Diseño de motor

Cristóbal Arroyo - Juan Alegría

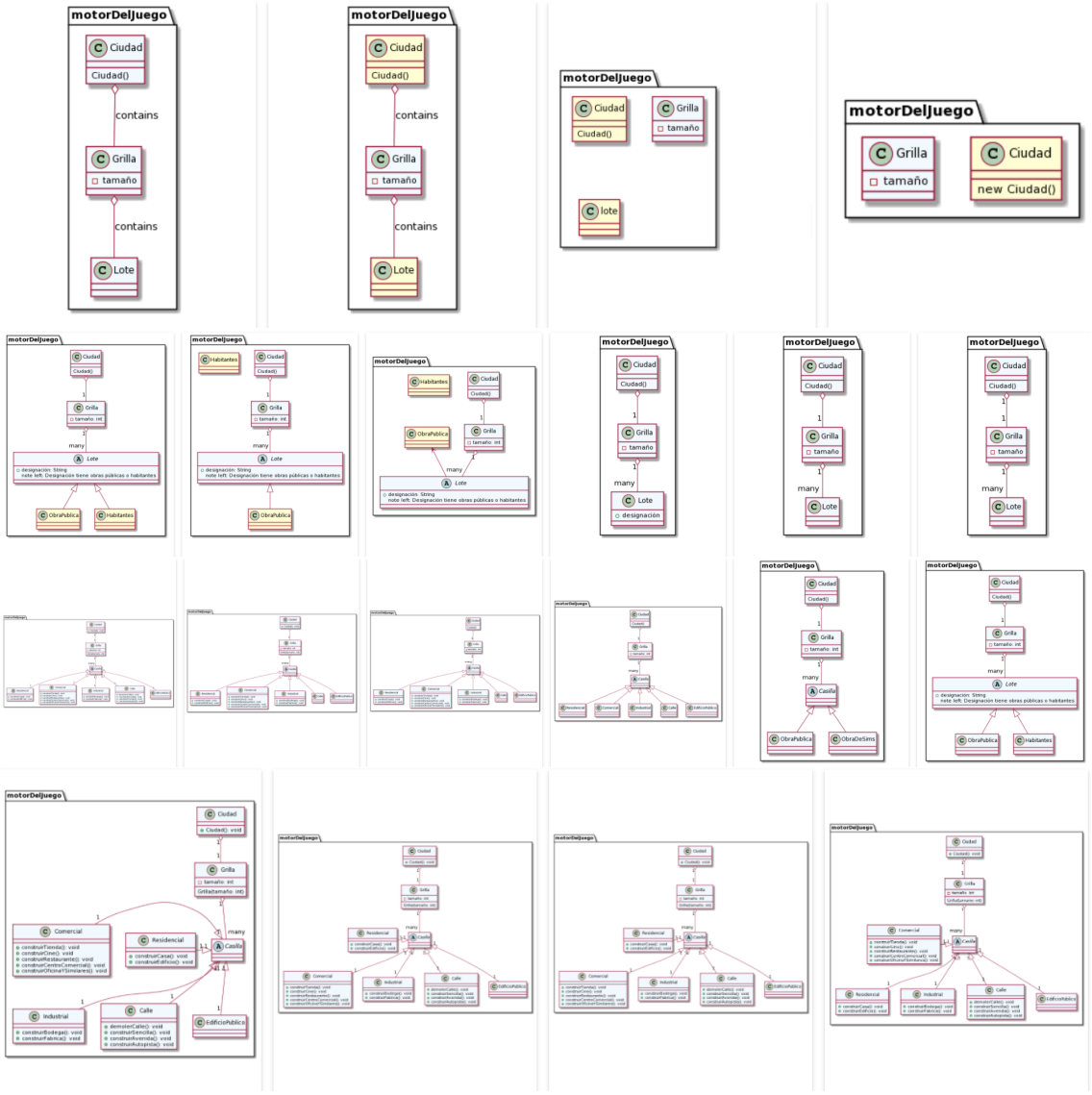
Departamento de Ingeniería de Sistemas y Computación, Universidad de los Andes

11 de marzo de 2021

Introducción

El propósito de este documento es exponer el diseño del motor de juego pensado para permitir que dado el estado de la ciudad en un día, y las decisiones que haya tomado el alcalde para ejecutar ese día, el motor permita calcular cuál será el estado de la ciudad al final de ese día. Es decir, se ha desarrollado un diseño de motor que servirá para avanzar un día en la simulación.

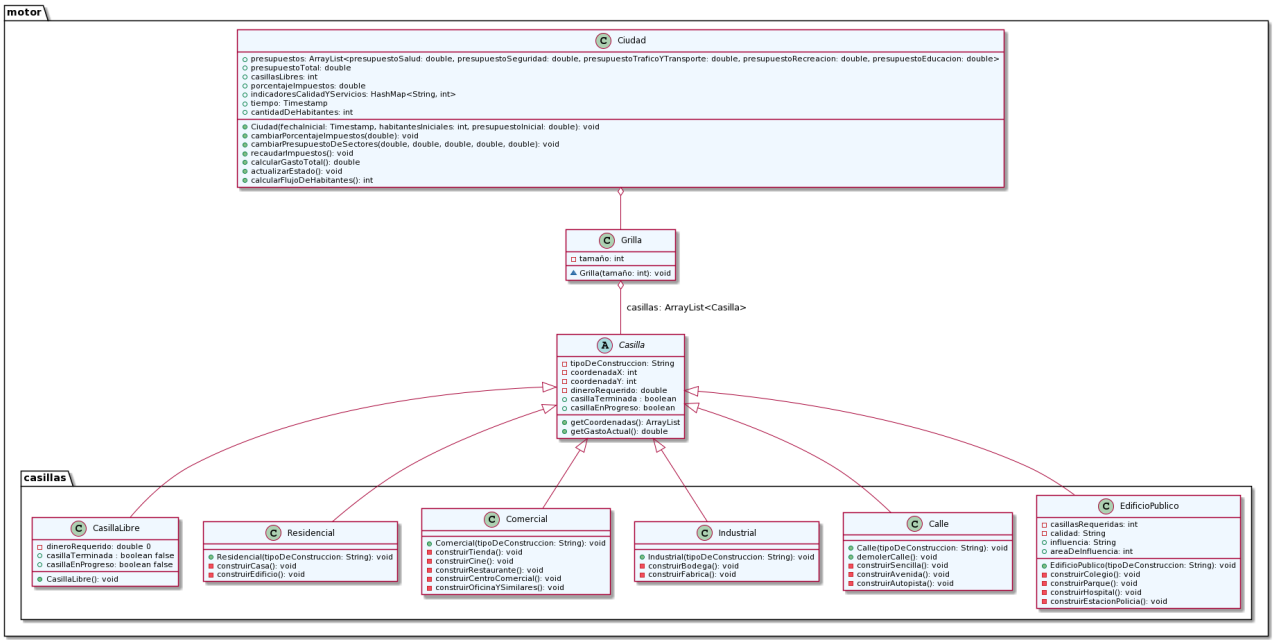
En primer lugar se iteró varias veces sobre un diagrama de clases de todo el mundo del problema, como se puede visualizar a continuación:



Resumiendo este proceso iterativo, se comenzó con los tres elementos más básicos del motor del juego que serían la ciudad, la grilla, y sus casillas. A partir de esto, poco a poco se fueron añadiendo sus atributos correspondientes y en algún momento se decidió crear subclases para algunos de ellos. Un ejemplo de esto ocurrió con las casillas, donde se tenía en primer lugar un atributo de tipo de casilla que fue cambiado por una clase abstracta y sus hijos. Finalmente, el diseño se completó luego de reorganizarlo y crear todos los métodos necesarios para suplir las necesidades requeridas para avanzar un día en la simulación.

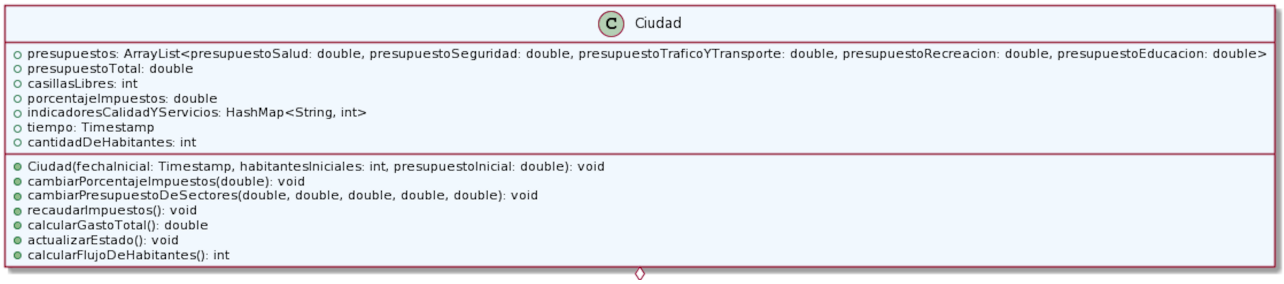
Diagrama de clases del motor

Como resultado se obtuvo el siguiente diagrama de clases, un diagrama que mantiene la idea principal pero tiene un paquete de casillas, dada la amplia cantidad de clases requerida, y múltiples métodos que serán necesarios para el funcionamiento del juego:



Clases del motor explicadas a detalle

Ciudad: esta clase es la encargada de inicializar el mundo de juego.



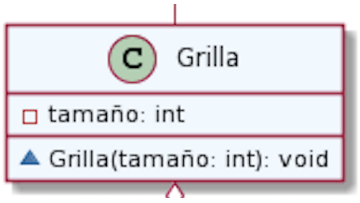
- **Atributos:**
  - presupuestos: es un arreglo que contiene el presupuesto de cada uno de los sectores en la ciudad.
  - presupuestoTotal: es la sumatoria de presupuestos.

- casillasLibres: es el conteo de casillas libres disponibles en el mapa, cada vez que una casilla se ocupa o desocupa el valor es cambiado.
- porcentajeImpuestos: un número decimal que representa el porcentaje de impuestos en la ciudad.
- indicadoresCalidadYServicios: es un mapa que contiene como llave un indicador y como valor un número indicando el nivel correspondiente. Ejemplo: la calidad del tráfico, el nivel de criminalidad, las ofertas de entretenimiento y la calidad de los servicios de salud.
- tiempo: es una marca de tiempo que guarda el valor del momento en el que se actualizó el último estado
- cantidadDeHabitantes: es el número de habitantes que se encuentran en la ciudad.

◇ **Métodos:**

- Ciudad(): es el constructor de la clase Ciudad que recibe la fecha en la que la ciudad debió fundarse, el número de habitantes iniciales y el presupuesto inicial.
- cambiarPorcentajeImpuestos(): este método recibe un double para actualizar con este valor el porcentaje de impuestos de la ciudad.
- cambiarPresupuestoDeSectores(): se recibe por parámetro todos los presupuestos de los sectores para cambiar sus valores.
- recaudarImpuestos(): este método recauda los impuestos pendientes en cada una de las casillas. Si el método getGastoActual retorna un dinero requerido que es positivo, se utiliza en este método, de lo contrario se ignora.
- calcularGastoTotal(): este método calcula el gasto total en cada una de las casillas. Si el método getGastoActual retorna un dinero requerido que es negativo, se utiliza en este método, de lo contrario se ignora.
- calcularFlujoDeHabitantes(): este método calcula el flujo de habitantes de acuerdo a todos los factores de interés o desinterés.
- actualizarEstado(): actualizar estado es un método que se ejecuta cada cierto tiempo y ejecuta la mayoría de los métodos que se encuentran en la clase Ciudad. Actualiza el atributo tiempo, recauda impuestos, calcula gastos, actualiza el presupuesto de cada sector y el total, calcula el flujo de habitantes, y actualiza la cantidad de estos. Cada vez que haya pasado un día, este método avisa al jugador generando un reporte del estado final de la ciudad en ese día.

Grilla: esta clase es la encargada de almacenar el arreglo de casillas de la ciudad.



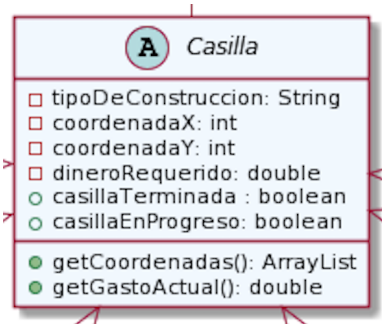
◇ **Atributos:**

- tamaño: al inicializar la ciudad este atributo es configurado y no volverá a cambiar.

◇ **Métodos:**

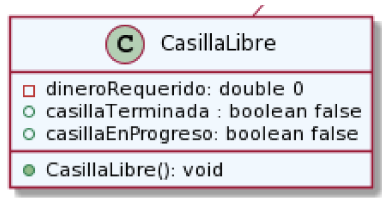
- Grilla(): este es el constructor de grilla, es llamado por el constructor de Ciudad y recibe un tamaño por parámetro.

Casilla: clase abstracta de las casillas.



- **Atributos:**
  - tipoDeConstruccion: recibe el tipo de construcción y será usado por cada uno de las clases hijo de Casilla para la creación de la edificación por tipo.
  - coordenadaX: guarda la posición de la columna de la casilla, o una coordenada de acuerdo al sistema de medidas del motor.
  - coordenadaY: guarda la posición de la fila de la casilla, o una coordenada de acuerdo al sistema de medidas del motor.
  - dineroRequerido: este atributo acumula las deudas o ganancias pendientes y se actualiza en los constructores de los subhijos de casilla o en el método actualizarEstado de la Ciudad.
  - casillaTerminada: si una casilla terminó de construirse este valor cambiaría a true. Es utilizado para conocer en que casilla se puede construir una nueva edificación.
  - casillaEnProgreso: si una casilla está en progreso de construcción este valor sería true. Es utilizado para conocer en que casilla se puede construir una nueva edificación.
- **Métodos:**
  - getCoordenadas(): retorna un arreglo con las coordenadas de la casillas.
  - getGastoActual(): retorna el valor de dineroRequerido.

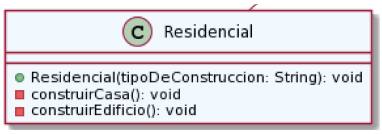
CasillaLibre: esta clase es usada cuando el motor inicializa la ciudad, para rellenar todo el arreglo de casillas de la grilla con casillas libres. Cuando una casilla de otro tipo va a ser creada, la variable que contiene una de estas casillas debe dar sus coordenadas para ser eliminada y reemplazada por cualquiera de los demás hijos de Casilla.



- **Atributos:**
  - dineroRequerido: al dinero requerido de Casilla se le asigna el valor constante de 0, ya que estas casillas no deberían implicar un gasto ni ganancia.
  - casillaTerminada: mantiene el valor de false.
  - casillaEnProgreso: mantiene el valor de false.

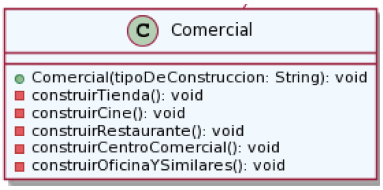
- ◇ **Métodos:**
  - CasillaLibre(): este constructor es llamado en un ciclo durante la creación de la clase Grilla. De acuerdo al tamaño de la grilla, se forma un ciclo que va asignando las coordenadas X y Y de cada casilla, manteniendo en null el tipo de construcción.

Residencial: subclase de Casilla de tipo Residencial.



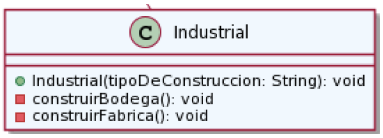
- ◇ **Métodos:**
  - Residencial(): constructor de la clase Residencial que recibe el tipo de construcción para llamar a la función construirCasa() o construirEdificio().
  - construirCasa(): construye una casa.
  - construirEdificio(): construye un edificio.

Comercial: subclase de Casilla de tipo Comercial.



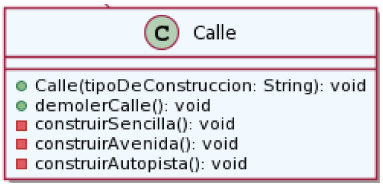
- ◇ **Métodos:**
  - Comercial(): constructor de la clase Comercial que recibe el tipo de construcción para llamar a la función correspondiente de construcción.
  - contruirTienda(): construye una tienda.
  - construirCine(): contruye un cine.
  - construirRestaurante(): construye un restaurante.
  - construirCentroComercial(): construye un centro comercial.
  - contruirOficinaYSimilares(): construye una oficina y edificaciones similares.

Industrial: subclase de Casilla de tipo Industrial.



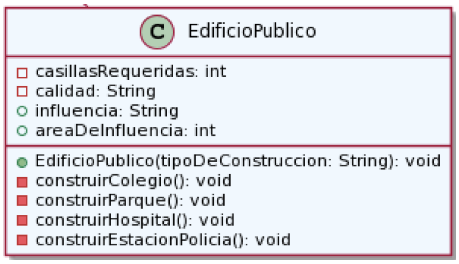
- ◇ **Métodos:**
  - Industrial(): constructor de la clase Industrial que recibe el tipo de construcción para llamar a la función correspondiente de construcción.
  - construirBodega(): construye una bodega.
  - construirFabrica(): construye una fábrica.

Calle: subclase de Casilla de tipo Calle.



- **Métodos:**
  - Calle(): constructor de la clase Calle que recibe el tipo de construcción para llamar a la función correspondiente de construcción.
  - demolerCalle(): reemplaza esta casilla de tipo Calle con una CasillaLibre. Si esta calle es de un tipo complejo como avenida o autopista, hace la misma acción con las casillas adyacentes y del mismo tipo a esta calle.
  - construirSencilla(): construye una calle sencilla.
  - construirAvenida(): construye una avenida, y de acuerdo a las coordenadas actuales se verifica y construye en las casillas adyacentes. Si no hay casillas disponibles para construir la avenida del tamaño requerido, no se hace nada.
  - construirAutopista(): construye una autopista, y de acuerdo a las coordenadas actuales se verifica y construye en las casillas adyacentes. Si no hay casillas disponibles para construir la autopista del tamaño requerido, no se hace nada.

EdificioPublico: subclase de Casilla de tipo EdificioPublico.



- **Atributos:**
  - casillasRequeridas: actualiza el tipo de casillas requeridas de acuerdo al tipo de edificio público que especifíca el tamaño. Ejemplo: se intenta crear un parque grande, que por defecto ocupa 10 casillas.
  - calidad: guarda la calidad del edificio.
  - influencia: almacena el tipo de influencia, e.g., inseguridad.
  - areaDeInfluencia: almacena el número de casillas afectadas por la influencia de esta edificación.
- **Métodos:**
  - EdificioPublico(): es el constructor de esta clase, recibe el tipo de construcción de donde obtiene el tipo de construcción, la calidad y el tamaño de la edificación. Busca en sus casillas adyacentes para ver si están disponibles para la construcción. Luego llama al subconstructor correspondiente.
  - construirColegio(): construye un colegio
  - construirParque(): construye un parque.
  - construirHospital(): construye un hospital.
  - construirEstacionPolicia(): construye una estación de policía.

Diagrama de secuencia de la interacción de avance diario