UML Actividad#3

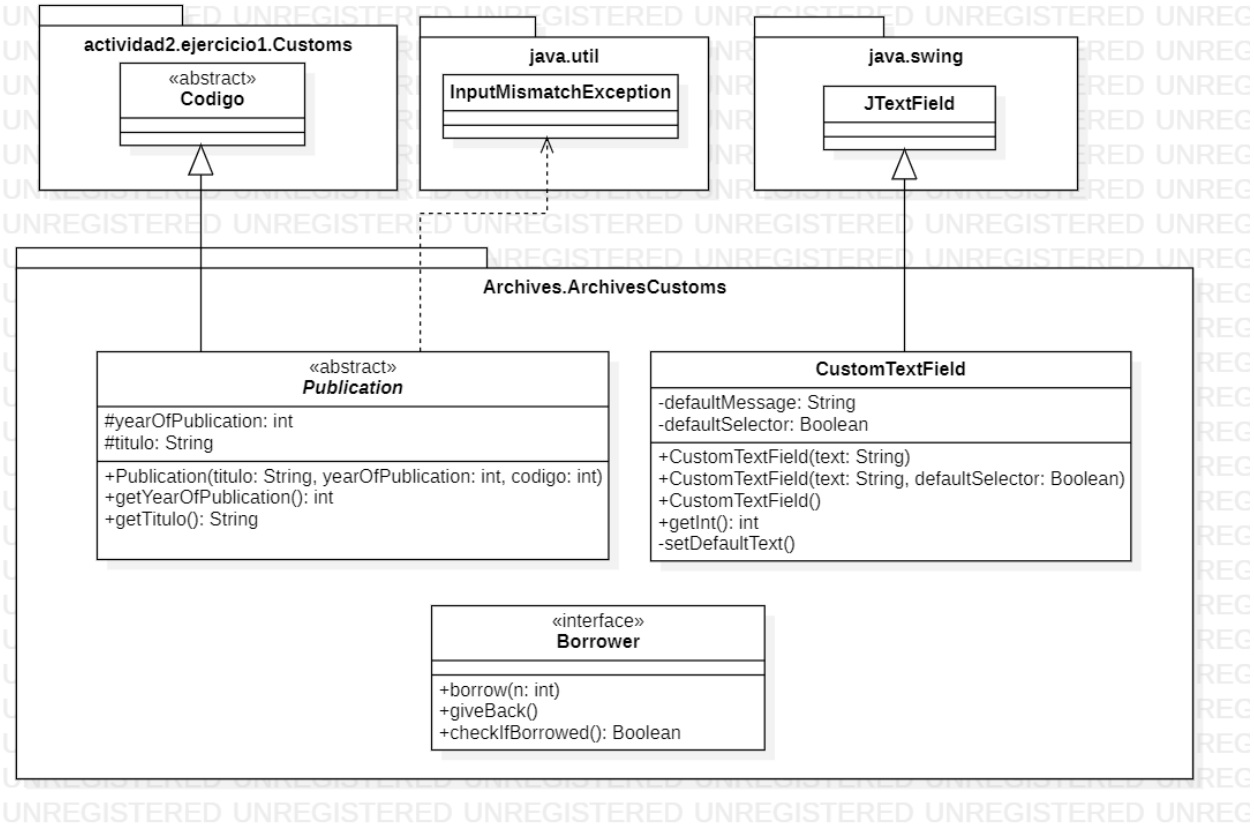
Jorge Enrique Celis Cortés

Facultad de Estudios a Distancia, Universidad Militar Nueva Granada

200267: Programa de ingeniería informática a distancia

26 de marzo de 2023

UML Archives Customs



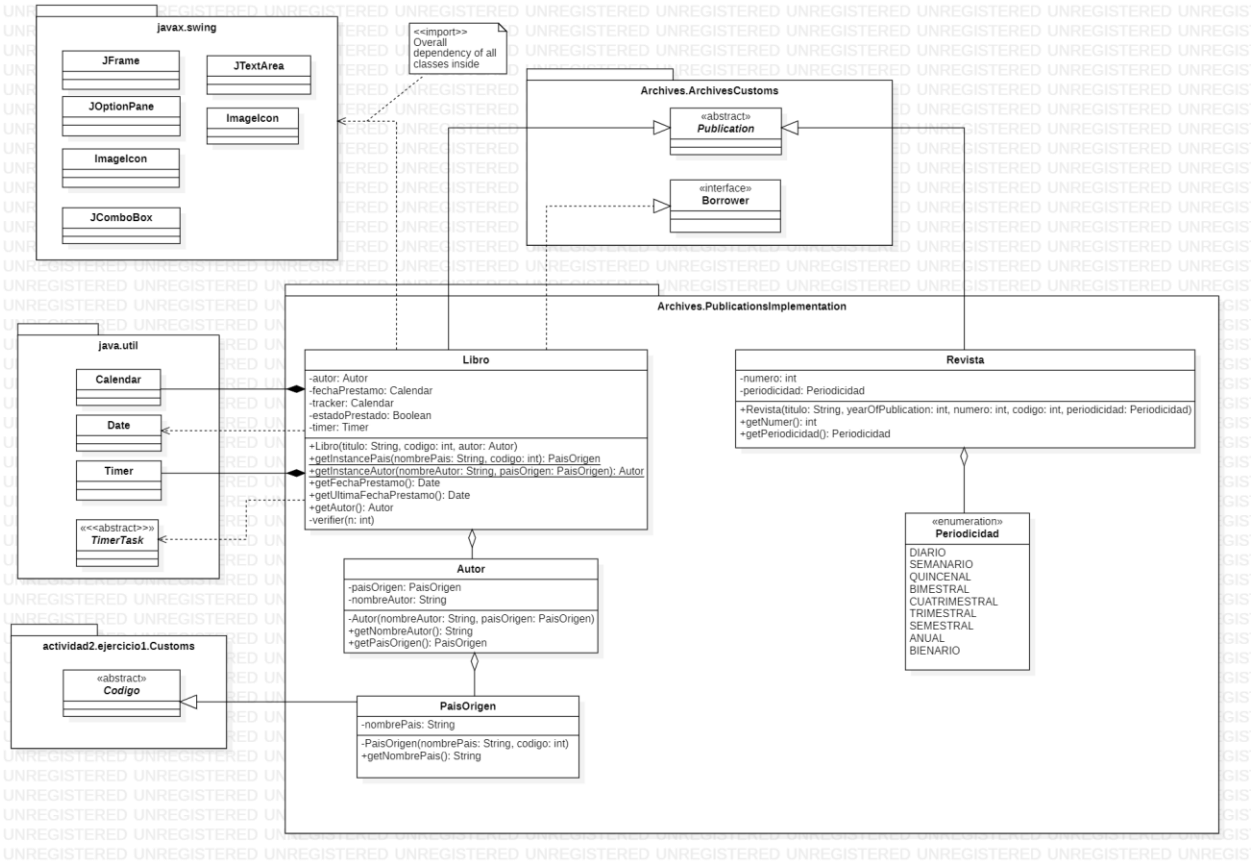
En el paquete nominado ArchivesCustoms se implementan las clases edificadoras del programa. La primera clase es la clase abstracta Publication, extensión de Codigo, debido a que todas las publicaciones poseen un código. A su vez, esta es dependiente de InputMismatchException (instancias de Publication solo usan a esta clase con fines de arrojar errores cuando se ingresan números negativos en su construcción, por lo que no es necesario que objetos de la clase InputMismatchException sepan de la existencia de los antedichos).

Después se tiene la clase CustomTextField, hija de la clase JTextField. Se creó está clase con el fin de tener una forma asertiva, más no segura, de conseguir input tipo integro de los usuarios utilizando el API de javax.swing.

Por último, se declaró la interfaz Borrower, cuya funcionalidad es la de formar un contrato donde cualquier clase que realice a esta interfaz se vea obligada a implementar

comportamientos de entidades que son capaces de ser prestadas por cierta cantidad de tiempo, de ser verificadas respecto a su "estado" de prestación, y, por supuesto, instancias de clases que implementen esta interfaz también poseen la facultad de poder devolverse.

UML Publications Implementation



En el paquete nominado PublicationsImplementation se implementaron las clases realizadoras del esqueleto hallado en el paquete ArchivesCustoms. Primero, se creo la clase Libro, extensión de Publication e implementadora de Borrower, con el fin de simular libros que poseen un autor, código de identificación, año de publicación, y un título. Instancias de Autor poseen una asociación de agregación con instancias de Libro debido a que estos últimos son capaces de retener referencias a los susodichos, empero no presentan un "ownership" estricto (es posible que varios libros compartan el mismo autor e inclusive que la extinción de instancias de

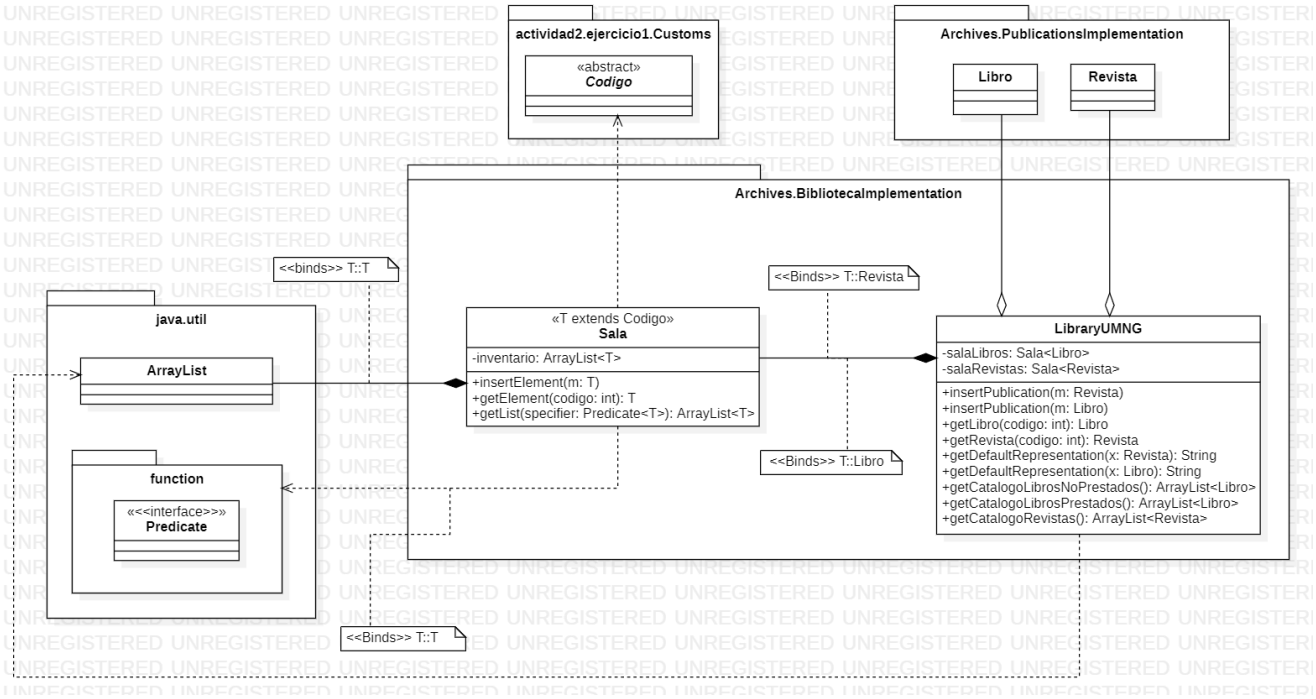
Libro no conlleve a la destrucción de los autores). La clase Libro también hace uso exclusivo de instancias de las clases Timer y Calendar, instancias que no se pueden compartir entre diferentes objetos y tampoco sobrepasan el lapso de vida de instancias de Libro. En cambio, debido a la implementación de los métodos, la clase Libro es dependiente de las clases nombradas en el paquete javax.swing dentro del diagrama, a la par que dispone de las clases Date y TimerTask por la mismas razones.

Dentro de la implementación de la clase Autor es apreciable la referencia a instancias de la clase PaisOrigen, que en un cariz semejante a la clase Libro, un país de origen puede ser compartido por más de un autor y su lapso de vida es generalmente independiente a instancias de Autor.

La clase Revista es una extensión de la clase Publication. Esta clase presenta una relación de agregación con la enumeración Periodicidad, ya que, entendiendo a esta como una clase, objetos de Periodicidad pueden ser compartidos entre varias instancias de Revista, e inclusive sobrepasar el lapso de vida de estas, aunque es valido argumentar que esto no es verdaderamente agregación y debiese considerarse mejor una asociación por el simple hecho de que se está tratando con una enumeración, y por su naturaleza, sus “elementos” son constantes pseudo indistinguibles de build-in types, impidiendo su alteración por el paso de referencias. Aún así, el autor considero que este diagrama es el adecuado por la última razón mencionada: son objetos que son pasados por valor a la referencia periodicidad.

Para más información se sugiere que se dirija a la documentación adjunta a este documento.

UML Biblioteca Implementation



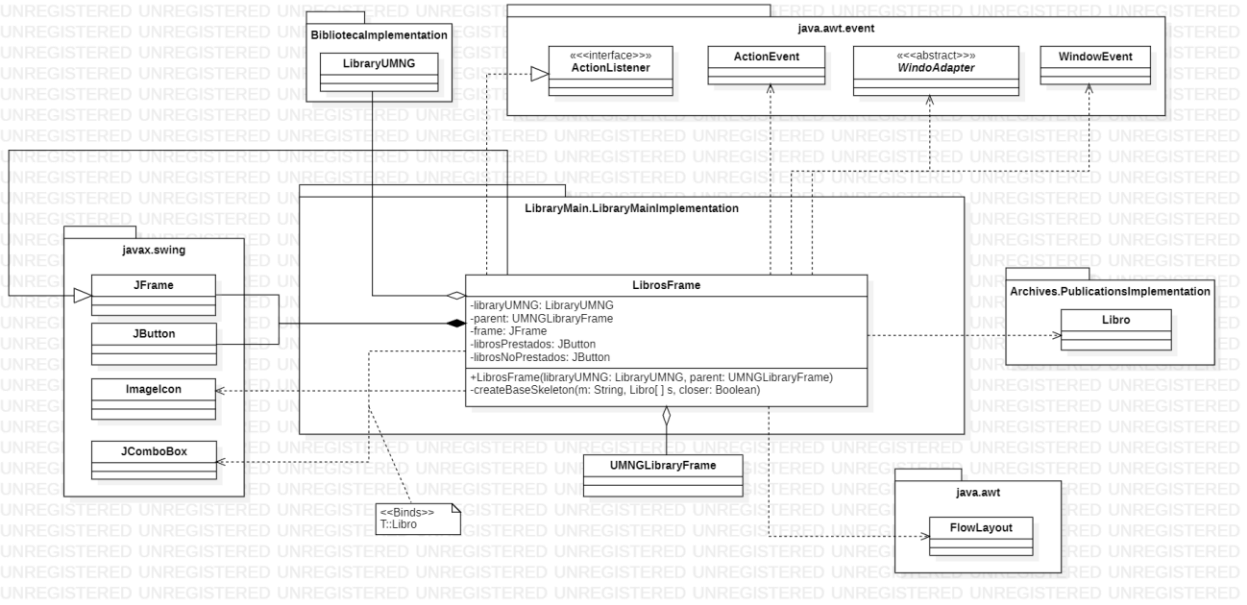
En este paquete se implementaron las clases “de ejecución simple” que ayudarán al autor más adelante. La primera clase que debería exaltar curiosidad es la clase genérica Sala, la cual emplea compositivamente una instancia de la clase ArrayList enlazada por parametrización de tipos derivados de la clase abstracta Codigo; consiguiendo esta una relación de dependencia con la clase Sala, dependiente también de la interfaz funcional Predicate para hacer una “composición simuladora de sobreescritura” en el método getList().

Luego, se hace la implementación de la clase LibraryUMNG. Esta se compone de dos instancias de la clase genérica Sala; una para manejar instancias de Libro y otra para manejar instancias de Revista. Debido a que se han de almacenar instancias tanto de Libro como de Revista, más no implica que estas se destruyen o se alteren significativamente con el uso de métodos de instancias de LibraryUMNG, es correcto afirmar que estas se relacionan por agregación con la clase LibraryUMNG.

Note: debiese de ser obvio la dependencia de la clase LibraryUMNG de la clase ArrayList debido a la naturaleza de implementación compositiva de la clase Sala, dejando que esta relación sea mejor deducida por contexto que dejada explícitamente. Sin embargo, esto es más una ayuda tanto para el autor como para el lector en caso de presentarse ambigüedades concernientes a la construcción de futuros modelos.

UML LibraryMain

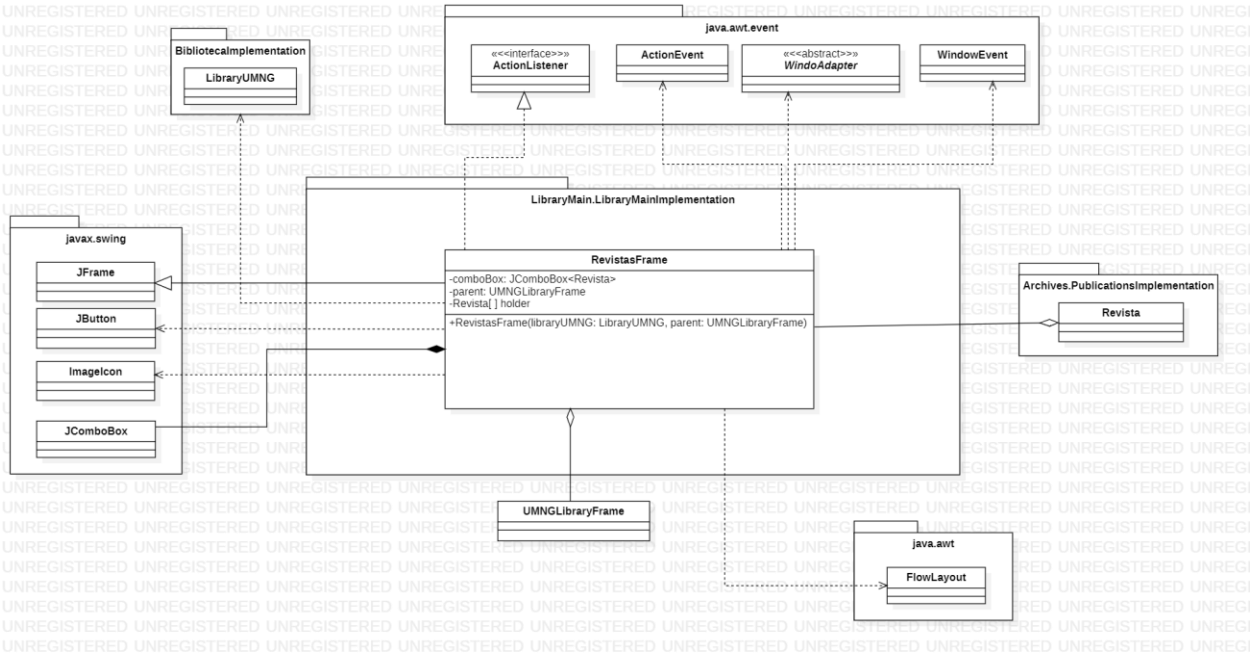
UML LibrosFrame



Clase hija de JFrame e implementadora de la interfaz ActionListener, la clase LibrosFrame tiene la funcionalidad de proporcionar un GUI a través del uso parcial del API javax.swing para que un usuario cualquiera pueda acceder a los catálogos de libros prestados y/o no prestados de la referencia, y por ende implementación por agregación, de instancias de LibraryUMNG. Esta hace uso compositivo de las clases JFrame y JButton, ya que no se pueden compartir con otros objetos y su duración dentro del programa llega a ser menor o igual a la de instancias de LibrosFrame. Esta clase implementa por agregación a instancias de la clase UMNGLibraryFrame y libraryUMNG, ya que no presenta una pertenencia exclusiva de estas.

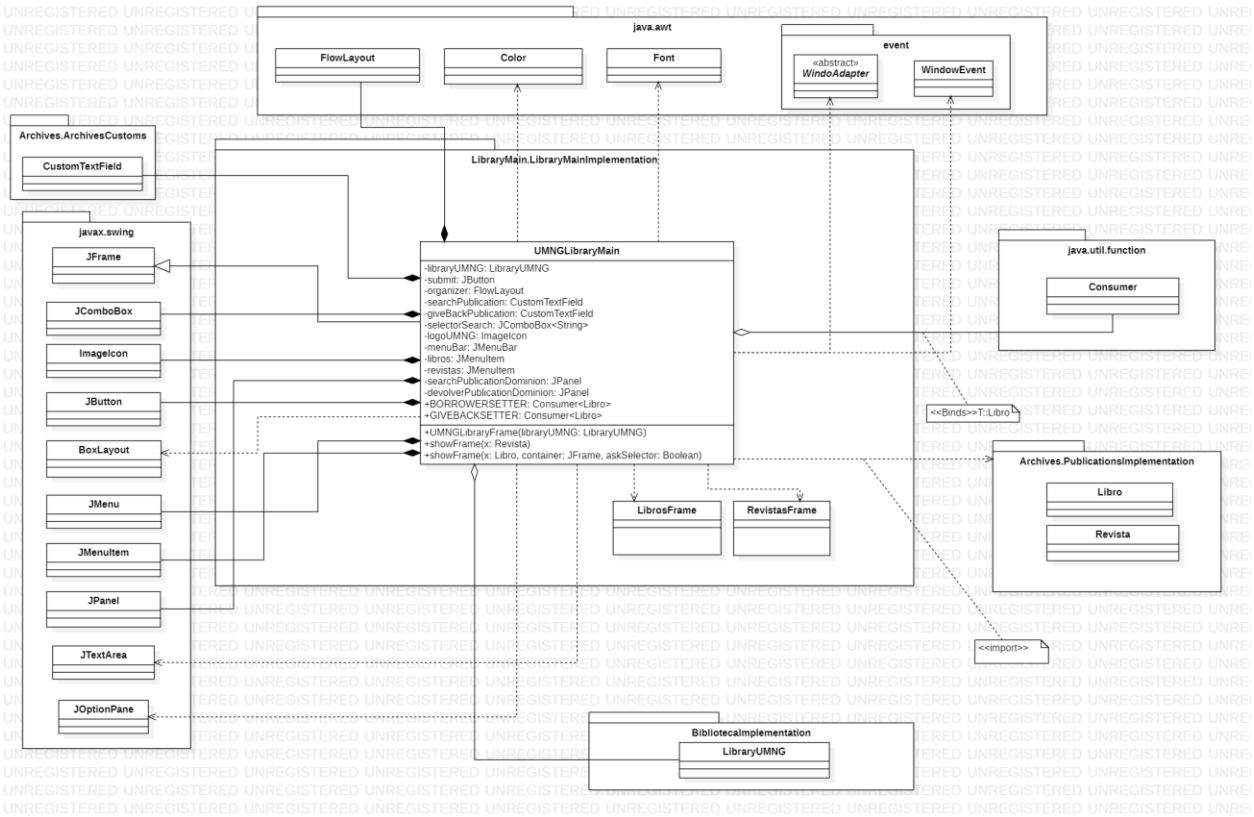
En términos de dependencia, debido a la naturaleza de su constructor y la sobreescritura del método implementado por la interfaz ActionListener, esta clase hace un uso extensivo de las clases ActionEvent, WindowEvent, WindoAdapter, ImageIcon, JComboBox, y Libro, pero el autor decidió dejarlo hasta ahí, ya que estos como tal, al menos a lo que respecta la modelación por medio de diagramas UML, no se presentan por instancias en la clase LibrosFrame; aunque podría argumentarse que existe una relación de agregación con la clase Libro debido a la referencia que se tiene con la clase LibraryUMNG, pero como esta última solo se pasa por referencia, no es un caso lo suficientemente fuerte como para denotar dicha relación.

UML RevistasFrame



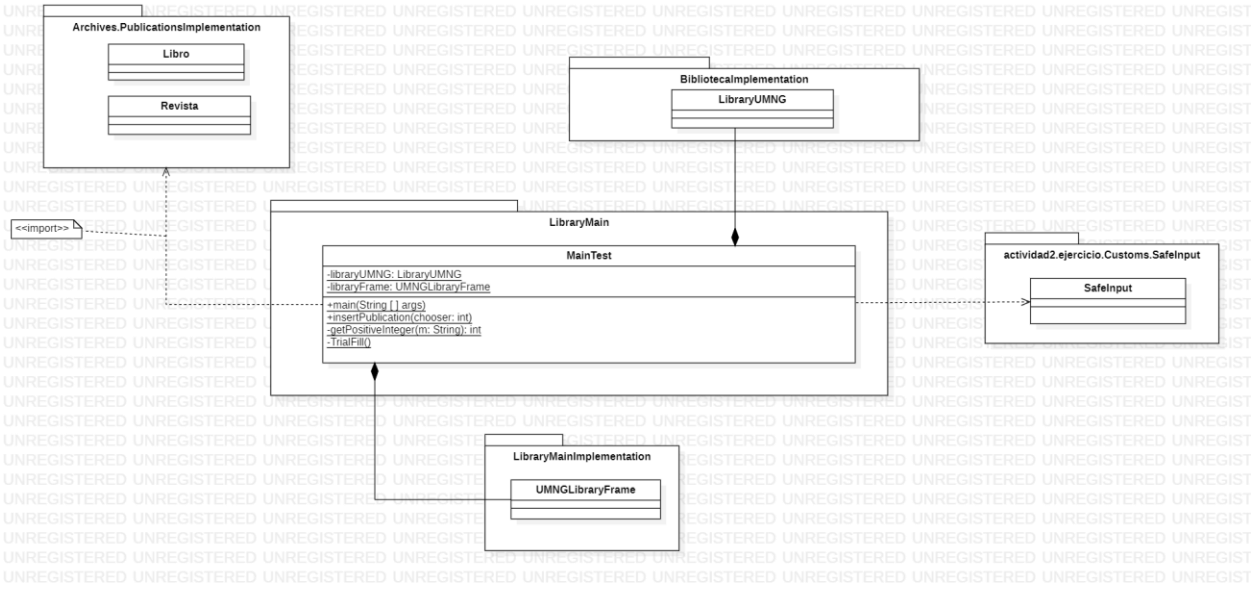
La clase RevistasFrame presenta un modo de actuar muy similar a la clase LibrosFrame, con la excepción de que implementa por agregación instancias de la clase Revista y se compone únicamente de una instancia de la clase genérica JComboBox.

UML UMNGLibraryFrame



La clase UMNGLibraryFrame es una clase hija de la clase JFrame. La idea de esta clase es la creación de un ambiente GUI al que el usuario o el administrador se verá expuesto para interactuar activamente con instancias de LibraryUMNG. Esta clase implementa compositivamente a todas las clases mostradas en el diagrama debido a que se usan activamente dentro de instancias de UMNGLibraryFrame y solo pueden existir dentro de estas, además que, debido a la naturaleza del constructor de esta clase y las dependencias, no existe forma alguna en que el periodo de vida de estas exceda el periodo de vida de la instancia. Por agregación se implementaron instancias lambda de la interfaz Consumer atada al parámetro T::Libro. Se dice que por agregación porque son atributos públicos.

UML MainTest



Este es el modelo UML de la clase Test que se utilizo para probar la implementación de las clases anteriores. Se utilizo un modelaje de composición con las LibraryUMNG y UMNGLibraryFrame modelando o entendiendo a la clase MainTest como un objeto por sí solo. Las dependencias son claras si se hace una lectura cabal de la documentación.