Diseño Y Programación Orientado A Objetos

Adriana Sofia Rozo Cepeda - 202211498

Repositorio taller 3:

<https://github.com/SofRozo/Taller-3-DPOO.git>

--------------------------------------------------------------------------------------------------------------

**Punto 1 – Requerimientos Funcionales**

Para documentar los requerimientos funcionales, se utilizaron descripciones para demostrar la funcionalidad de este requerimiento para cada uno de los autores.

1. **Autor de un artículo (usando la aplicación)**
   1. *Registrar autor:*

Debido a que uno de los objetivos de la plataforma es brindar facilidades para cada uno de sus usuarios, se realizó la clase Sistema Editorial donde los autores de los artículos mediante su registro con primero su login y contraseña y segundo nombre e email, podrán hacer los envíos, consultar los comentarios de los revisores, y cargar toda la información necesaria para la publicación final.

* 1. *Enviar nueva versión del articulo:*

Conociendo que, si el revisor no aprueba el artículo, si es decisión de los autores, se puede enviar una nueva versión del artículo con los cambios, se creó una clase llamada Autor donde uno de sus métodos *CambiosArticulo*, permite que estos puedan mandar una versión corregida a la editorial para comenzar nuevamente con el proceso. El proceso continua hasta que en la clase Tramite y Revisión cambie el estado de *continuaProceso* a False.

* 1. *Añadir autores cuando el artículo sea aprobado:*

Cuando se manda la versión preliminar del artículo (primer paso para la publicación), se utiliza un Corresponding Author, quien en la primera versión no suministra los datos de los demás autores. Con lo anterior, al ser aprobada la publicación, se debe enviar una nueva versión con esta información añadida. Este proceso se puede realizar en la clase Autor con su método *cambiosArticulo*, la cual recibe la respuesta (que se realiza en la clase Tramite y Revisión) que siendo afirmativa puede enviar el artículo con la información.

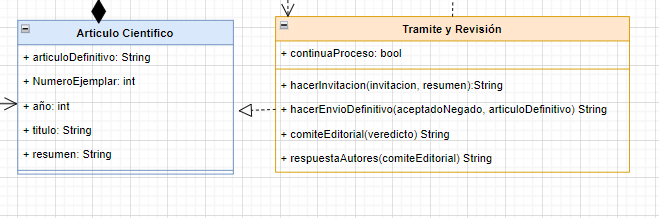
* 1. *Enviar sus artículos e información a la Editorial*

|  |
| --- |
| **Historia de Usuario** |
| *Como autor de un artículo, deseo tener una plataforma que esté habilitada para el envío de mis requerimientos para que estos entren en el proceso de revisión y posterior publicación. Es importante que la plataforma me permita enviar los cambios pertinentes de mi entrega y que esto sea intuitivo.* |

Para suplir con esta solicitud, se realizó una clase llamada SistemEditorial, la cual está encargada de tratar los datos de sus miembros. Con los métodos *cargarDatosyPublicaciones* y *enviarArticulosCambios,* se puede suplir este requerimiento.

1. **Revisor**
   1. *Ser invitado por parte de la editorial:*

En la clase Tramite y Revisión se tiene un método llamado hacerInvitacion, este es usado por la Editorial para solicitar a los revisores que validen un texto, este método permite tener en sí una invitación formal y el abstract del artículo científico, llamado en la clase Articulo Científico como Resumen.



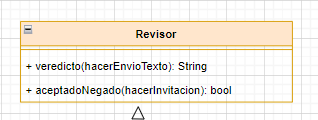


* 1. *Acceder al artículo y revisarlo*

*En* la clase Revisor con el método *aceptadoNegado*, se recibe la información anteriormente explicada, este método retorna un bool donde se dice si este es aceptado o no para que así los autores y la editorial reciban la información de su decisión y sepan que paso seguir.

* 1. *Aceptar revisión articulo completo*

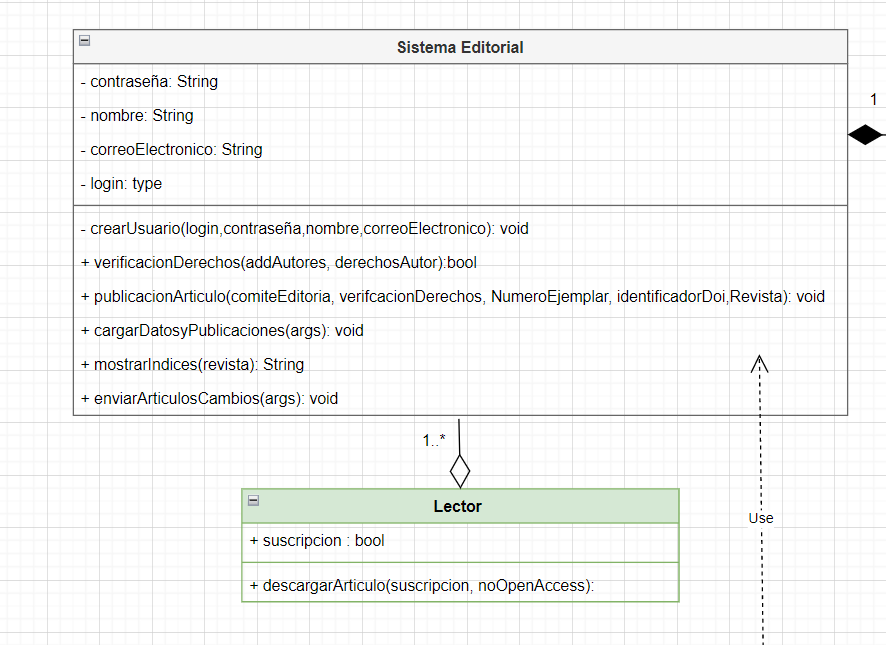
El requerimiento funcional anterior, dicta si el abstract fue aprobado, de ser así se hará la revisión del artículo completo. Cuando esto sucede, los revisores necesitan mandar su veredicto a la Editorial que es Aceptado, Pendiente de cambios menores, Pendiente de cambios mayores, o Rechazado definitivamente. Esto se hace con el método *veredicto* en la clase Revisor que recibe el texto y retorna las respuestas anteriormente indicadas.

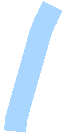




1. **Lector de la revista**
   1. *Descargar artículos:*

Como se había mencionado para los autores, los lectores también están registrados con su login, contraseña, nombre y email, en la clase sistema editorial. El lector puede tener también una suscripción en una plataforma que no sea OpenAccess que le permite descargar Artículos siempre que sea posible. A su vez, si la plataforma es OpenAccess sus artículos también lo son y en ese caso los pueden descargar. También, es importante mostrar los índices, los cuales son ajenos a la editorial pero que deben estar ahí.





**Punto 2 – Diagrama UML**

***Clases:***

**Sistema Editorial:**

Esta clase representa el sistema como tal, podemos relacionarlo con la “consola” del programa y procesamiento puesto que es donde se recibe la información del login de cada uno de los usuarios, se cargan datos y artículos y es donde se pueden realizar búsquedas.

**Lector:**

Esta clase representa al usuario directo del sistema, quien realiza lectura de las publicaciones, en ciertos casos paga una suscripción y debe tener facilidades como la búsqueda de artículos y revisar estadísticas de estos.

**Editorial:**

La clase editorial tiene en su información los daros de los autores con sus correspondientes artículos. Esta tiene un método llamado *addInfoAutores*, el cual dado un artículo y sus autores permite actualizar este ultimo en dado caso que la primera revisión (abstract) fuera aprobada. Este método facilita realizar ciertas actualizaciones en la segunda etapa del proceso de revisión.

**Revista:**

Esta es una de las clases más importantes, puesto que se relaciona directamente con la Editorial. Esta relación es de uno a muchos puesto que una editorial, puede tener una o más revistas. Cada una tiene su nombre, la verificación de OpenAccess, su ISSN y el área de conocimiento para que sea clasificada. A su vez, contiene el método para identificar qué acción realizar en caso de no ser OpenAccess. El método noOpenAccess recibe información de la preferencia de visibilidad del autor y de la revista como tal y:

1. Si la revista es OpenAccess, todos sus artículos lo son.
2. Si no es OpenAcess y el autor dice que su articulo si lo es, este debe pagarle a la Editorial el monto que ganarían si este no lo fuera.
3. Si no es OpenAccess se cobra para verlo (suscripción).

**Artículo científico:**

Esta clase esta conectada a Revista, esta compuesta por un objeto llamado artículo y contiene dentro un método llamado *hacerResumen* que recibe el articulo y realiza el abstract que es enviado a la primera etapa de revisión a los revisores.

**Autor:**

Los autores tendrán atributos básicos los cuales darán datos importantes para la revisión y carga del articulo a la plataforma de las revistas, estarán asociados con la clase editorial para a su vez relacionarse con el articulo y finalmente se relacionan con el sistema editorial el cual les confiere sus credenciales de acceso y en este se realice todo el cargue de documentos.

**DOI:**

**Numero el cual lo tramita la editorial para distinguir al artículo.**

**Revisor:**

Clase en la cual se realizan la revisión de si un artículo es aprobado o no, está conectada con Tramites y revisión para recibir toda la información que requiere para realizar sus métodos de manera correcta.

**Tramite y Revisión:**

En esta es donde llegan todos los archivos y veredictos dados por los revisores y el comité de la editorial para dar respuesta al autor o a la clase publicación para dar el siguiente paso en el proceso.

