FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2 — AlgoCraft

[7507/9502] Algoritmos y Programación III
Curso 1
Primer cuatrimestre de 2019

Alumno:	CAMBIANO, Agustín
Número de padrón:	102291
Email:	aguscambiano@gmail.com
Alumno:	SORIANO, Ivan
Número de padrón:	102342
Email:	ivan.soriano@live.com
Alumno:	ROLANDO, Marcos
Número de padrón:	102323
Email:	marcosrolando.mr@gmail.com
Alumno:	MONTENEGRO, Lucas
Número de padrón:	102412
Email:	lu.montenegro98@gmail.com

Índice

1. Introducción

El presente informe reúne la documentación de la solución del segundo trabajo práctico de Algoritmos y Programación III, que consiste en crear una imitación del juego Minecraft, haciendo uso de patrones de diseño. El patrón más destacado en el trabajo es el denominado "Double dispatch".

2. Supuestos

Debido a que las indicaciones proporcionadas no cubren toda la información necesaria para llevar a cabo el trabajo de una manera determinada, se realizaron ciertas suposiciones que permitieron su realización:

- El mapa es estático y tiene siempre que se ejecute el juego la misma distribución.
- El jugador no puede posicionarse en un lugar del mapa que contenga un material.
- Al romperse un material, este se agrega automáticamente al inventario del jugador.
- Al fallar la construcción de una herramienta se recuperan los materiales utilizados.
- Las herramientas no pueden acumularse en un lugar de un inventario.
- No puede acumularse en un lugar del inventario elementos distintos.
- La máxima cantidad de elementos acumulables en una posición de un inventario es 64.
- Al gastarse completamente la durabilidad de una herramienta, esta desaparece del inventario (se rompe).

3. Modelo de dominio

El diseño del trabajo consiste en la existencia de una clase Jugador, que interactúa con materiales presentes en un mapa, para poder así crear herramientas para obtener otros materiales.

- Jugador: elemento controlado por el usuario, contiene un inventario para guardar las herramientas creadas y los materiales recolectados. Contiene además una mesa de crafteo para poder construir las herramientas disponibles. Se desplaza por el mapa y puede utilizar las herramientas que almacena.
- Herramienta: se utiliza para obtener materiales, hay distintos tipos de herramientas que permiten obtener distintos tipos de materiales. Tienen durabilidad, y cuando esta se agota se rompe.
- Hacha: tipo de herramienta utilizado para obtener madera, puede ser de madera, piedra o metal (todas tienen el mismo comportamiento).
- Pico: tipo de herramienta utilizado para obtener piedra, metal y diamante, puede ser de madera, piedra, metal o un pico fino.
- Materiales: son obtenidos luego de ser golpeados hasta romperse por la herramienta adecuada. Se utilizan para construir las herramientas.
- Pico de madera: es un tipo de pico construido utilizando únicamente madera y solo puede ser utilizado para obtener piedra.
- Pico de piedra: es un tipo de pico construido utilizando madera y piedra y puede ser utilizado para obtener piedra y metal. El pico de metal tiene el mismo comportamiento, con la diferencia de que es construido con madera y metal.

- Pico fino: es un tipo de pico construido utilizando madera,piedra y metal y solo puede ser utilizado para obtener diamante.
- Inventario: es utilizado para guardar los materiales recolectados y las herramientas creadas.
- Paquete: es utilizado para guardar un tipo de elemento almacenable, puede almacenar más de una instancia de un tipo de material pero no puede almacenar más de una instancia de herramienta.
- Mesa de crafteo: es utilizada para crear las herramientas, recibe un código que debe coincidir con el de la herramienta que se desea obtener, en cuyo caso devuelve una instancia de ella.
- Codigo de crafteo: es el código que recibe la mesa de crafteo para la creación de herramientas, su objetivo es representar una distribución de materiales en un espacio de 9 lugares.

4. Diagrama de clase

4.1. Herramientas y Materiales

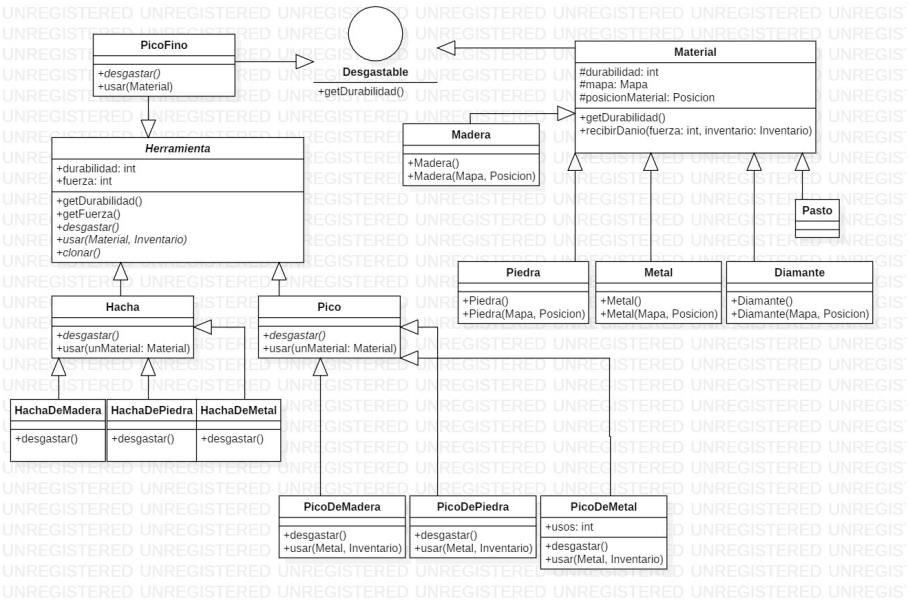

Figura 1: Diagrama de las herramientas y los materiales

4.2. Jugador y elementos agregables a través de la mesa de crafteo

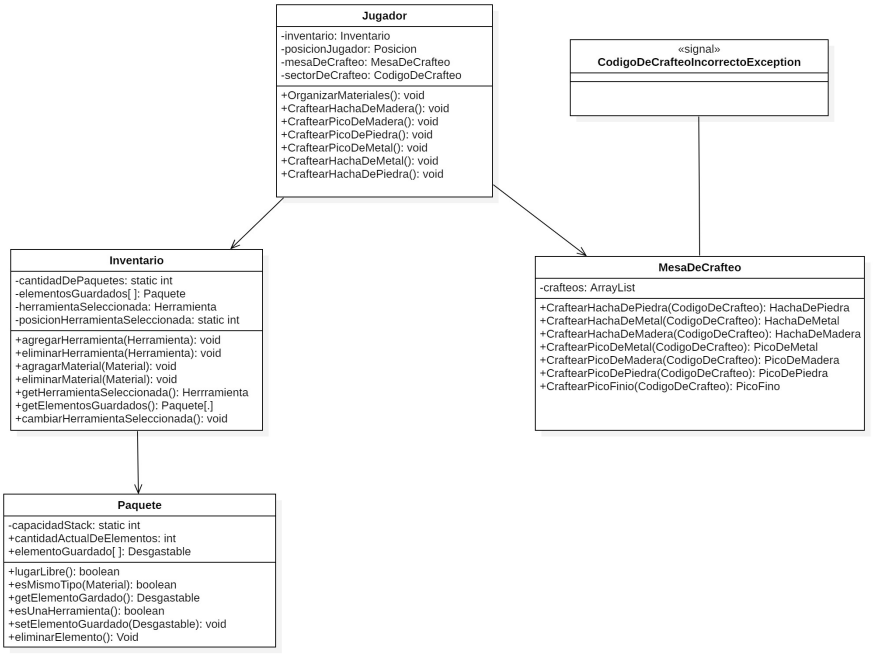

Figura 2: diagrama esquelético con la relación entre el jugador y la mesa, mostrando en términos simples la agregación de materiales y herramientas

4.3. Jugador y Mapa

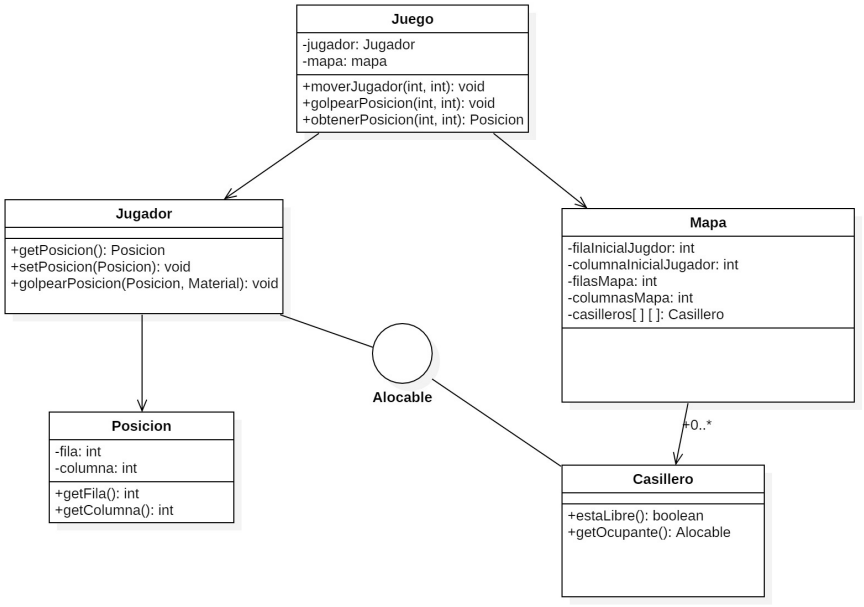

Figura 3: diagrama esquelético con la relación entre el jugador y el mapa, principalmente los movimientos

5. Detalles de implementación

5.1. Materiales

Guardan una durabilidad, una referencia al mapa y una posición. Pueden ser utilizados para construir el código de la herramienta que se quiera craftear. La referencia al mapa permite una implementación sencilla y eficiente de control de estado del material, el cual al romperse notifica al mapa para que este se actualice.

Los materiales que forman parte del juego son:

- Madera: material cuya durabilidad es 10.
- Metal:material cuya durabilidad es 50.
- Piedra: material con durabilidad 30.
- Diamante: material con durabilidad 100.

Además se creó una clase Pasto cuyo comportamiento es nulo (acorde al patrón "null object") para evitar el chequeo de null y poder utilizar polimorfismo sin importar el material al que el jugador golpee.

5.2. Herramienta

Es una clase abstracta utilizada para generalizar el comportamiento perteneciente a una herramienta, contiene los atributos durabilidad (que indica en general cuánto le falta a la herramienta para romperse) y fuerza (que indica cuánta durabilidad le quitará al material correcto en el que se use). Las clases que hereden de esta deben implementar métodos de uso para los distintos materiales (se utilizó el patrón denominado "double dispatch"). Algunas de las clases que heredan de herramienta (que no son necesarias de detallar ampliamente ya que en su estructura son similares) son:

- HachaDeMadera:Herramienta de durabilidad 100 y fuerza 2 y cuyo desgaste es de la forma Durabilidad-=Fuerza.Sirve para usar contra madera.
- HachaDePiedra:Herramienta de durabilidad 200 y fuerza 5 y cuyo desgaste es de la forma Durabilidad-=Fuerza.Sirve para usar contra madera.
- HachaDeMetal:Herramienta de durabilidad 400 y fuerza 10 y cuyo desgaste es de la forma Durabilidad-=Fuerza/2.Sirve para usar contra madera.
- PicoDeMadera:Herramienta de durabilidad 100 y fuerza 2 y cuyo desgaste es de la forma Durabilidad-=Fuerza.Sirve para usar contra piedra.
- PicoDePiedra:Herramienta de durabilidad 200 y fuerza 4 y cuyo desgaste es de la forma Durabilidad-=Fuerza/1.5.Sirve para usar contra piedra y Metal.
- PicoDeMetal:Herramienta de durabilidad 400 y fuerza 12 y que, al golpear 10 veces, pierde su durabilidad.Sirve para usar contra piedra y Metal.
- PicoFino:Herramienta de durabilidad 1000 y fuerza 20 y cuyo desgaste es de la forma Durabilidad-=Durabilidad*0.1,y con la particularidad de que solo se desgasta al usar contra el diamante.

5.3. Jugador

Contiene un inventario, una mesa de crafteo, un código de crafteo y una posición. Puede utilizar alguna de las herramientas que guarda en el inventario para obtener materiales si está seleccionada. Puede utilizar la mesa de crafteo para crear herramientas cargando los materiales recolectados en el sector de crafteo y poniendo este en la mesa. El crafteo se realiza seleccionando en una lista la herramienta que se desea construir, agregando el código necesario y pulsando el botón craftear. El desplazamiento en el mapa y el uso de las herramientas son manejados con teclas.

Como fue dicho previamente, el jugador contiene un sector de crafteo, que es lo que recibirá la mesa para corroborar si puede crearse la herramienta deseada, el código perteneciente al agregado de un material al sector de crafteo es el siguiente:

```
public void agregarMaterialASectorDeCrafteo(int posicionEnCodigo, Material material) {

    inventario.eliminarMaterial(material);
    sectorDeCrafteo.agregarMaterial(posicionEnCodigo, material);

}
```

5.4. Inventario

Contiene un arreglo de 28 paquetes, en los cuales se guardan instancias de elementos equipables de una clase determinada (siendo guardada en el caso de una herramienta una sola instancia), y una herramienta seleccionada, que es la que usará el jugador para obtener los materiales. EL cambio de la herramienta equipada se realiza con una tecla.

Si recibe un material, buscará primero un paquete en el que este se encuentre guardado, sino, buscará el primer paquete vacío, esto se encuentra implementado de la siguiente manera para la madera:

```
public void agregarMaterial(Madera madera) {

    for(int i = 0; i < cantidadDePaquetes; i++) {
        if(elementosGuardados[i].esMismoTipo(madera)) {
            elementosGuardados[i].setElementoGuardado(madera);
            return;
        }
    }

    for(int i = 0; i < cantidadDePaquetes; i++) {
        if(elementosGuardados[i].lugarLibre()) {
            elementosGuardados[i].setElementoGuardado(madera);
            break;
        }
    }

}
```

5.5. Paquete

Contiene un arreglo de 64 posiciones, en las que se guardan instancias de una clase con interfaz equipable.

5.6. Mapa

Se trata de una matriz de diez por trece casilleros dentro de los cuales se guarda algún material o, en su defecto, el jugador. El mapa se encarga de verificar si los casilleros se encuentran libres con el objeto de actualizar la posición del jugador, además de eliminar al material del mapa en caso de que este se haya roto.

5.7. Casillero

Guarda un alocable, que puede ser un material de construcción, el jugador o pasto.

5.8. Posición

Guardas las coordenadas de cualquier objeto alocable, dichas coordenadas las almacena por fila y columna.

5.9. Mesa de crafteo

Contiene una lista con el código de crafteo de cada herramienta. Para construir una herramienta verifica que el código recibido coincida con el de la herramienta que se desea construir, en cuyo caso la devuelve, sino tira una excepción (CodigoDeCrafteoIncorrecto).

5.10. Código de crafteo

Contiene un treemap (un tipo de dato que almacena datos con clave de manera tal que estos se encuentran ordenados según la clave), que es el que se ocupa de indicar la disposición de los materiales en el código. La clave del treemap es un número de 1 a 9, que representa la posición en el sector de crafteo del material almacenado con esa clave.

Una porción de código interesante es la perteneciente a la eliminación de los elementos del código de crafteo, utilizada al fallar la creación de una herramienta, la implementación fue realizada de la siguiente manera:

```
public List<Material> obtenerMateriales() {

    List<Material> materialesGuardados= new ArrayList<Material>();

    for(Entry<Integer, Material> materialPosicionado : codigo.entrySet()) {
        int posicion = materialPosicionado.getKey();
        materialesGuardados.add(codigo.get(posicion));
    }
    codigo.clear();
    return materialesGuardados;

}
```

6. Excepciones

Código de crafteo incorrecto Esta excepción es generada al introducir un código de crafteo incorrecto en la mesa de crafteo para la herramienta que se quiere construir.

No herramienta Esta excepción se genera cuando no se tiene una herramienta equipada, caso que únicamente puede darse si se rompen todas las herramientas y el jugador intenta golpear (ya que con tener al menos una herramienta en el inventario, esta estará equipada).

7. Interfaz Gráfica

La totalidad de la interfaz gráfica fue realizada con javaFx.para la realización del trabajo se siguió con el patrón Modelo-vista-controlador (MVC) que separa los datos y la lógica de negocio de una aplicación de su representación y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario.

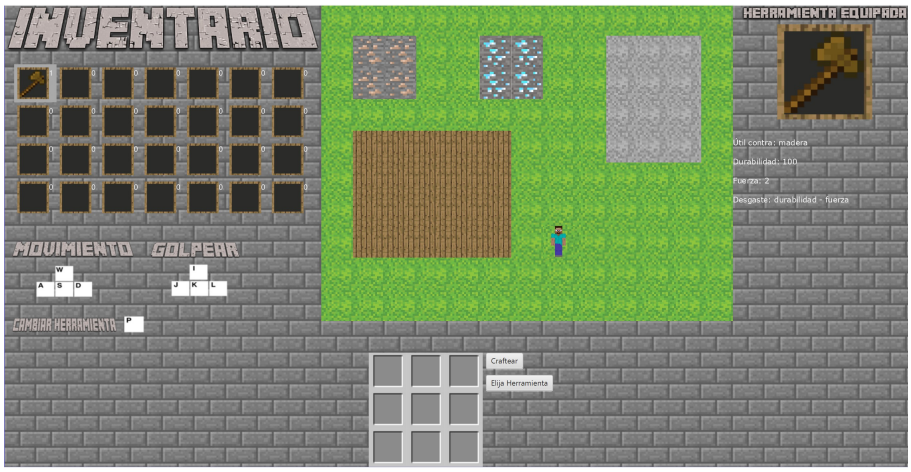

Figura 4: Interfaz gráfica

8. Diagramas de estado

Se presentan algunos diagramas de estado que representan elementos pertenecientes al modelo:

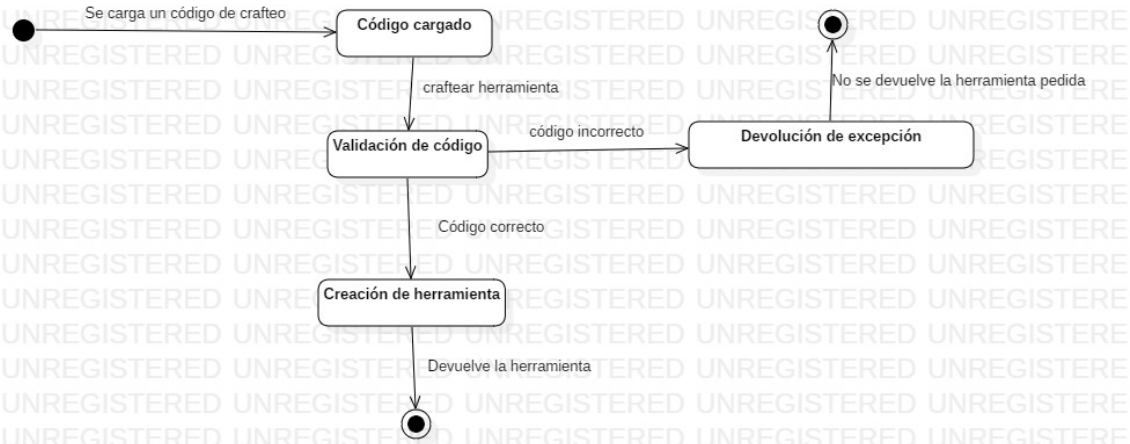

Figura 5: representa el estado de la mesa de crafteo al recibir un código para construir una herramienta.

Figura 6: representa el estado del material al ser golpeado

9. Diagramas de secuencia

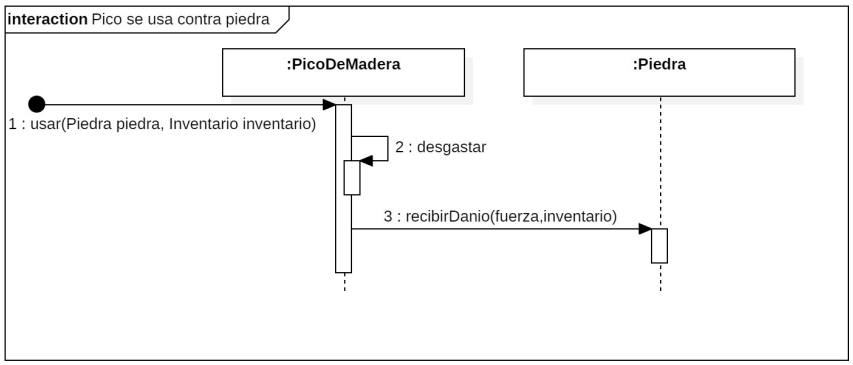

Figura 7: diagrama que muestra que al usar una herramienta, en este caso pico, se desgasta tanto su durabilidad como el material

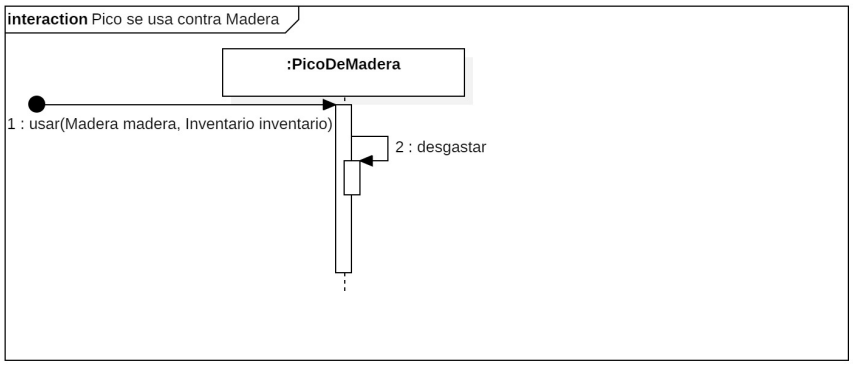

Figura 8: diagrama que muestra que al usar un pico contra madera solo se desgasta a si mismo y no al material

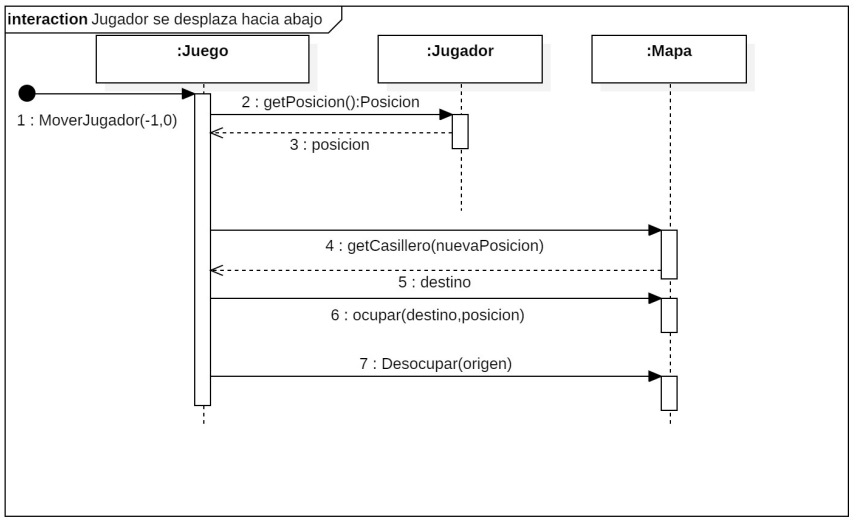

Figura 9: diagrama que muestra el desplazamiento del jugador

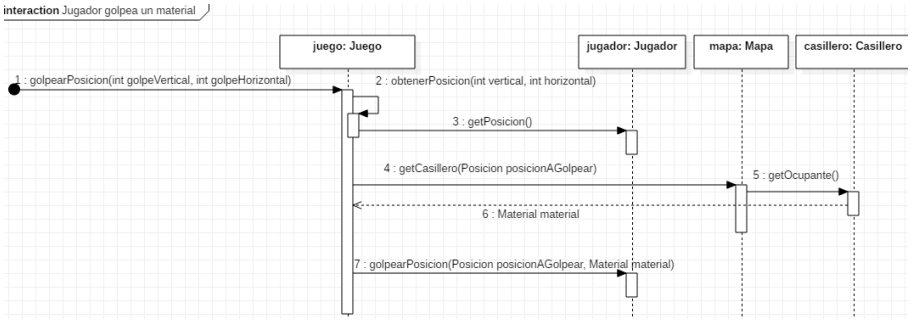

Figura 10: diagrama que muestra el proceso del jugador al golpear a un material.

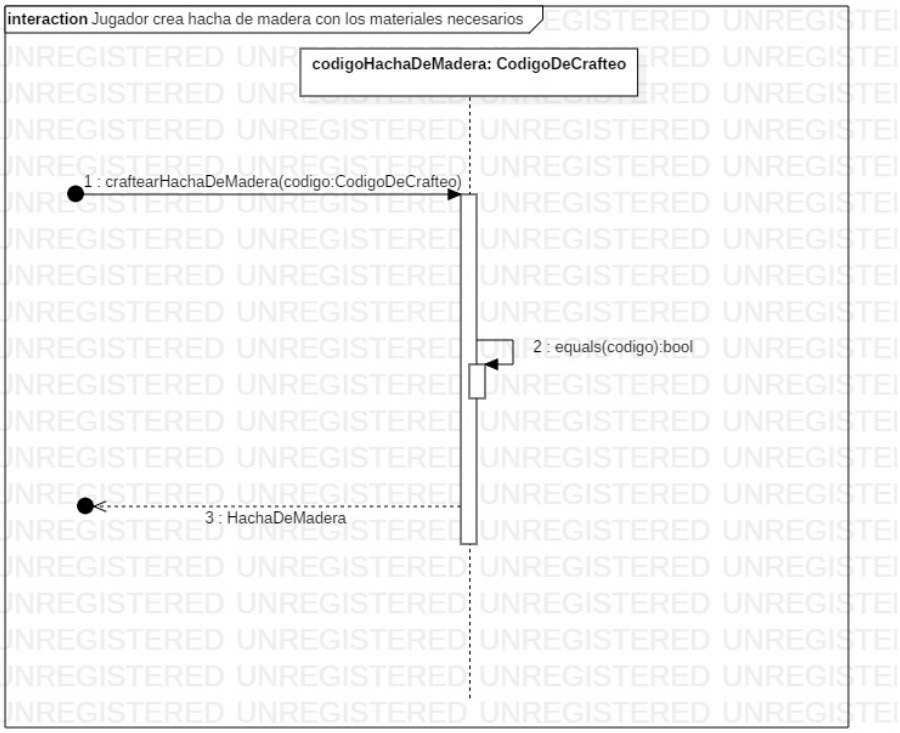

Figura 11: Muestra los procesos realizados en el caso de que se reciba el código de crafteo apropiado para la creación del hacha de madera.

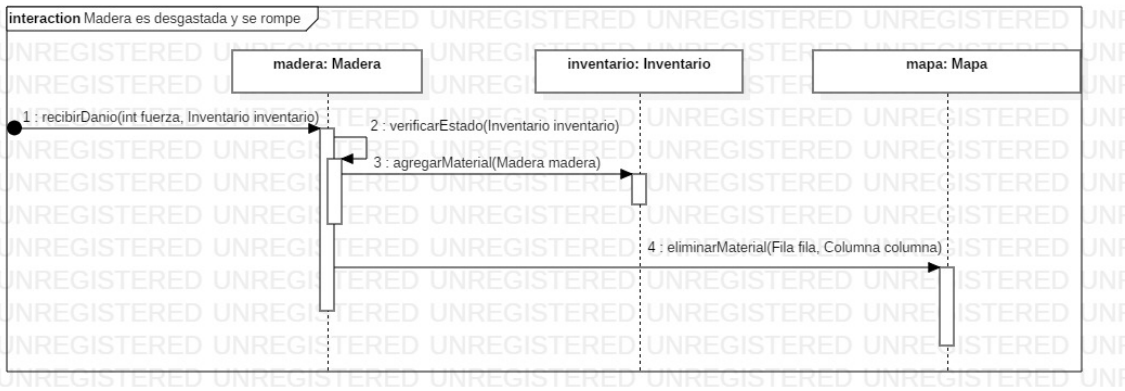

Figura 12: Muestra el proceso de eliminación de un material (particularmente madera) del juego.

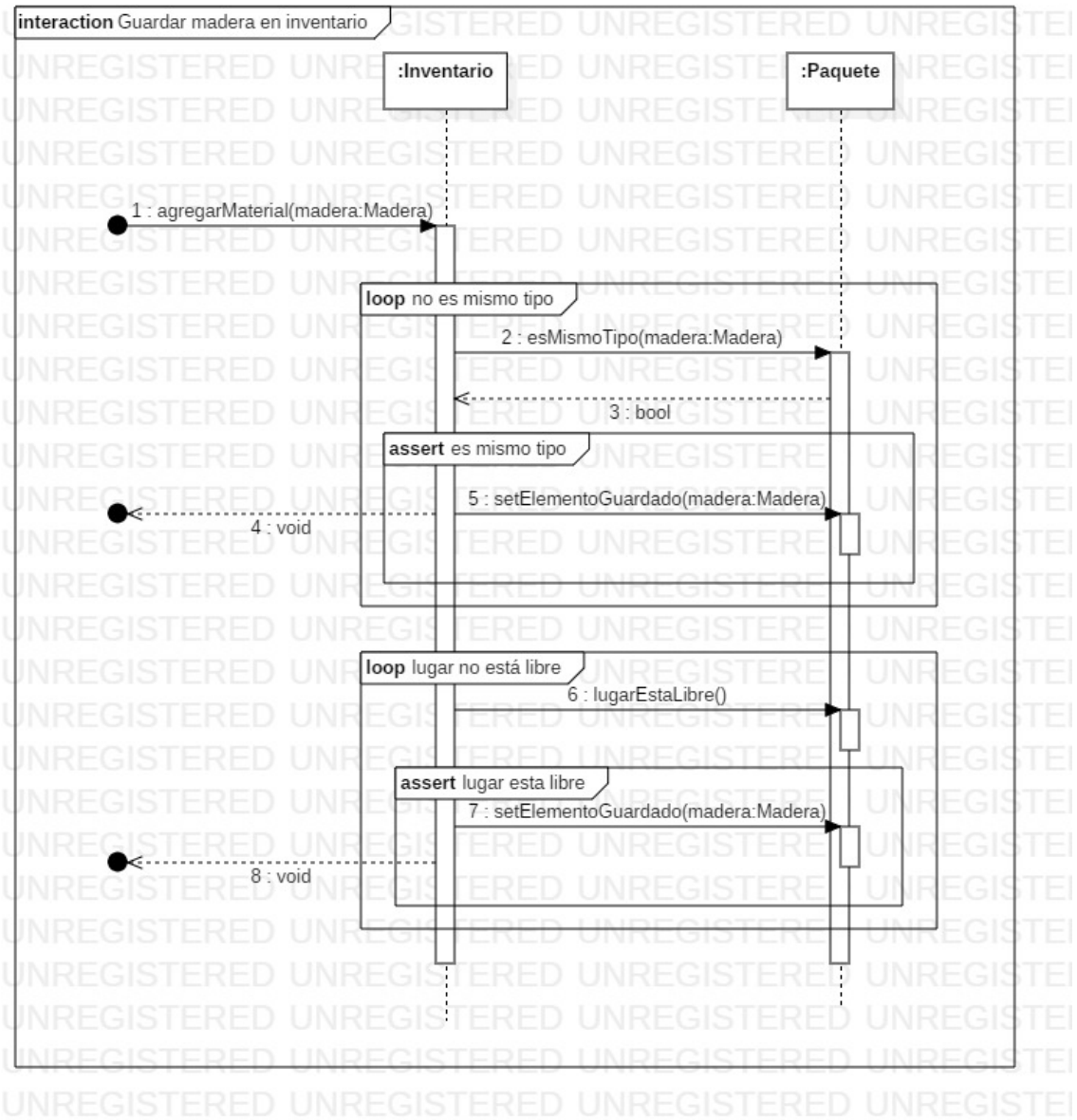

Figura 13: Muestra el proceso de guardado de un material, en el inventario, se muestra solo el caso de la madera ya que existe un método para cada material debido al uso del patrón double dispatch.

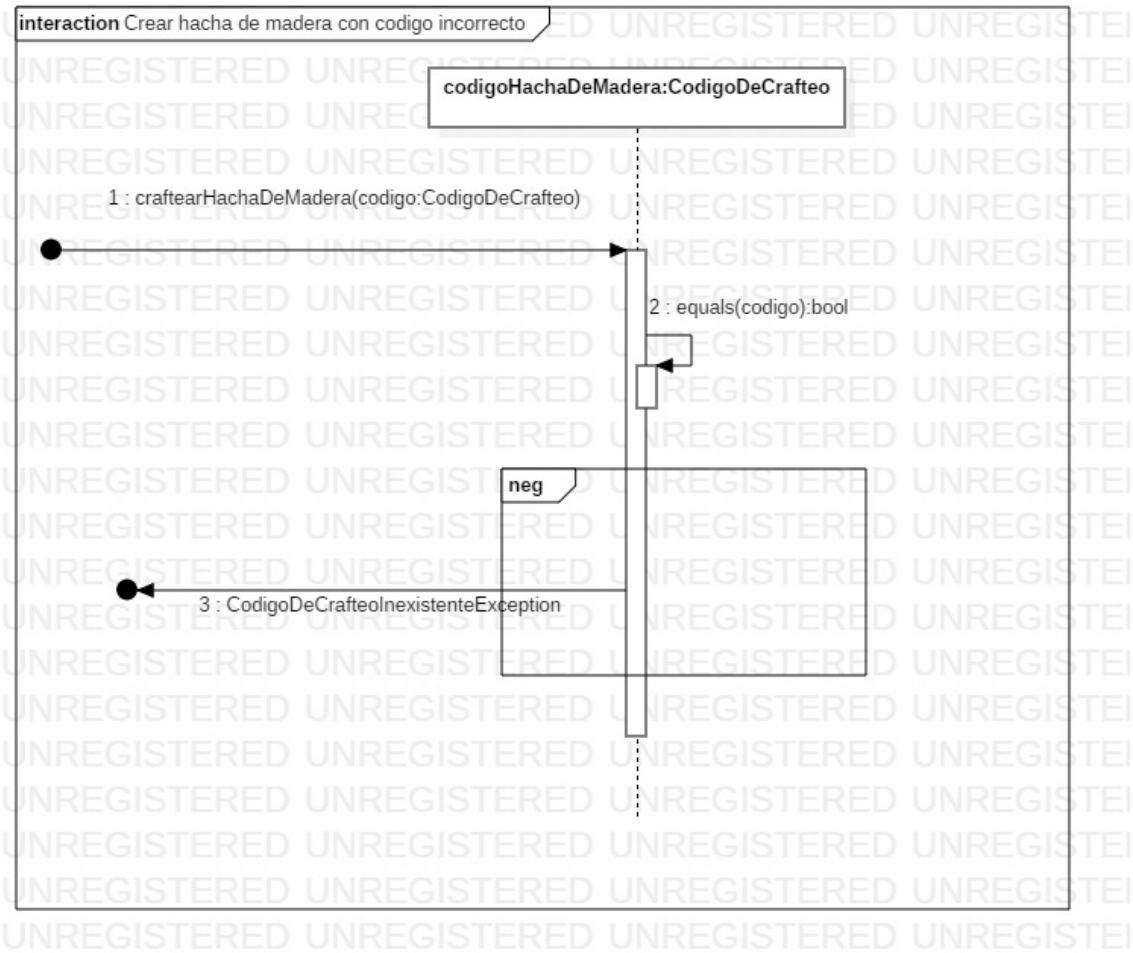

Figura 14: Muestra los procesos realizados en el caso de que se reciba un código de crafteo incorrecto.

10. Diagrama de Paquetes

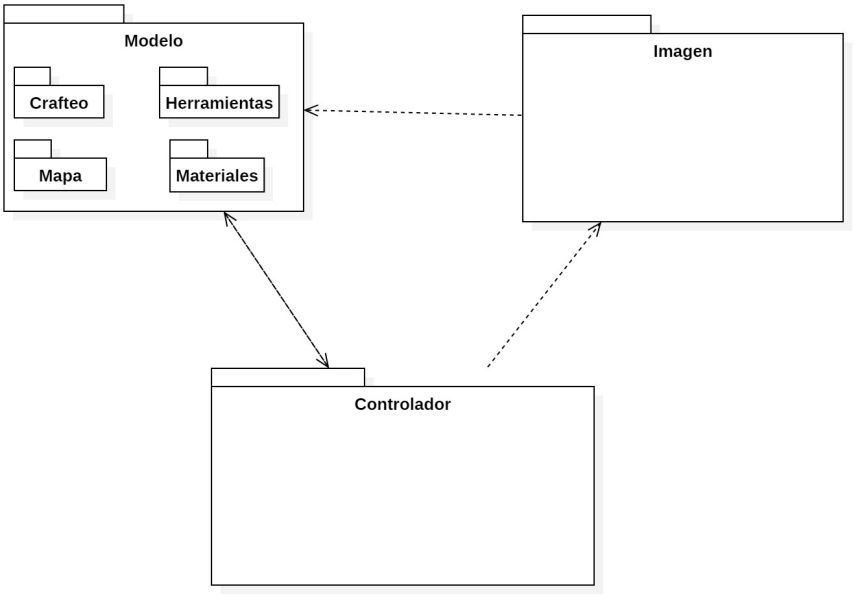

Figura 15: Diagrama de paquetes MVC