INTRODUCCIÓN A LA PROGRAMACIÓN Y COMPUTACIÓN 2
PROYECTO 2

201709149 – Percy Juventino Duarte Gálvez

Resumen

Se ha diseñado un programa simulador de la atención de clientes en un punto de atención al cliente perteneciente a una cierta empresa. La simulación brinda información de los tiempos de espera y atención mínima, máxima y media para escritorios y puntos de atención en general.

El programa recupera la información de las empresas, puntos y escritorios de archivos de entrada en formato XML y es almacenado en estructuras de datos tipo TDA. El usuario puede seleccionar una empresa y un punto de atención para realizar las pruebas.

De una prueba de simulación se obtiene la visualización de una cola de clientes, los tiempos medios de espera en la cola para cada cliente, tiempo de atención, escritorios activos y una visualización gráfica de la estructura activa en la simulación.

Palabras clave

Coordenada: sistema de referencia para ubicación de posición.
Matriz: arreglo bidimensional de números o datos.
Ortogonal: 90° respecto de una referencia. Para un movimiento se da en dirección norte, sur, este y oeste.
TDA: Tipo de Dato Abstracto (modelo de datos en memoria dinámica).
XML: Xtensible Markup Language (Lenguaje de marcado extensible).

Abstract

A simulator program for customer service has been designed at a customer service point belonging to a certain company. The simulation provides information on waiting times and minimum, maximum and average service for desks and service points in general.

The program retrieves information on companies, points and desks from input files in XML format and is stored in TDA-type data structures.

The user can select a company and a service point to carry out the tests.

From a simulation test, the display of a customer queue, the average waiting times in the queue for each customer, service time, active desks and a graphic display of the active structure in the simulation are obtained.

Keywords

ADT: *Abstracta Data Type (dynamic memory data model).*
Coordinate: *reference system for position location.*
Matrix: *bidimensional array of numbers or data.*
Orthogonal: *90° from a reference. For a movement it occurs in a north, south, east and west direction.*
XML: *Xtensible Markup Language.*

Introducción

Una cola es una estructura de datos en la que el primer elemento en ingresar a ella, es el primero en salir. Siguiendo esa idea un elemento que no sea el primero tendrá que esperar la salida del primero para poder salir, llevando ello a un cierto tiempo de espera. Dado un punto de atención al cliente sobre servicios de una empresa que brinda transacciones a sus clientes en un grupo de escritorios del punto de atención puede estudiarse y simularse el comportamiento de la cola lo cual ha sido el principal propósito del sistema desarrollado.

Se han implementado colas con información de clientes y sus respectivas transacciones a realizar en un escritorio, cada transacción tiene un tiempo que le toma en atenderse, y el cliente puede realizar varias transacciones del mismo y distinto tipo.

La simulación consiste entonces en manejar el tiempo medio de atención de un escritorio y el punto de atención simulado, así como tiempos máximos y mínimos de atención y espera de los clientes.

a) Estructura del programa

El algoritmo principal solución se divide en dos grandes procesos, entrada de un cliente y salida de un cliente.
La entrada es manejada en una cola de disponibilidad en la cual están todos los clientes potenciales, si un cliente solicita atención es entonces agregado a la cola de atención en la cual está a la espera. Al agregarse a la cola se calcula el tiempo que le tomará en ser atendido y el tiempo medio que tiene que esperar. El tiempo medio de espera es dado para el punto de atención y se obtiene de los tiempos de atención sobre la cantidad de clientes ingresados.

La salida de un cliente remueve al cliente de la cola y da por concluida su atención brindando información de tiempos de atención para el escritorio y punto de atención.

b) Clases

b.1) Clase menú:

Se encarga de desplegar las opciones iniciales del programa. En él, el usuario solo debe escribir un número correspondiente a la opción deseada y presionar la tecla ENTER para acceder a dicha opción.

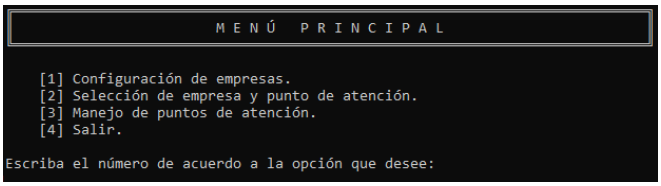

Figura 1. Menú principal
Fuente: elaboración propia.

b.1.1) [1] Cargar archivo

Permite al usuario elegir entre escribir directamente la ruta de carga del archivo o abrir el explorador de archivos del sistema para seleccionar el archivo de entrada. El archivo de entrada debe poseer la extensión .xml

b.1.2) [2] Procesar archivo

Una vez cargado un archivo de entrada puede accederse al menú 2 en el cual se hace uso de la clase Lector la cual se encarga de leer todos los datos de ciudades y de los robots como de los atributos de cada uno de ellos.

Los datos son almacenados en listas simplemente enlazadas para las ciudades, quienes contienen listas de recursos, unidades militares, civiles y entradas que

contenga el archivo de entrada. Y listas para los robots de ambos tipos, para recursos y rescates.

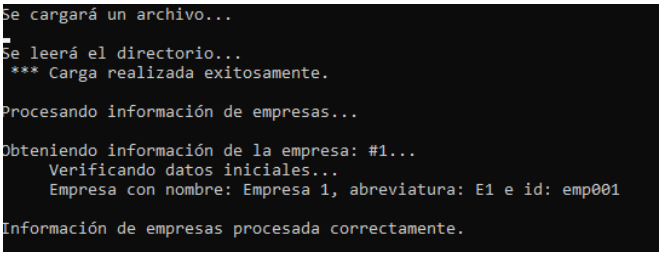

Figura 2. Menú 2: despliegue de datos procesados.
Fuente: elaboración propia.

b.1.3) [3,4] Iniciar simulación

Los menús [3] y [4] nos dan las opciones para misiones de rescate y extracción de recursos respectivamente.

En estos menú, se dan a elegir entre los robots disponibles, las ciudades que contentan unidades civiles o de recursos según correspondan y la entrada a utilizar para empezar la misión mientras se cumplan todos los requisitos mencionados.

Figura 3. Ejemplo opciones previas para realizar misión.
Fuente: elaboración propia.

Tras haber elegido una matriz y haber sido el archivo DOT, se abrirá automáticamente la imagen generada.

b.2) Clase Lector

La clase lector se encarga de leer el archivo de entrada y extraer los datos del formato XML almacenándolos en listas. Para almacenar losdatos de empresas y configuraciones extrae los datos de las etiquetas <piso> verificando sus propiedades de dimensiones, patrones inicial y final. Omite si se encuentran irregularidades en el archivo de entrada, de lo contrario almacena todos los pisos y sus datos en una lista simplemente enlazada.

b.3) Clase Empresa

Lista simplemente enlazada cuyos nodos son las ciudades obtenidas del archivo de entrada.

b.4) Clase Punto

Nodo de la clase Lista, almacena los datos de todas las unidades dentro de la ciudad y estructura del mapa.

b.5) Clase Cliente

Estructura de datos ortogonal con apuntadores para cada nodo que la conforma. Se encarga de almacenar a los nodos y de realizar las operaciones lógicas búsqueda de rutas para cada tipo de misión.

b.6) Clase Nodo

Nodo de la clase Matriz que a todas las unidades con sus respectivas coordenadas.

b.7) Clase Escritor

Se encarga de la escritura del código en lenguaje DOT necesario para graficar el camino obtenido si la misión fue exitosa. Almacena la matriz elegida y procede a extraer sus datos para colocar la información necesaria para poder graficar. Una vez finalizado el código lo compila y ejecuta de inmediato.

Anexos:

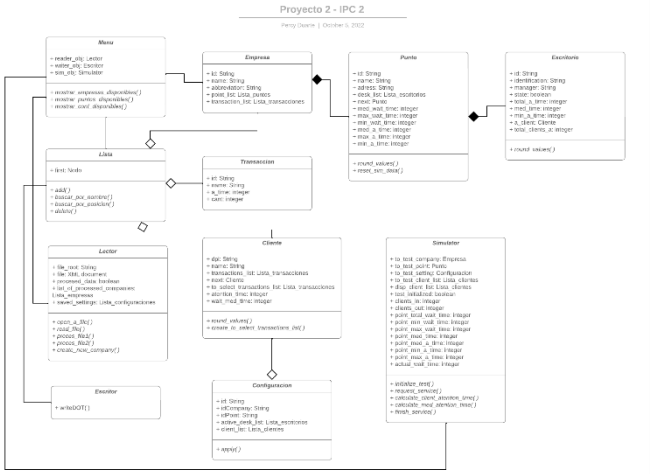

Figura 3. Diagrama de clases del programa.
Fuente: Elaboración propia.

Referencias bibliográficas

Graphviz. *Documentation*, 2010.
Disponible en: graphviz.org

Serrano. M. *Estructura de datos*. Universidad de Valladolid. Segovia.
Disponible en: www.infor.uva.es