcondición</td><td>- La herramienta muestra el contest creado</td></tr>
<tr><td>Flujo Principal</td><td>
<table>
<tr><th>Actor</th><th>Sistema</th></tr>
<tr><td>1. Ingresa a la botón contests.</td><td></td></tr>
<tr><td></td><td>2. Hace búsqueda en la base de datos contest con valores en cero en los campos de inner y post.</td></tr>
<tr><td></td><td>3. Lista los contests encontrados y muestra un icono para la creación de un nuevo External contest.</td></tr>
<tr><td>4. Clickea el icono “add new contest”.</td><td></td></tr>
<tr><td></td><td>5. Se visualiza el formulario de Add External contest.</td></tr>
<tr><td>6. Ingresa los datos correspondientes, guarda formulario.</td><td></td></tr>
<tr><td></td><td>7. Verificación de información ingresada.</td></tr>
<tr><td></td><td>8. Guardado de datos en la base de datos.</td></tr>
<tr><td></td><td>9. Regresa al listado de External contests.</td></tr>
</table>
</td></tr>
<tr><td>Flujos alternativos</td><td>7. Error al momento de validar los datos. – Mostrar mensaje de error.</td></tr>
</table>

Fuente: Los Autores. 2019.

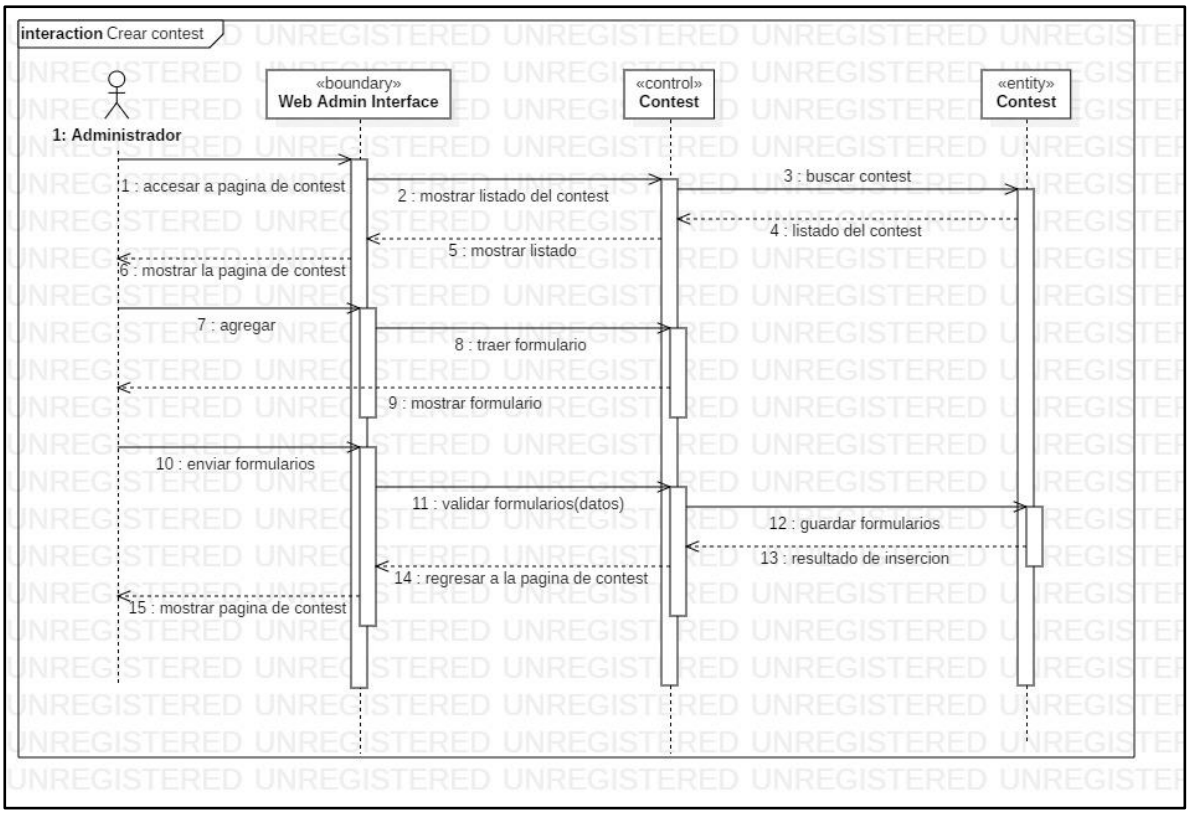
Los demás diagramas de casos de uso, pueden ser consultados en el siguiente anexo.

Anexo 4. Modelado Software Hardware

7.5 Diseño de diagramas de secuencia

Haciendo uso de la información obtenida de la herramienta, se realizarán los diagramas de secuencia para cada uno de los casos de uso, permitiendo modelar las interacciones entre los objetos del sistema. La siguiente figura muestra el diagrama de secuencia para la creación de un contest.

Figura 4. Diagrama de secuencia Crear Contest



Fuente: Los Autores. 2019.

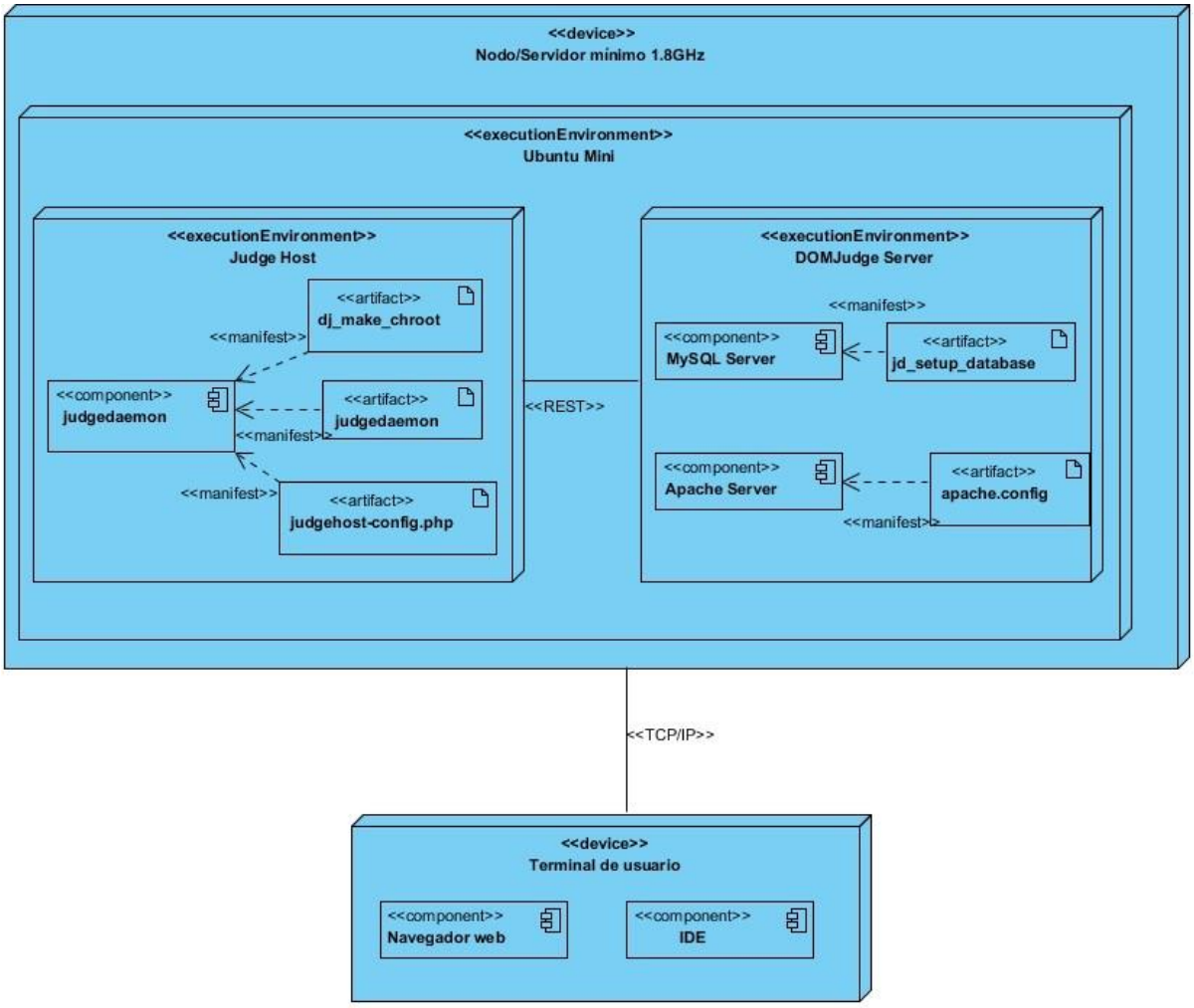
Los demás diagramas de secuencia, pueden ser encontrados en el siguiente anexo.

Anexo 5. Modelado Software Hardware

7.6 Diseño de diagrama de despliegue

El diseño del diagrama de despliegue permitirá conocer los componentes de software y hardware que maneja la herramienta, lo cual permitirá conocer en caso de algún fallo, que parte de esta estructura es la que funciona mal. Se realiza teniendo como base la documentación del juzgador. La siguiente figura muestra el diagrama de secuencia para la creación de un contest.

Figura 5. Diagrama de Despliegue

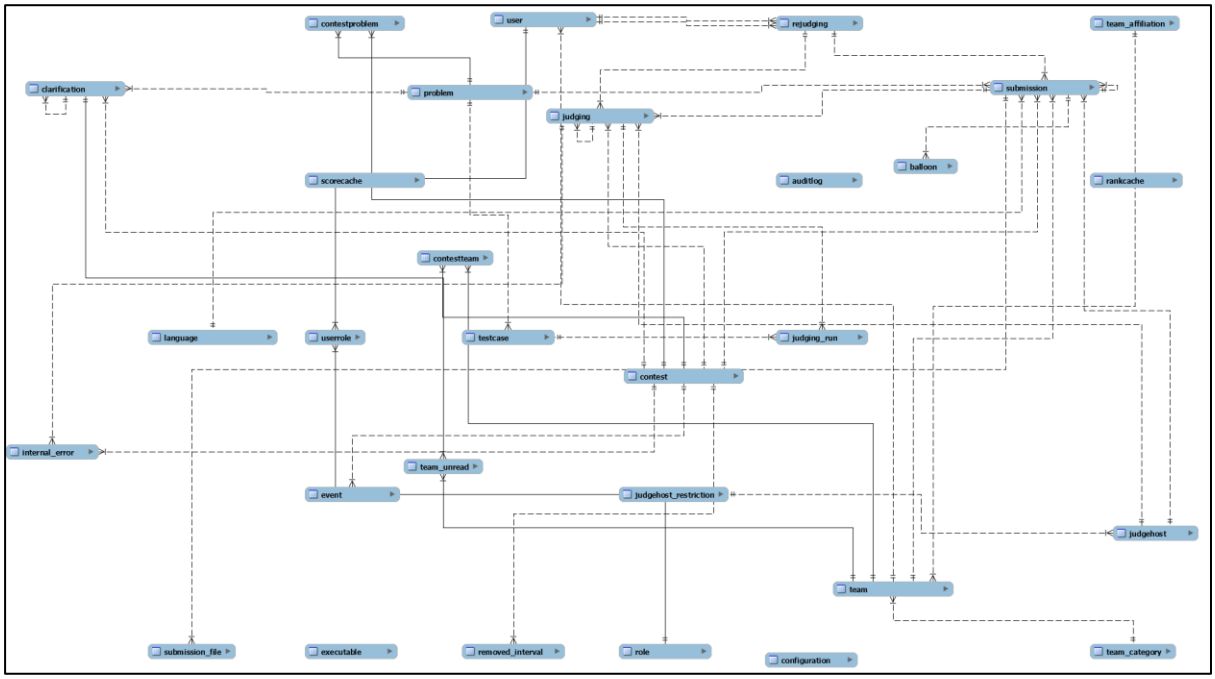


Fuente: Los Autores. 2019.

7.7 Modelo Entidad Relación

Para realizar el diseño del modelo entidad relación de la herramienta escogida, se hará uso de un programa de modelamiento UML estos tipos de diagramas, así mismo de ser posible se buscará en la carpeta raíz del juez el archivo que permite la creación de la base de datos del juzgador, para así tener más claro el diseño que se realizará. La siguiente figura muestra el diagrama entidad relación correspondiente a la base de datos de domjudge que se instala con el sistema de juzgamiento DOMJudge.

Figura 6. Modelo Entidad Relación de DOMJudge



Fuente: Modelado por los autores, Script SQL DOMJudge. 2019.

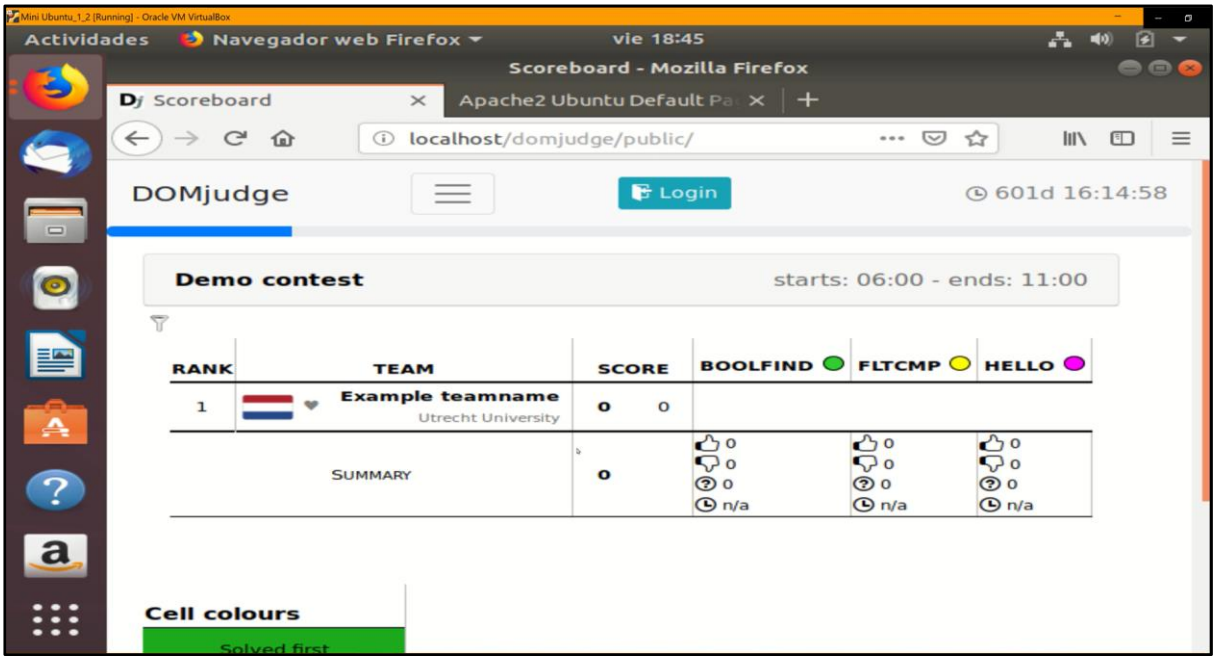
La descripción de cada tabla presentes en el modelo entidad relación de DOMJudge, pueden ser revisadas en el siguiente anexo.

Anexo 6. Modelo Entidad Relación (DOMJudge) v 6.0.2

7.8 Instalación del aplicativo

Se realizará la instalación de la aplicación haciendo uso de una máquina virtual. Durante el proceso de instalación se exportará las veces que sea necesarias el sistema operativo (con y sin la herramienta instalada), debido a que se esperaran que surjan errores durante este proceso de instalación. Esto se hará basándose en la documentación preexistente del aplicativo. La siguiente figura muestra el menú de inicio de la competencia que aparece una vez se instala DOMJudge.

Figura 7. Menú de inicio DOMJudge



Fuente: Los Autores. 2019.

Los pasos de instalación de la aplicación, pueden ser consultada en el siguiente anexo.

Anexo 7. Manual Instalación

7.9 Elaboración de manuales

Una vez instalada la aplicación, se comenzó el proceso de pruebas, lo cual permitió familiarizarse con la herramienta y comprender que elementos se deben configurar para la realización de una maratón de programación.

Los manuales de usuario son los siguientes:

- Manual de instalación: Este documento proporciona un paso a paso para el proceso de instalación de DOMJudge v6.0.2 en una máquina virtual. Puede ser consultado en el Anexo 7. Manual Instalación.
- Manual de administrador: Proporciona una guía para dar inicio a una maratón de programación. Puede ser consultado en el Anexo 8. Manual Administrador.

Anexo 8. Manual Administrador

- Manual de juez: Proporciona una guía que permite monitorear un concurso, además; chequear sumisiones, atender clarificaciones, inspeccionar el estado de los equipos, entre otros. Este manual puede ser consultado en el Anexo 9. Manual de Juez.

Anexo 9. Manual de Juez

- Manual de equipo: Proporciona una guía para entender toda la interfaz web del equipo, así como los pasos a seguir para enviar la solución de un problema. Puede ser consultado en el Anexo 10. Manual Equipo.

Anexo 10. Manual Equipo

- Manual del público: Proporciona una guía completa para entender toda la interfaz web del público. Puede ser consultado en el Anexo 11. Manual Publico.

Anexo 11. Manual Publico

- Consejos para el organizador del evento: Este documento proporciona los elementos a tener en cuenta para la planeación de un maratón de programación. Puede ser consultado en el Anexo 12. Consejos Organizador Del Evento.

Anexo 12. Consejos Organizador Del Evento

7.10 Prueba piloto

Para la ejecución de la prueba piloto, es necesario realizar un proceso de planeación en el cual se establezcan; cantidad de equipos a inscribir, infraestructura de hardware necesario, lenguajes de programación permitidos, además de disponer de una máquina de respaldo que contenga del juzgador en caso de emergencia. La siguiente figura muestra el scoreboard board de la primera Maratón interna de la Universidad ECCI, en el cual participaron estudiantes del SENA, Universidad Distrital, Universidad Konrad Lorentz, Universidad ECCI y finalmente el Politécnico Sur América.

Figura 8. Scoreboard primera Maratón interna Universidad ECCI

Scoreboard 1°ra Maratón interna UECCI

starts: 14:00 - ends: 17:00 (the public scoreboard is frozen since 16:20)

RANK	TEAM	SCORE		3N	ALMOST	CHOCOLATE	FEYNMAN	PIZZA	POLE
1	Dígame ustedes Programador competitivo UECCI	3	161	3	0	1/20	1/89	1/52	0
2	MyBigData Programador competitivo UECCI	1	9	0	0	1/9	0	0	0
3	Los lfdeotas Programador competitivo UECCI	1	33	0	0	1/33	0	0	0
4	No creen en Javemus fiesta Universidad ECCI/Independiente	1	37	0	0	1/37	0	0	0
5	Eccitosos Programador competitivo UECCI	1	54	0	0	1/54	0	1	0
6	Kekistan Army Konrad Lorentz / Universidad Distrital	1	165	0	0	2/145	0	0	0
7	Asistentes SENA / ETITC	0	0	0	0	0	0	0	0
	Berlin Universidad ECCI/Independiente	0	0	0	0	0	0	0	0
	Error404 Universidad Distrital	0	0	0	0	1	0	0	0
	Javaboys SENA	0	0	0	0	4	0	1	0
	Javalistemadre Estudiante ECCI	0	0	0	0	0	0	1	0
	JavastaFreezer Programador competitivo UECCI	0	0	0	0	3	0	0	0
	JSSENA SENA	0	0	0	0	0	0	0	0
	Los Criminalistas Politecnico Sur America	0	0	0	0	0	0	0	0
	Lucario SENA	0	0	0	0	6	0	0	0
	Natos SENA	0	0	0	0	2	0	0	0
	Niupi SENA	0	0	0	0	0	0	0	0
	ProPlus Universidad ECCI	0	0	0	0	0	0	0	0
	Rasta Code Universidad ECCI/Independiente	0	0	0	0	1	0	1	0

Fuente: Los Autores. 2019.

El informe de la prueba piloto puede ser consultada en el Anexo 13. Informe Prueba Piloto.

Anexo 13. Informe Prueba Piloto

Capítulo 8. Fuentes para la Obtención de Información

8.1 Fuentes Primarias

- Ya que durante un tiempo el ex docente de la Universidad ECCI Msc. Esp. Ing. Simar Enrique Herrera Jiménez, fue participante de maratones de programación, además de ser entrenador y creador del grupo de maratones, se puede tener la oportunidad de aclarar el por qué se fundó el grupo de maratones de programación de la universidad ECCI, así como la fecha de creación del mismo.
- Ya que se tiene la oportunidad de pertenecer al grupo de maratones de programación de la Universidad ECCI, y se ha logrado participar en circuitos de maratones programación, así como en la maratón nacional del 2017 y 2018, nos permite contar con el conocimiento necesario para la realización de estos eventos, así mismo, se tiene conocimiento directo de algunos aplicativos de juzgamiento y los beneficios que representan.

8.2 Fuentes Secundarias

Las fuentes secundarias de las cuales se hace uso para el desarrollo del proyecto son artículos, documentación de las herramientas de juzgamiento Open Source que se consideren importantes, además de diferentes tesis y proyectos de investigación que se asocian al presente proyecto. Estas fuentes provienen de bases de datos y plataformas tales como Google Scholar, ScienceDirect, Ebscohost, e-libro entre otras.

Capítulo 9. Recursos

9.1 Recursos físicos

Los recursos físicos necesarios para el desarrollo del proyecto fueron:

- Un computador que soporte Oracle VM Virtual Box, con una capacidad de almacenamiento suficiente para realizar copias de seguridad de la máquina virtual.
- Tres computadores para la realización de pruebas de la herramienta
- La cantidad de computadores para llevar a cabo la simulación, dependerá de la cantidad de equipos (teams) inscritos, esto a su vez permitirá conocer el número de salas conectadas (LAN) que se requerirán.

9.2 Recursos humanos

Las personas que se emplearon para el desarrollo del proyecto son:

- Jhon Fredy Plazas Hurtado. Estudiante.
- Miguel Ángel Castellanos Ibáñez. Estudiante.
- Diego Fernando Rodríguez Castañeda. Estudiante.
- Sebastián Contreras Gómez. Estudiante.

Capítulo 10. Cronograma de Actividades

En este capítulo se va a representar el cronograma de actividades donde están plasmadas las actividades completas a realizar, para esto se puede ver en el Anexo 14. Cronograma de Actividades:

Anexo 14. Cronograma de Actividades

Miguel Ángel Castellanos Ibáñez – Jhon Fredy Plazas Hurtado
miguela.castellanosi@ecci.edu.co – jhonf.plazash@ecci.edu.co

Conclusiones

- Aunque existan una gran cantidad de juzgadores Open Source, muchos de estos no cumplen con los lineamientos estipulados por ACM/ICPC, razón por la cual, no prepara al estudiante con las reglas que se presenta el día de una maratón de programación.
- El estudio realizado de los juzgadores Open Source y de los lineamientos que se escogieron para elegir el mejor juzgador para este proyecto, permitió llegar a determinar que DOMJudge es la mejor herramienta que posibilita el juzgamiento de algoritmos.
- Al realizar la investigación y diseño de la infraestructura necesaria para desarrollar un concurso de programación, se pudo comprobar que la topología nodo/servidor es la más sencilla de implementar y la menos costosa en ámbitos de tiempo y dinero.
- A la hora de realizar el montaje de la herramienta, se pudo observar que la documentación existente para hacer dicho proceso omitía algunos detalles relacionados a errores comunes que pueden surgir a lo largo de este procedimiento. Por esta razón en el manual de instalación se especifica detalladamente los pasos que se deben seguir para lograr con éxito el levantamiento del juzgador.
- Este proyecto de grado permitió el desarrollo de una simulación la cual fue presentada como “1 maratón interna de la Universidad ECCI”, confirmando así, las funciones y características que el juzgador DOMJudge proporciona.
- El presente proyecto permitió escoger una herramienta que satisficiera a las necesidades que el grupo de maratones de programación competitiva de la Universidad ECCI presentaba, esto con el fin de que los equipos logren un nivel más competitivo además de generar entrenamientos más cercanos a la realidad.

Recomendaciones

Una vez concluido el proyecto de grado, se recomienda:

- Implementación del juzgador escogido, como una nueva herramienta pedagógica para la evaluación de los temas vistos en las asignaturas de programación y algoritmia.
- Modificar el juzgador para que permita dar a conocer al estudiante los casos de prueba que el algoritmo implementado no solucione.
- Implantar el juzgador en un servidor en la nube facilitando así la creación de maratones de programación para tener acceso desde cualquier lugar con acceso a internet.
- Crear un aplicativo web que permita a los estudiantes el envío de códigos fuente que dé solución a un problema de un banco de problemas, además posibilitando el acceso a material de estudio.
- Modificar el juzgador para que permita mostrar además del PDF del problema, un video explicando en lenguajes de señas el ejercicio a desarrollar, esto con el fin de permitir la inclusión de personas con discapacidad auditiva a este tipo de eventos.

Bibliografía

Acerca de | Google Code Jam. (n.d.). Retrieved September 28, 2018, from https://code.google.com/codejam/about

Apache License, Version 2.0. (n.d.). Retrieved November 10, 2018, from https://www.apache.org/licenses/LICENSE-2.0

Artistic Licenses | Open Source Initiative. (n.d.). Retrieved November 10, 2018, from https://opensource.org/licenses/artistic-license

Becerra, L. m. j. (2012). herramienta de estudio para las maratones de programacio´ n promovidas en el programa de ingenier´ia de sistemas y computacio´ n de la universidad tecnolo´ gica de pereira. 159.

BSD license definition. (n.d.). Retrieved November 10, 2018, from http://www.linfo.org/bsdlicense.html

Campos, C. P. D., Ferreira, C. E., & Anido, R. (2010). *Brazilian Contest Infrastructure: BOCA and Maratona Linux∗*. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.679.4528

Concurso de computadora del concurso de programación ACSL. (n.d.). Retrieved September 28, 2018, from http://www.acsl.org/

El Concurso Internacional de Programación Universitaria ICPC. (n.d.). Retrieved September 28, 2018, from https://icpc.baylor.edu/

Equipo de la U.N., a Maratón Mundial de Programación. (n.d.). Retrieved May 24, 2019, from Agencia de noticias website: {$config.domain}http://agenciadenoticias.unal.edu.co/detalle.html&tx_ttnews[tt_news]=

GNU General Public License v2.0 only | Software Package Data Exchange (SPDX). (n.d.). Retrieved November 10, 2018, from https://spdx.org/licenses/GPL-2.0.html

gnu.org. (2018). Retrieved September 30, 2018, from http://www.gnu.org/philosophy/free-sw.es.html

gnu.org. (n.d.-a). Retrieved November 10, 2018, from https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html

gnu.org. (n.d.-b). Retrieved November 10, 2018, from https://www.gnu.org/licenses/gpl-3.0.en.html

Gómez, J., León, E., Cubides, C., Rodríguez, A., Mahecha, J., & Rubiano, J. C. (2014). Evolution of teaching and evaluation methodologies: The experience in the computer programming course at the Universidad Nacional de Colombia. *Evolución de Las Metodologías de Enseñanza y Evaluación: La Experiencia Del Curso de Programación de Computadores En La Universidad Nacional de Colombia.*, *34*(2), 85–89.

League 2019, C. C. P. (n.d.). Colombian Collegiate Programming League 2019. Retrieved May 24, 2019, from Colombian Collegiate Programming League 2019 website: https://www.programmingleague.org/

ley 603 de 2000 en Colombia. (2015, October 21). Retrieved September 23, 2018, from E-SOLUCIONES TIC website: http://www.e-solucionestic.com/ley-603-del-2000/

ley44_1993.pdf. (n.d.). Retrieved from http://propiedadintelectual.unal.edu.co/fileadmin/recursos/innovacion/docs/normatividad_pi/ley44_1993.pdf

Ley_1273_2009.pdf. (n.d.). Retrieved from http://www.sic.gov.co/recursos_user/documentos/normatividad/Ley_1273_2009.pdf

LEY_1581_DE_2012.pdf. (n.d.). Retrieved from https://www.unicauca.edu.co/versionP/sites/default/files/files/LEY_1581_DE_2012.pdf

Manzoor, S. (2008). Common Mistakes in Online and Real-time Contests. *XRDS*, *14*(4), 10–16. https://doi.org/10.1145/1375972.1375976

Maratón de Programación | ACIS. (n.d.). Retrieved October 1, 2018, from http://acis.org.co/portal/content/acis-marat%C3%B3n-de-programaci%C3%B3n

Moreno, J., Pineda, A. F., & Montoya, L. F. (2013). *Uso de un ambiente virtual competitivo para el aprendizaje de algoritmos y programación - Experiencia en la Universidad Nacional de Colombia*. 4.

Quora. (n.d.). To Get Coding Jobs At Companies Like Google And Microsoft, Should I Learn C++ Or Java? Retrieved May 28, 2019, from Forbes website: https://www.forbes.com/sites/quora/2017/03/28/to-get-coding-jobs-at-companies-like-google-and-microsoft-should-i-learn-c-or-java/

Reglas IPSC 2018 - IPSC. (n.d.). Retrieved September 28, 2018, from https://ipsc.ksp.sk/rules

The ICPC International Collegiate Programming Contest. (n.d.). Retrieved October 1, 2018, from https://icpc.baylor.edu/regionals/finder/world-finals-2018

The MIT License | Open Source Initiative. (n.d.). Retrieved November 10, 2018, from https://opensource.org/licenses/MIT

The Q Public License Version (QPL-1.0) | Open Source Initiative. (n.d.). Retrieved November 10, 2018, from https://opensource.org/licenses/QPL-1.0

Timus Online Judge. (n.d.). Retrieved September 28, 2018, from http://acm.timus.ru/

UVa Online Judge. (2018). In *Wikipedia*. Retrieved from https://en.wikipedia.org/w/index.php?title=UVa_Online_Judge&oldid=852359152