FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2 — AlgoStar

[7507/9502] Algoritmos y Programación III
Curso 2
Segundo cuatrimestre 2022

Alumno:	Número de padrón:	Email:
VERNIERI,Anita	104734	avernieri@fi.uba.ar
ZANARDI, Amparo	108360	azanardi@fi.uba.ar

Introducción

En este informe se explayará el desarrollo de la aplicación AlgoStar, para el cual utilizamos el paradigma orientado a objetos y trabajamos con el lenguaje de tipado estático Java y con las técnicas de TDD e Integración Continua.

Supuestos

Para poder llevar a cabo el juego, tuvimos que tomar algunos supuestos que a nuestro criterio tendrían sentido para el buen desarrollo del juego

- Cada jugador puede elegir solo una opción entre mover, construir o atacar en cada turno, o en su defecto puede elegir pasar turno.

- Los rangos de radio x (del crecimiento de moho, el rango del pilón, el rango de ataque, etc.) se calcula como el cuadrado que encapsula x casilleros encima, debajo y a los costados del casillero central.

- Un mapa tiene como mínimo 4 bases (Las dos donde inician los jugadores y dos más) y como máximo 8.

- El mapa crece 10 casilleros de lado por base. (Siempre es un cuadrado de área $(n * 10)^2$ casilleros con n el número de bases)

- Las bases están compuestas por un volcán en su centro y, en un radio de 3 casilleros desde este, 7 nodos de minerales.

- El Nexo Mineral recolecta 20 de mineral por turno.

Diagramas de clases

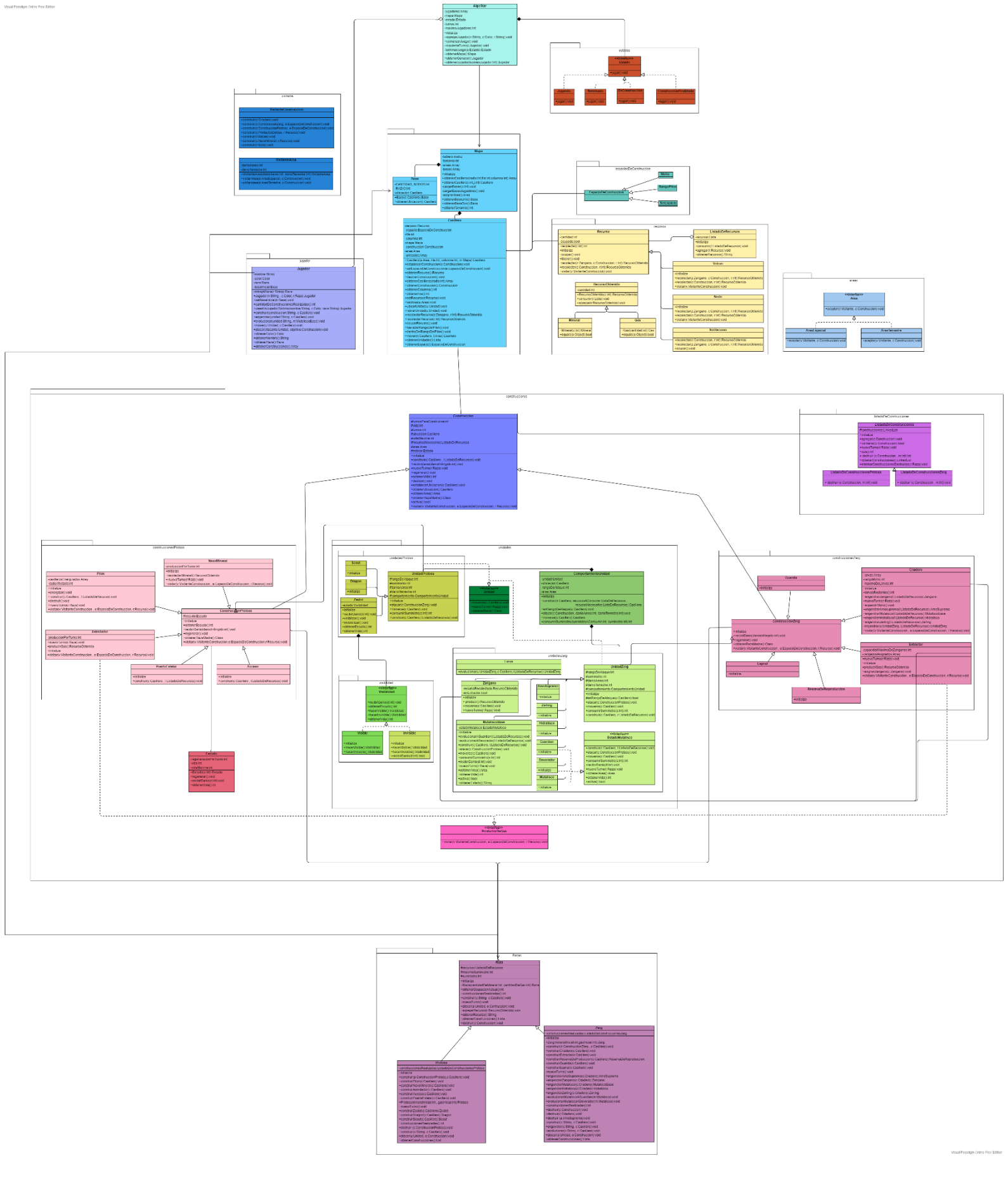


Diagrama de clases completo.

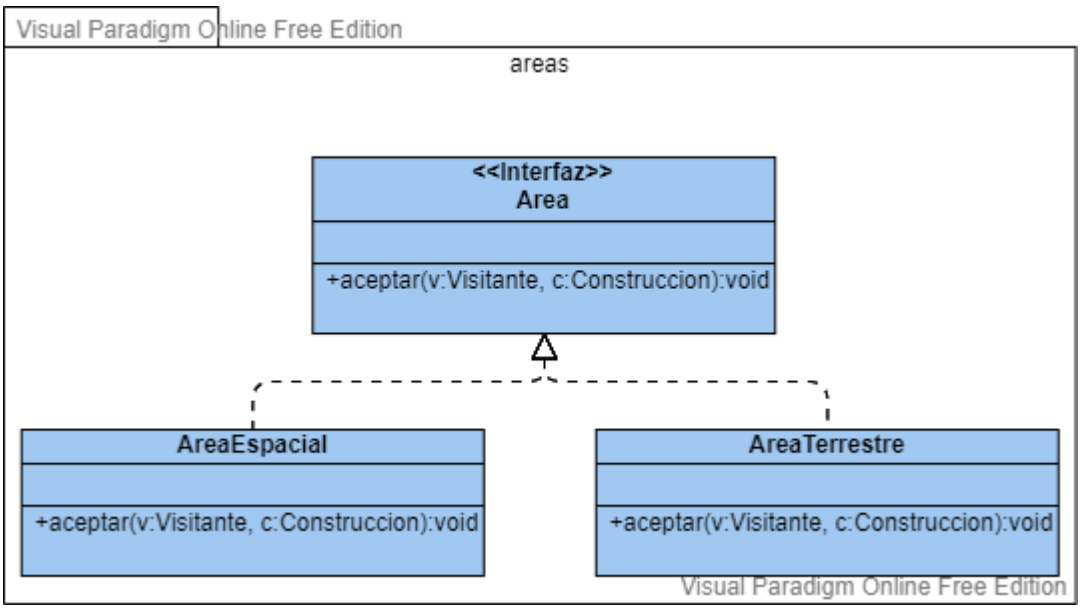


Diagrama de paquete áreas.

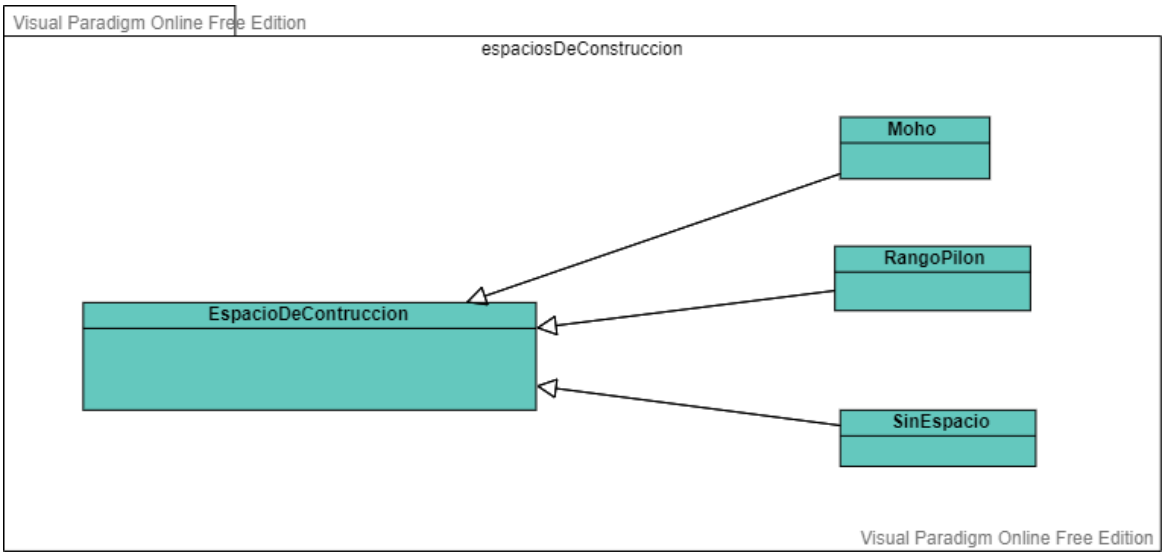


Diagrama de paquete espaciosDeConstruccion.

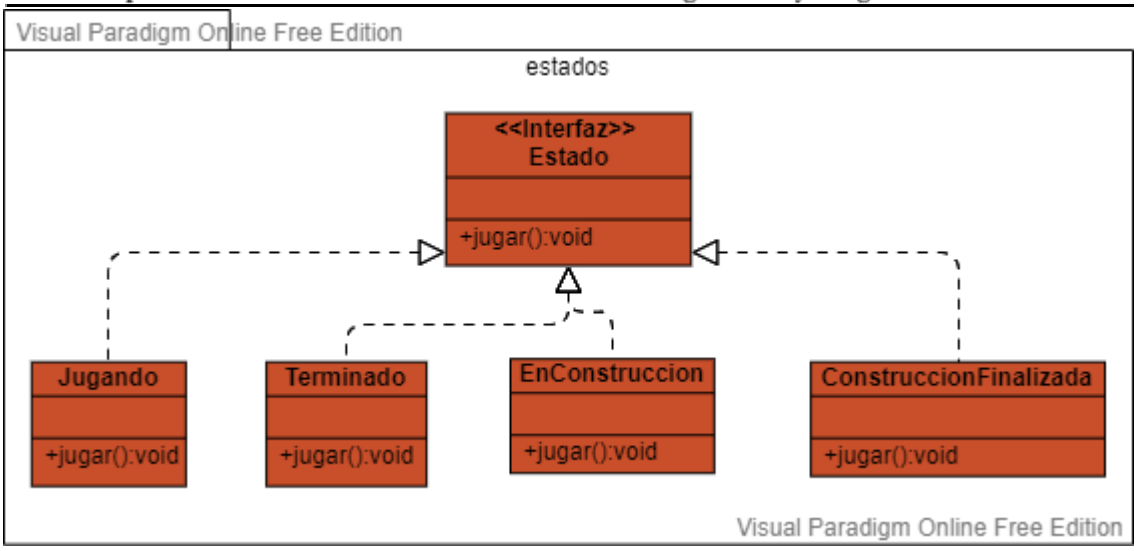


Diagrama de paquete estados.

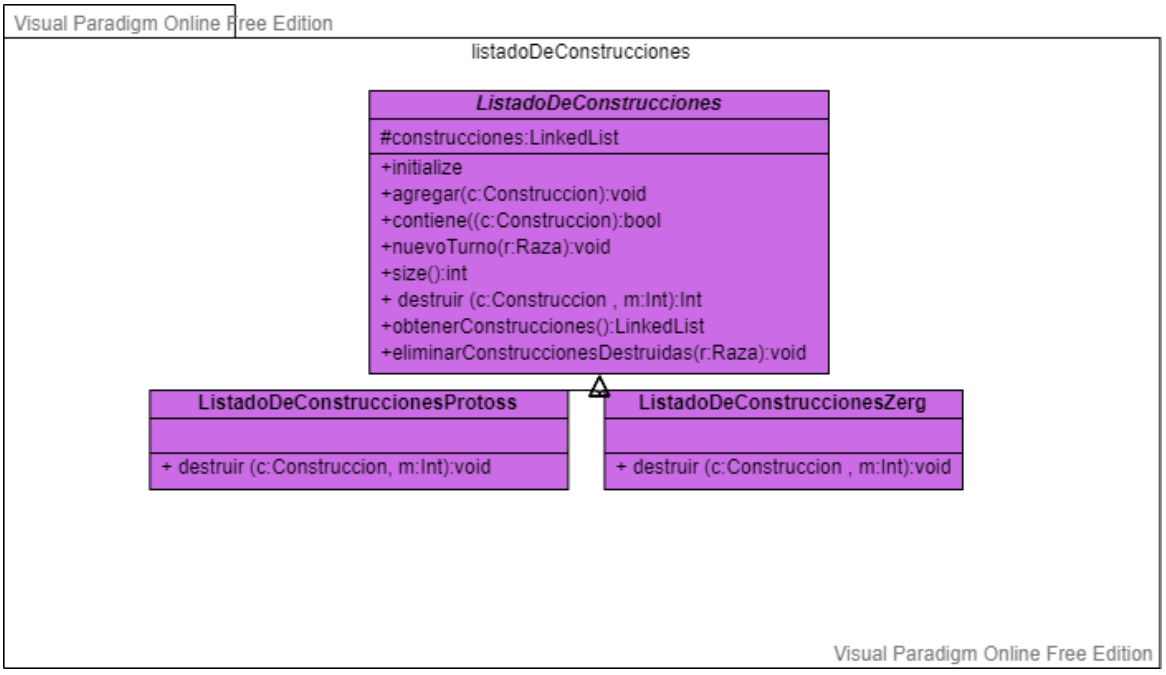


Diagrama de paquete listaDeConstrucciones.

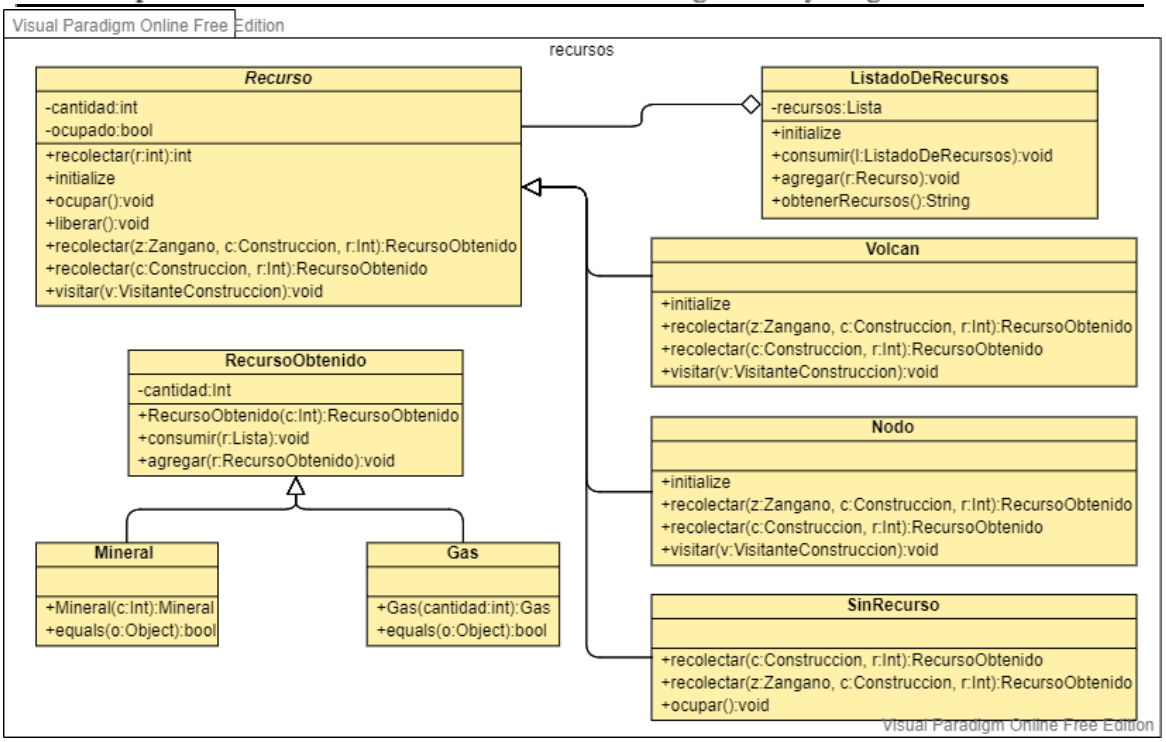


Diagrama de paquete recursos.

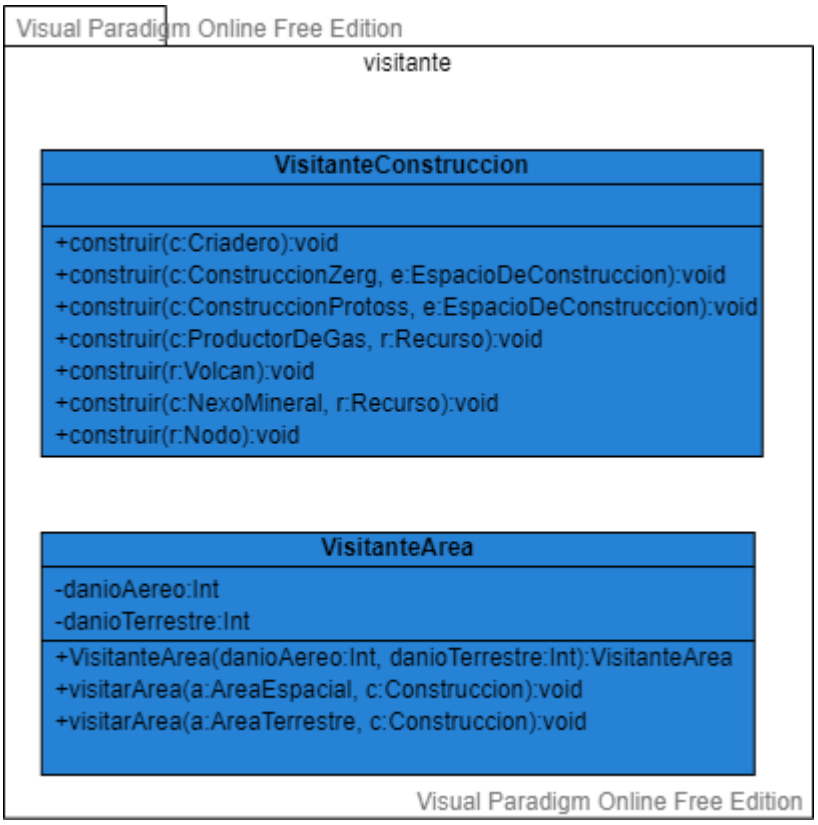


Diagrama de paquete visitante.

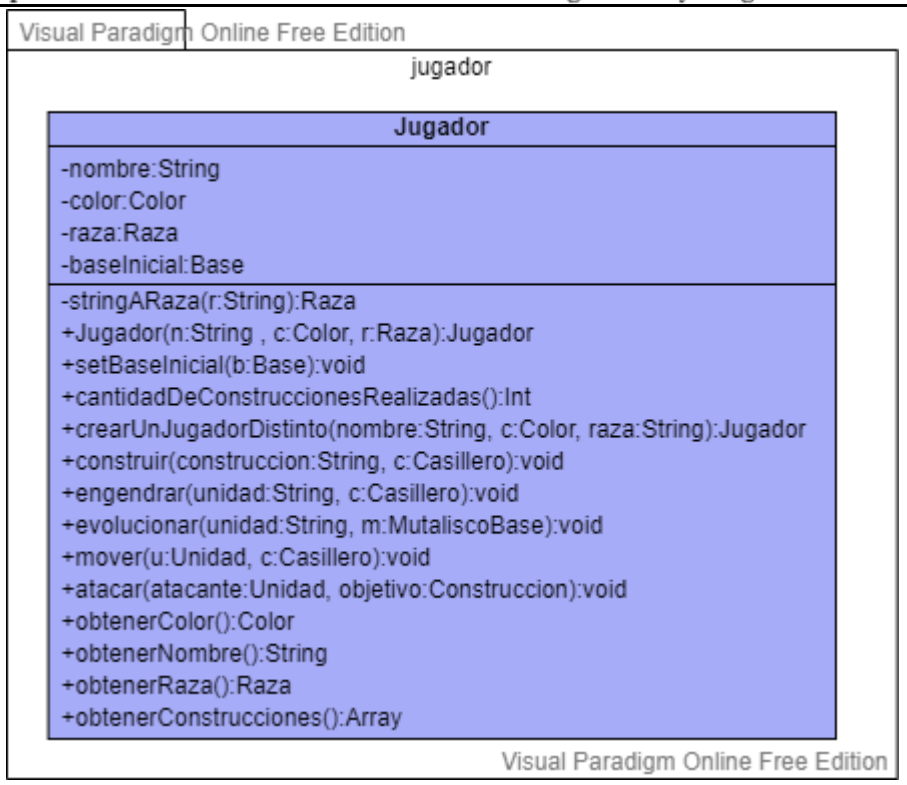


Diagrama de paquete jugador.

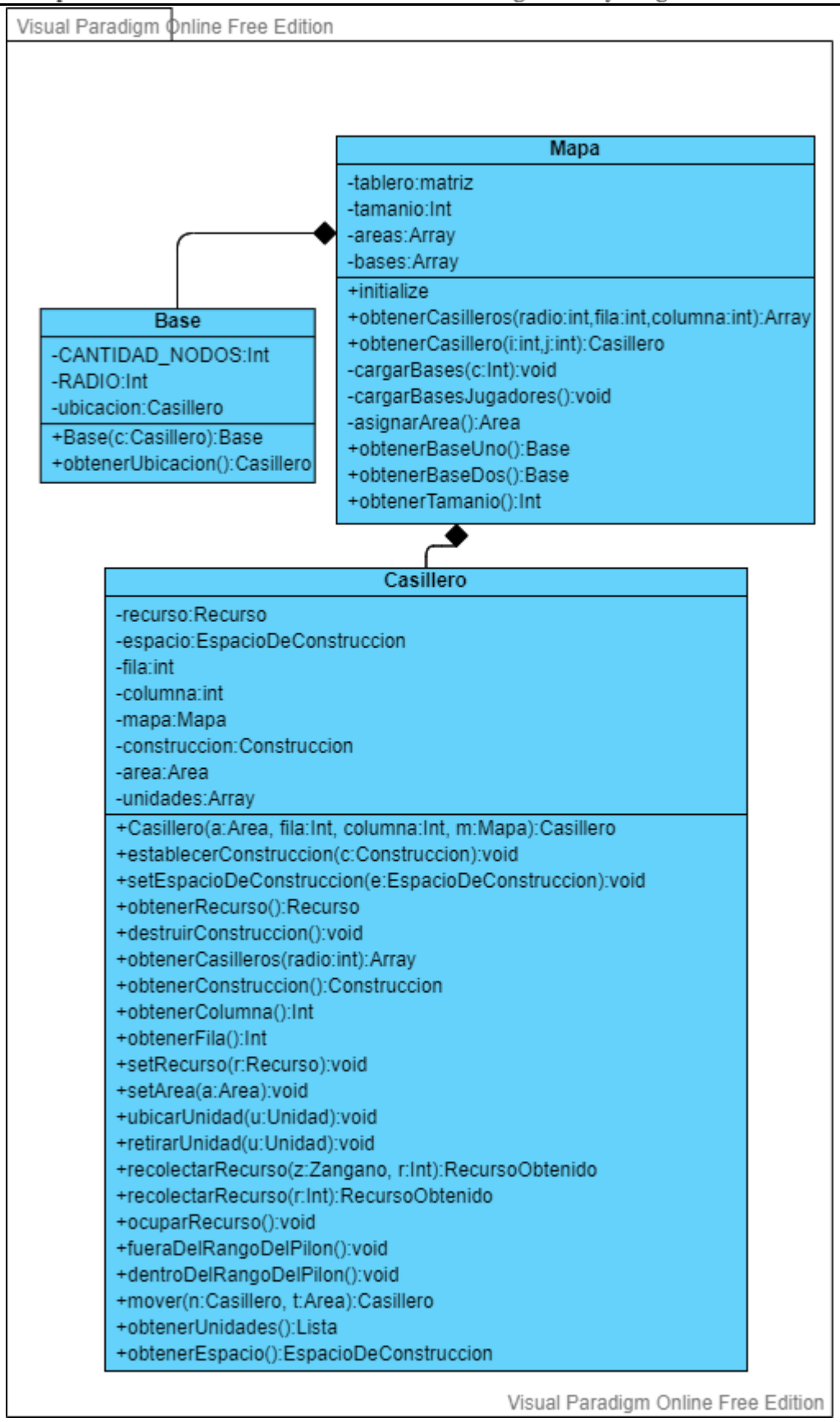


Diagrama de paquete mapa.

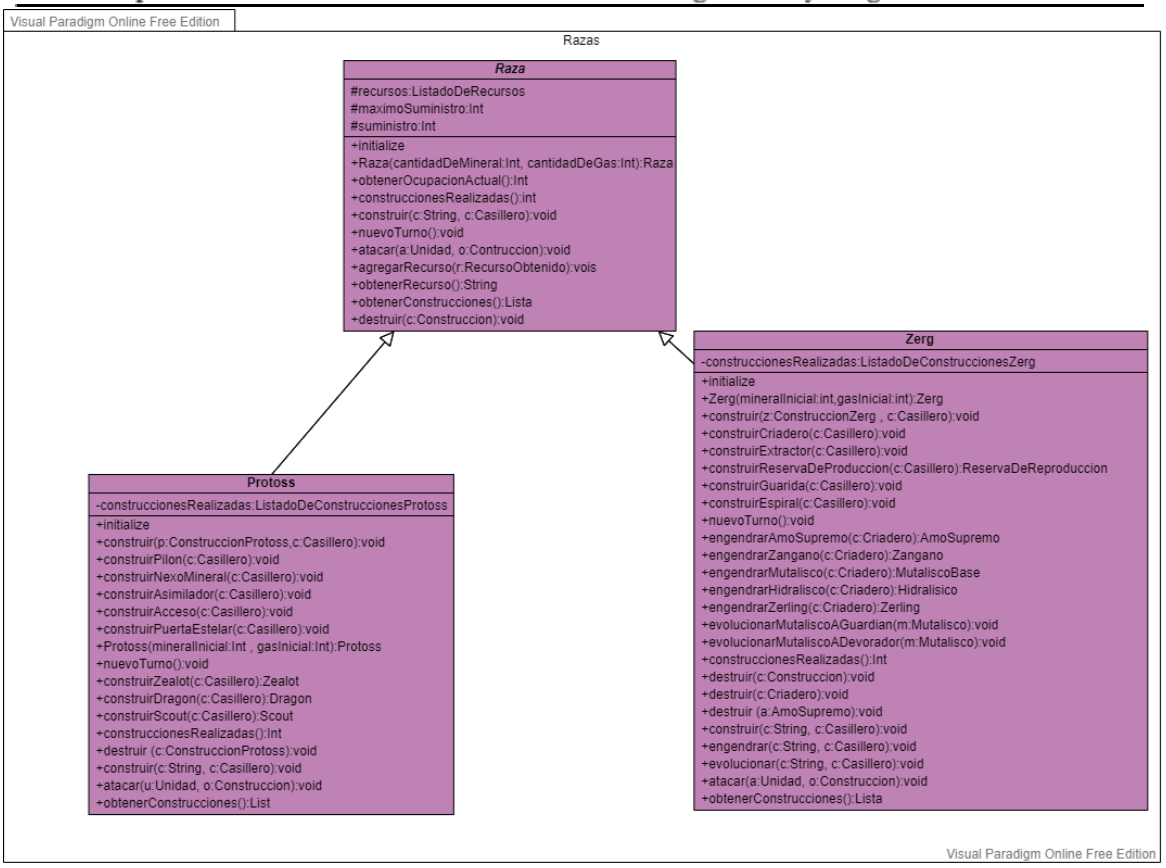


Diagrama de paquete razas.

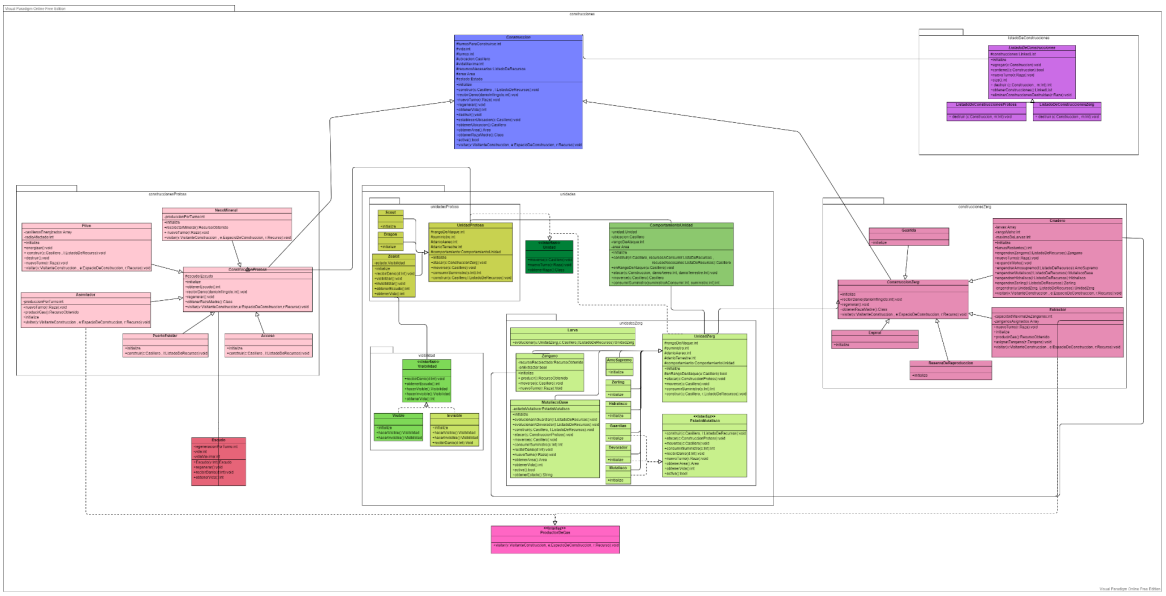


Diagrama de paquete construcciones.

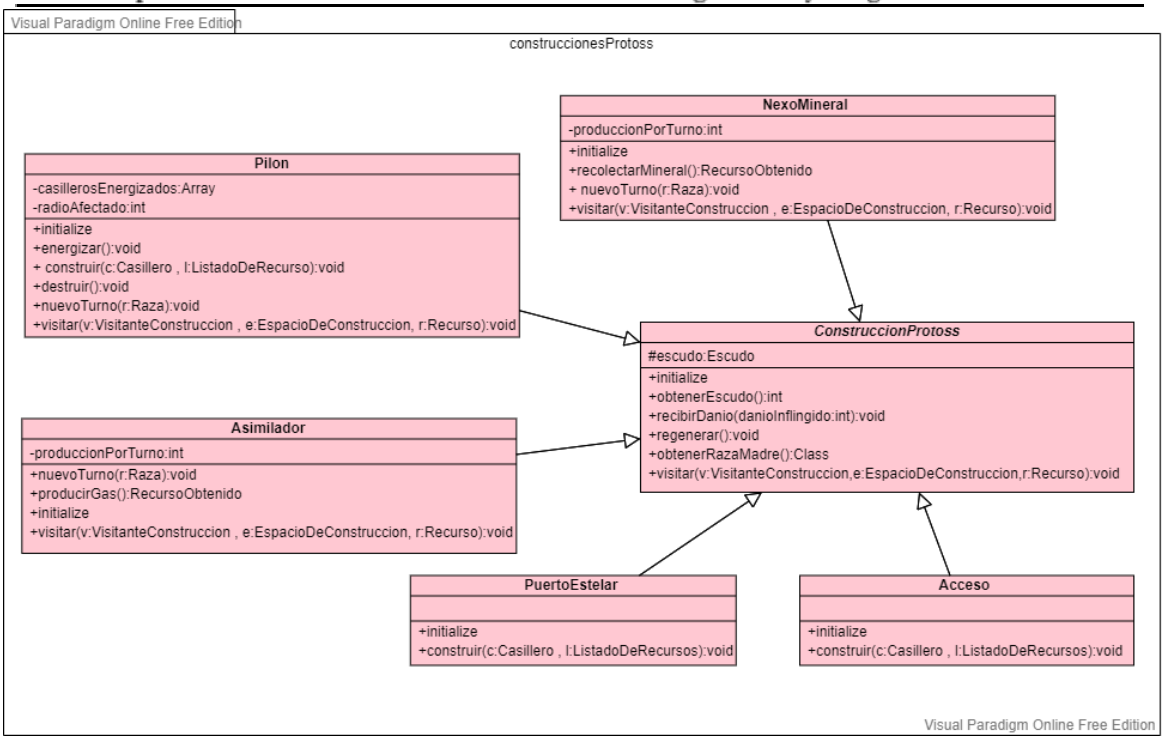


Diagrama de paquete construccionesProtoss.

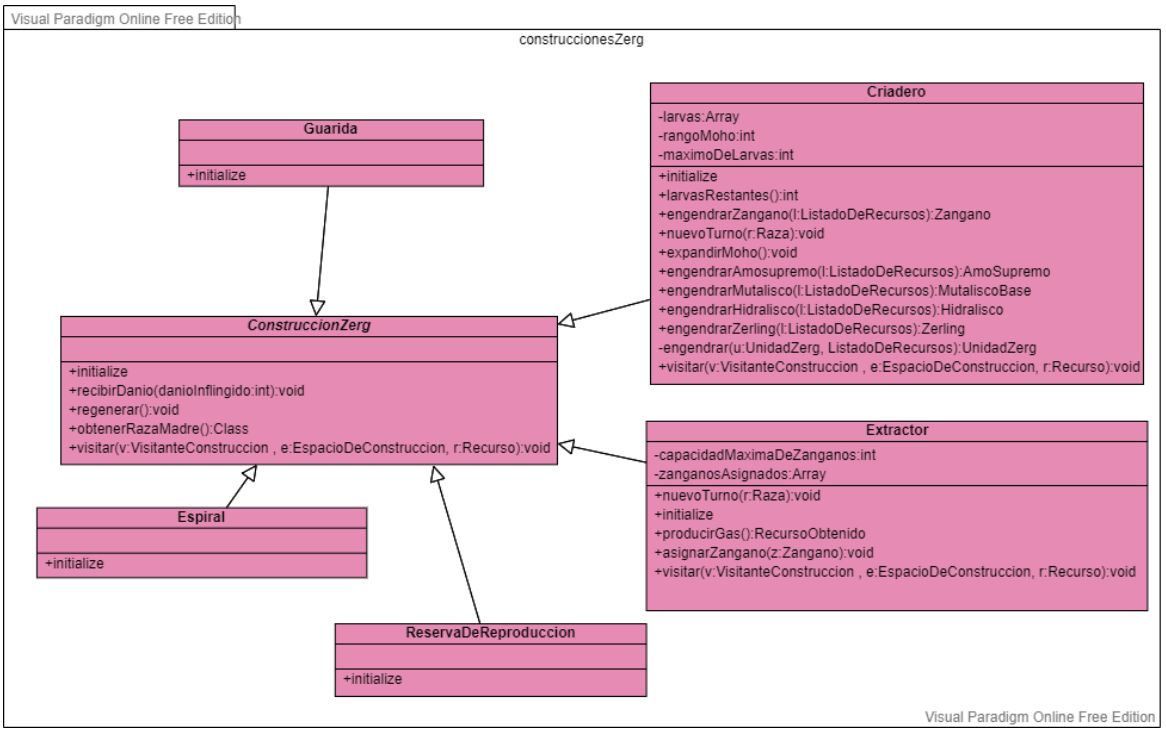


Diagrama de paquete construccionesZerg.

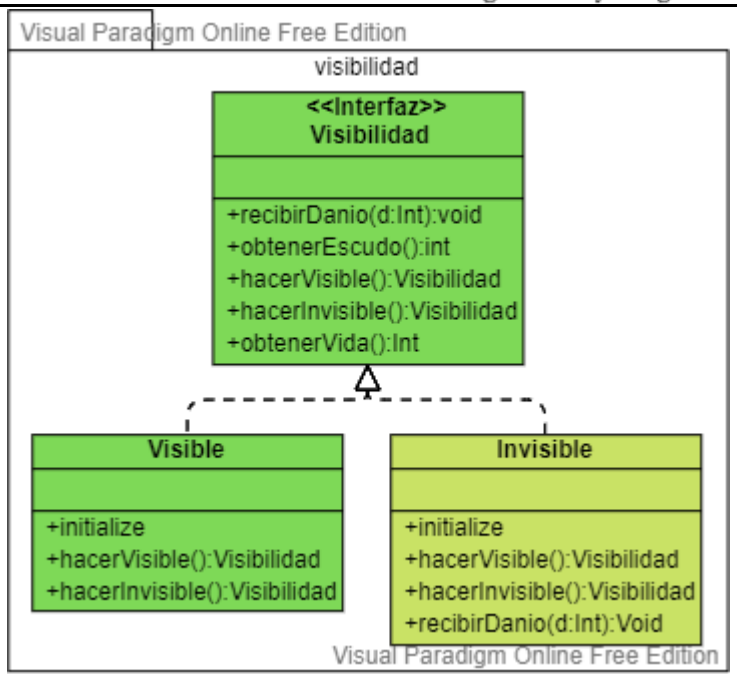


Diagrama de paquete visibilidad.

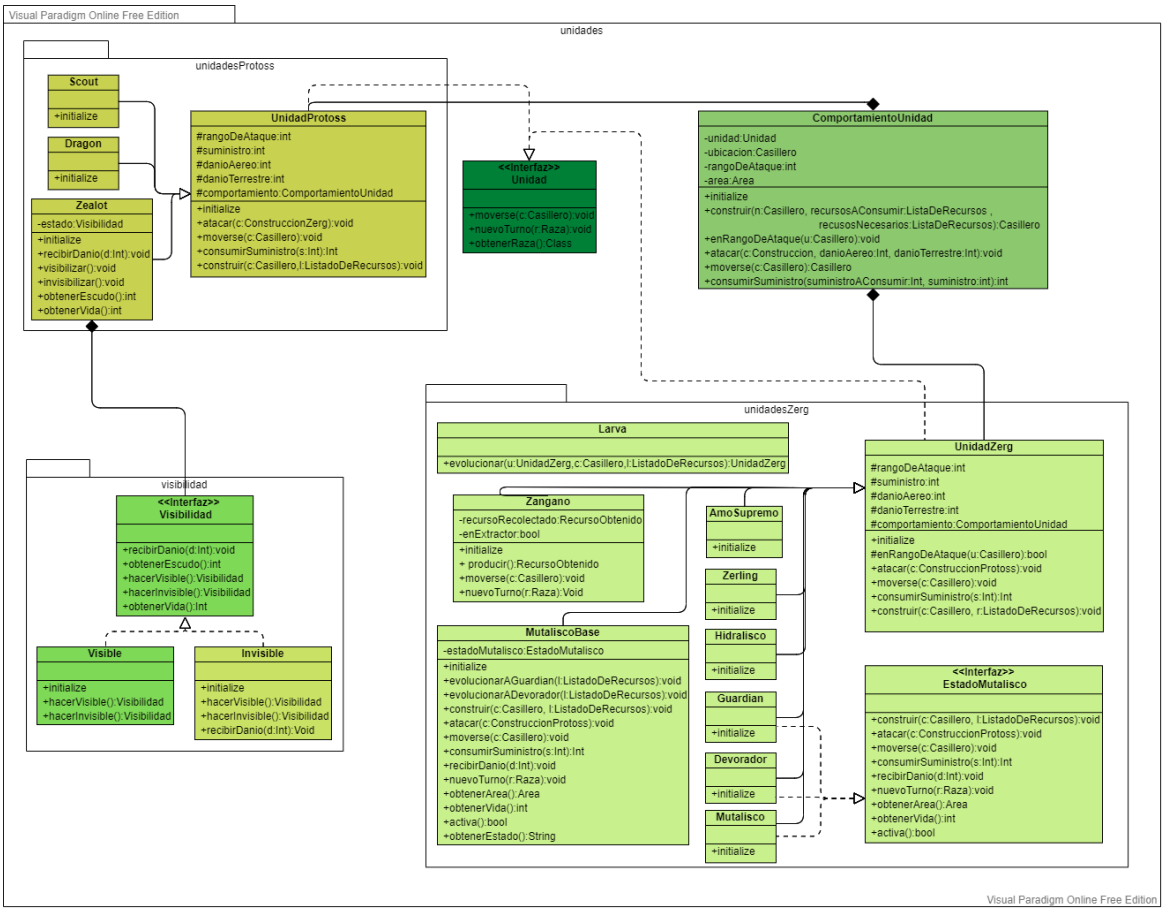


Diagrama de paquete unidades.

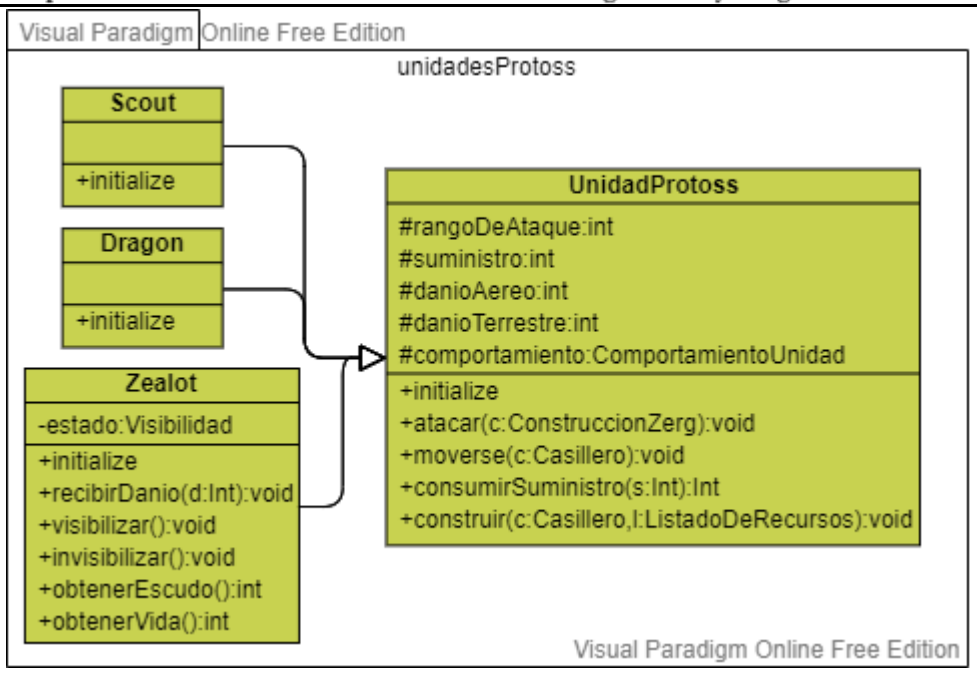


Diagrama de paquete unidadesProtoss.

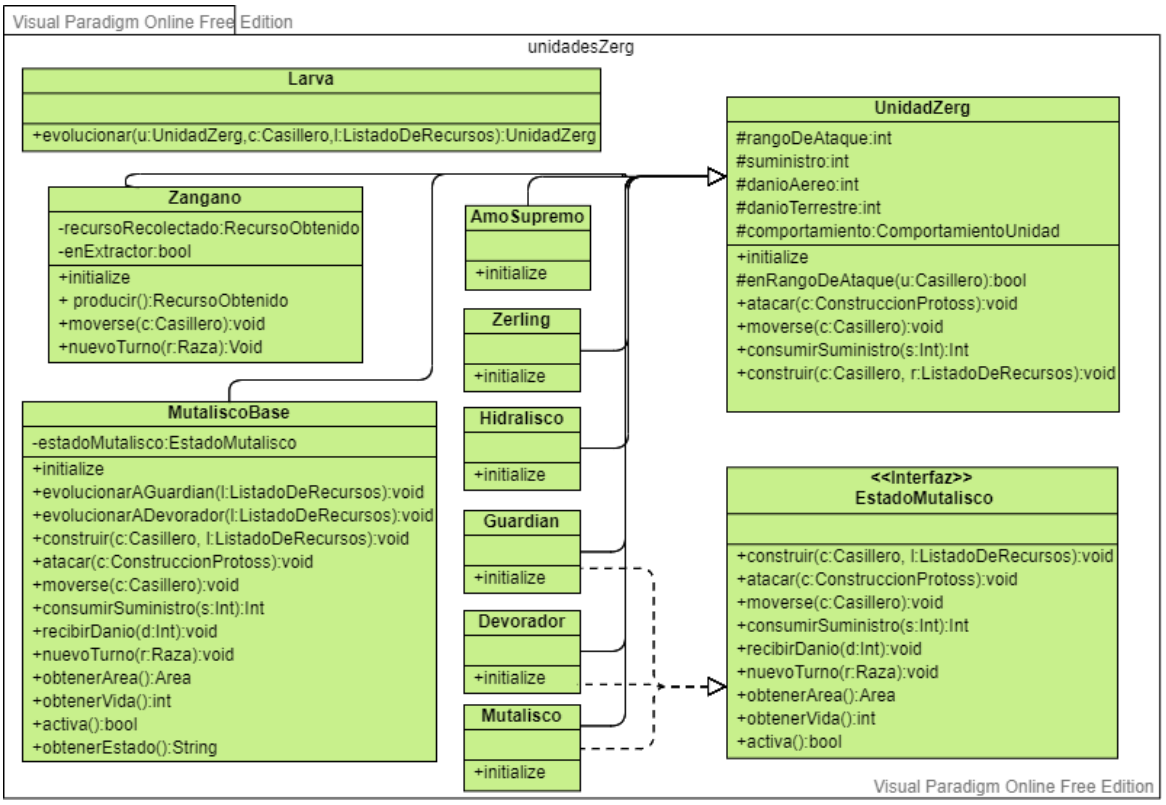


Diagrama de paquete unidadesZerg.

Diagramas de secuencia

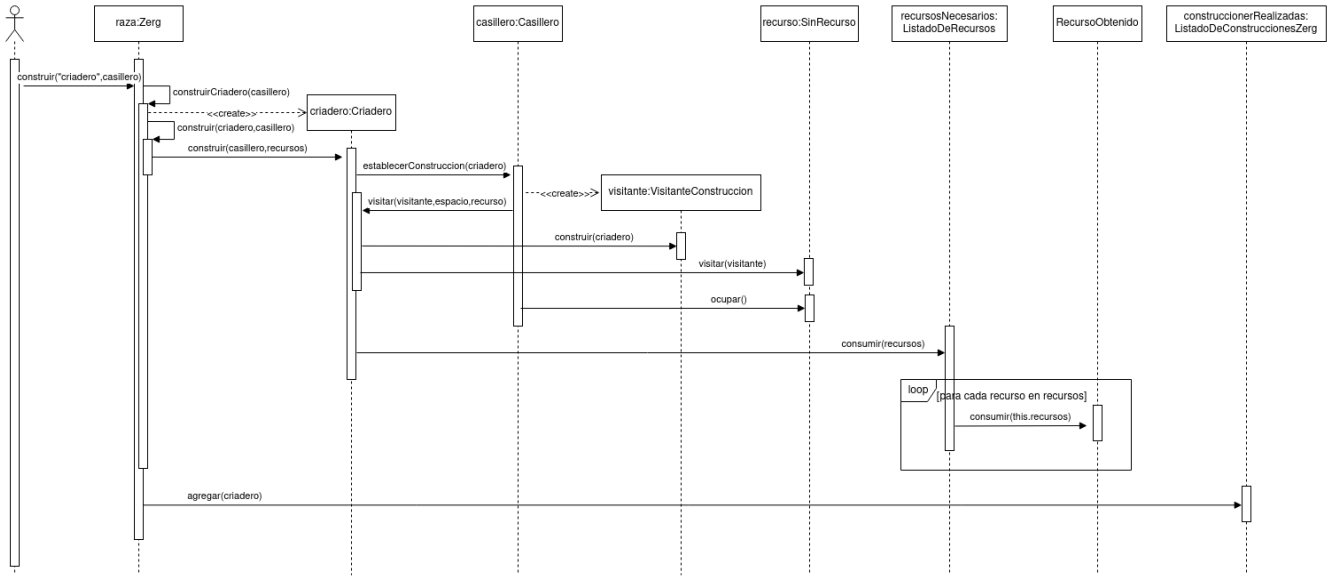


Diagrama de secuencia: Construcción exitosa de un criadero

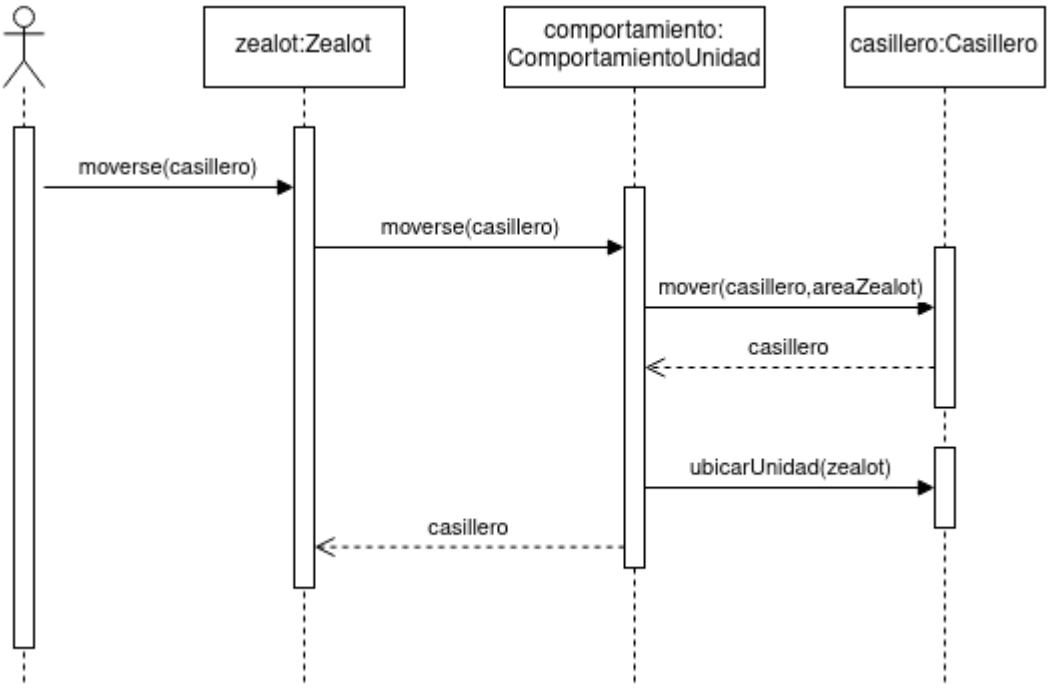


Diagrama de secuencia: Movimiento exitoso de un Zealot

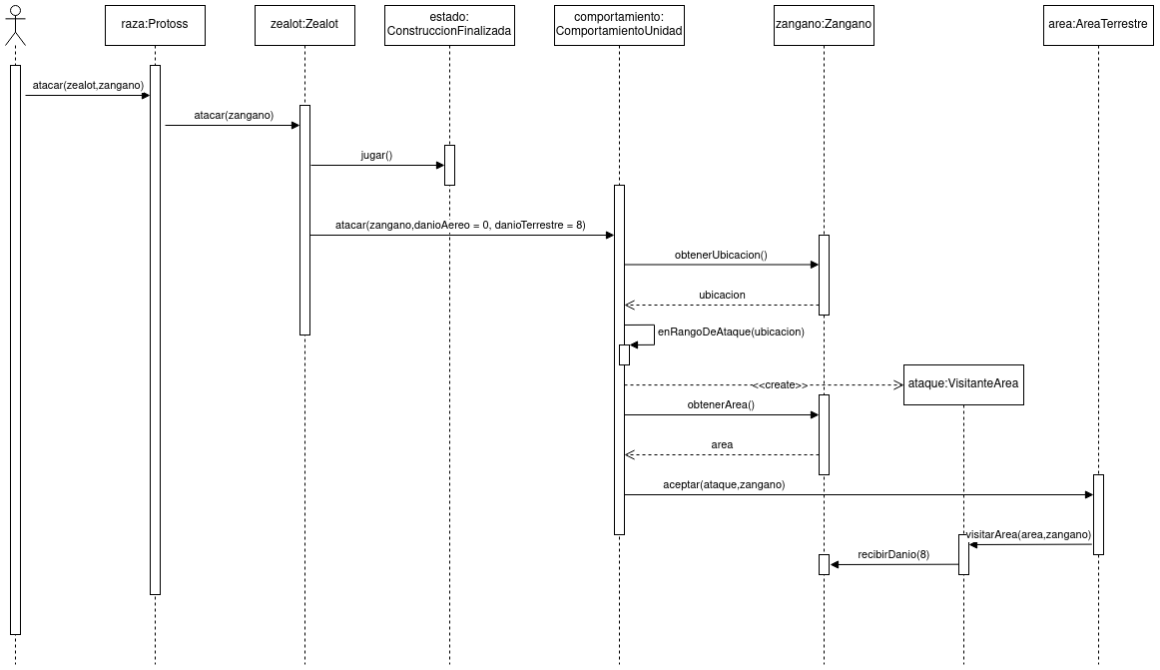


Diagrama de secuencia: ataque exitoso de Zealot a Zangano

Diagrama de paquetes

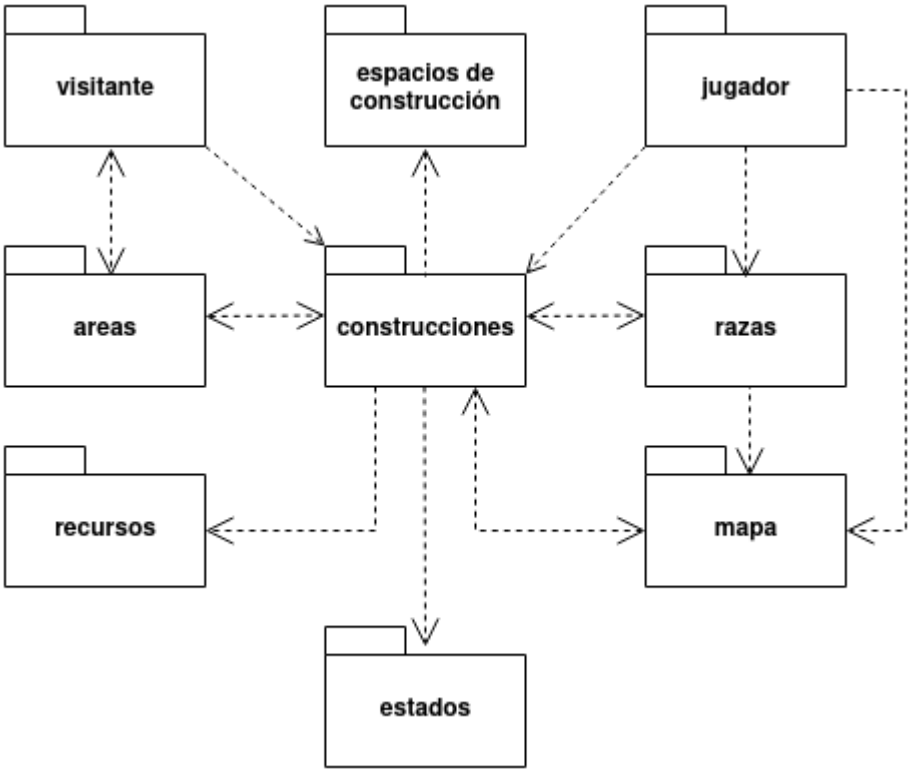
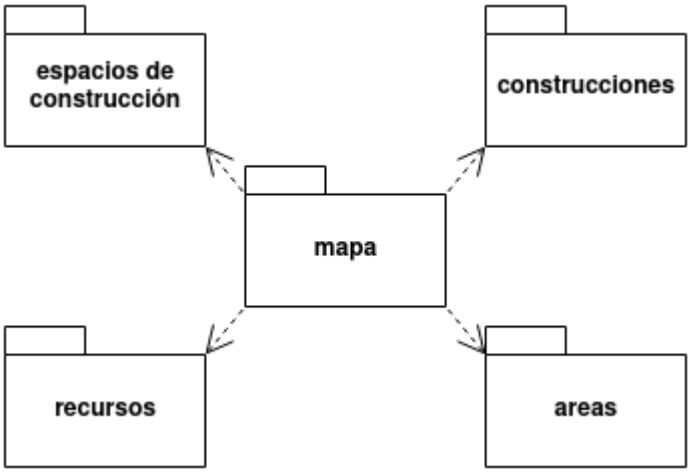
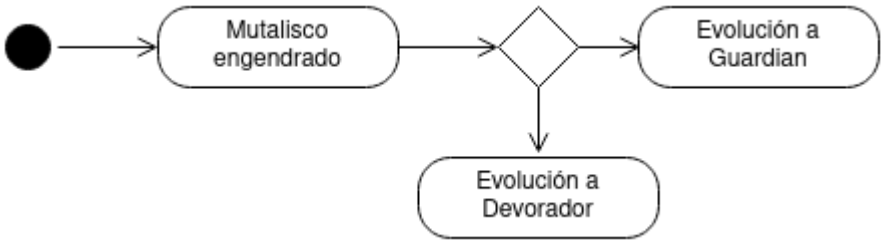


Diagrama general de paquetes

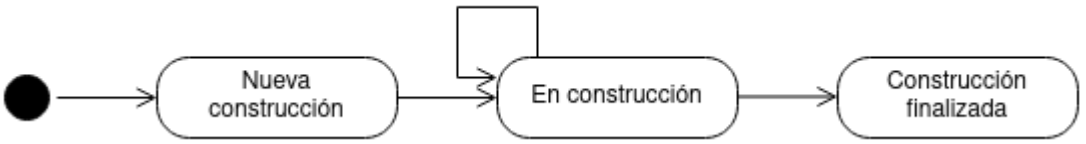


Dependencias del mapa

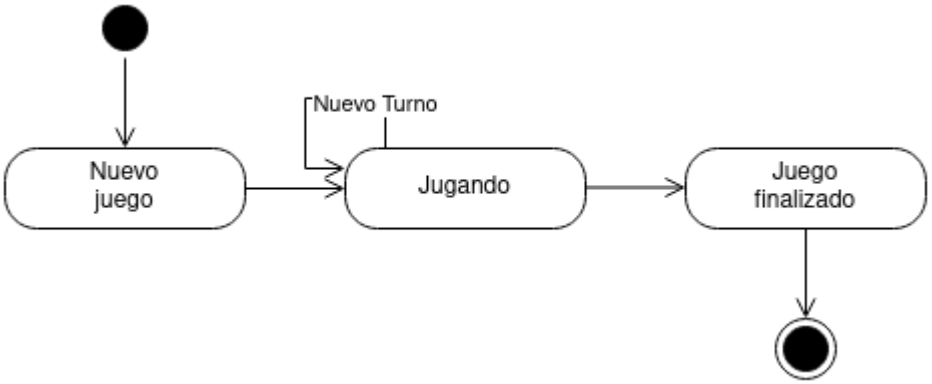
Diagramas de estado



Evoluciones del mutalisco



Estados de construcción



Estados de juego

Detalles de implementación

Consideramos que las distintas construcciones, las distintas razas, los permisos para atacar/mover/construir y la evolución del mutalisco son los puntos más conflictivos del trabajo práctico, y a continuación detallaremos qué estrategias utilizamos para resolverlos.

- Distintas razas/construcciones: usamos herencia, ya que en ambos casos hay una clara relación de “es un” entre madre e hija.
- Permisos para atacar/mover/construir: para resolver este punto usamos el patrón visitor.
- Evolución del mutalisco: para resolver este punto usamos delegación, en particular el patrón state.
- Para resolver los casos en que se necesitaba que una clase herede el comportamiento de dos clases, usamos el patrón strategy.

Excepciones

FueraDeRangoDelPilon: se tira cuando se intenta construir fuera del rango del pilón una construcción que requiere estar dentro de este para ser construida.

ConstruccionPreviaNoConstruida: se tira cuando se intenta construir una construcción sin haber construido la construcción previa requerida.

MaximoDeZanganosAsignados: se tira cuando se intenta asignar un cuarto Zángano a un Extractor.

NoSePuedeConstruir: se tira cuando una construcción no se puede construir.

NoSePuedeEngendrar: se tira cuando una unidad no se puede engendrar.

SuministroAgotado: se tira cuando se intenta construir una construcción que requiere más suministro que el disponible.

EdificioNoEstaOperativo: se tira cuando se intenta utilizar una construcción antes que pasen los turnos necesarios para construirse.

JuegoFinalizado: se tira cuando se intenta pasar el turno, peroun jugador ya se quedó sin construcciones, finalizando el juego.

DatosRepetidos: se tira cuando se intenta ingresar un segundo jugador con datos iguales a los del primer jugador.

NombreInvalido: se tira cuando se intenta anotar un jugador con un nombre con menos de 6 caracteres.

RazaInvalida: se tira cuando se intenta anotar un jugador con una raza que no es ninguna de las dos disponibles.

CasilleroSinGas: se tira cuando se intenta construir una construcción recolectora de gas sobre un casillero sin volcán.

CasilleroSinMineral: se tira cuando se intenta construir una construcción recolectora de minera sobre un casillero sin nodo.

CasilleroSinMoho: se tira cuando se intenta construir fuera en un casillero sin moho una construcción que requiere estar sobre este para ser construida.

RecursoOcupado: se tira cuando se intenta ubicar/construir una construcción sobre un recurso (volcán/nodo) cuando ya hay otra haciendo uso de este.

RecursosInsuficientes: se tira cuando se intenta comprar una construcción sin los recursos suficientes.

NoSePuedeMover: se tira cuando se intenta mover una unidad a un casillero en un área en la cual no puede estar.

ObjetivoFueraDeRango: se tira cuando se intenta atacar una construcción que está fuera del rango de ataque.

ObjetivoInvalido: se tira cuando se intenta atacar una construcción que está en un

área a la que no puede atacar.

MaximoDeJugadoresAlcanzado: se tira cuando se intenta agregar más jugadores de los permitidos en el juego;

NoExisteElJugador: se tira cuando se intenta obtener un jugador que no es parte del juego;