ALTERNATIVA ALGORÍTMICA PARA LA GENERACIÓN DE FIXTURES DEL FPC



**Julio Mario Daza-Escorcia**

**DESCRIPCIÓN DE LAS ETAPAS PARA EL DESARROLLO DEL SOFTWARE**

**DEL FIXTURE DEL FPC**

A continuación, se presenta la descripción de las etapas que requiere el desarrollo de un software como son: el análisis, el diseño, la implementación y la validación.

1. **ANÁLISIS**

El siguiente software, Fixture del FPC usando cuadrados latinos, se enfoca en la optimización de la generación del fixture del FPC, el cual pretende reducir los costos de la organización de los torneos en temas de viaje y transporte de los jugadores. Estos sistemas están teniendo un auge considerable en los últimos años, debido a que ofrecen una alternativa a la organización de los encuentros optimizando de esta forma, los recursos de la Federación y ayuda a reducir el agotamiento mental que sufren los jugadores.

En esta aplicación, se pretende dar respuesta a los problemas crear fixtures aleatorios, con el modelo actual que usa la Federación Colombiana de Futbol. Una vez integrados, se desarrollan algoritmos para optimizar los encuentros de los equipos en el fixture para reducir costos de transporte y estrés en sus jugadores.

Ante el dinamismo actual, la misión de lograr nuevas ventajas competitivas y/o fortalecer las existentes, apunta a resolver práctica, ágil y satisfactoriamente muchos de los problemas organizacionales. Dentro de las empresas, la logística es una fuente inagotable de opciones estratégicas, por la enorme oportunidad de reducir costos (y con esto, su posicionamiento en la mente del cliente) y aprovechar de mejor forma los medios a disposición. El siguiente árbol de causas y efectos establece estas oportunidades en la gestión del transporte, que es un área identificada como crucial para la optimización de sistemas logísticos, y el problema de enrutamiento que se tiene actualmente en la creación de los fixtures.

Deficiencias en la creación y gestión de los fixtures (específicamente en la optimización de distancias y costos)

Alto estrés en los jugadores de los equipos.

Alto costo de la gestión de transporte.

Baja asistencia en los partidos.

Altos gastos monetarios por parte del FPC, altos niveles de estrés en los jugadores.

Metodologías viejas que generan perdidas.

Políticas de gerencia ineficientes, manejo de recursos deficiente.

Decisiones basadas en información no confiable.

Desconocimiento de alternativas de solución

Desconocimiento de las necesidades del área

Sistema de información inadecuado

Problema central

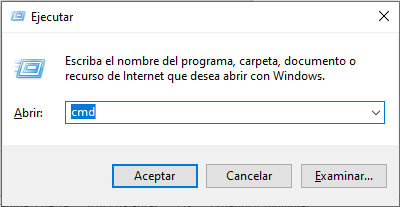
Figura 1. Árbol de causas y efectos del problema general.

1. **DISEÑO**

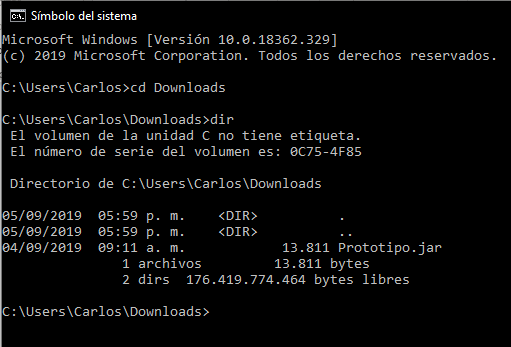
SPORTS SCHEDULING es un software, diseñado para el ordenamiento y creación automática del fixture del Futbol profesional colombiano, el cual busca optimizar las distancias recorridas por los clubes para de esa forma, disminuir los costos de viaje en los que incurren no los clubes, sino la Federación de Futbol Profesional Colombiano. El software en la actualidad no presenta una interfaz gráfica, sin embargo, este presenta ya unas vistas y menús que se pueden visualizar en el command prompt. El presente documento sirve de material de consulta para un eventual proceso de aprendizaje de la utilización del software. A continuación, se realiza una explicación del software y los procedimientos necesarios para su correcta utilización:

**2.1. Ejecución de la aplicación**

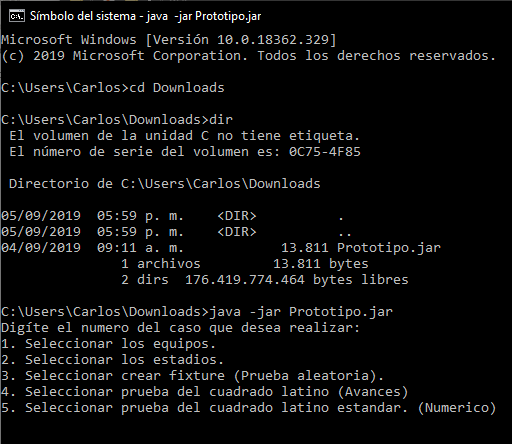
Para ejecutar el programa SPORTS SCHEDULING es necesario ejecutar primero la terminal, para esto debemos presionar las teclas de **Windows + R**, luego escribir **CMD** para ejecutar la ventana de comandos.



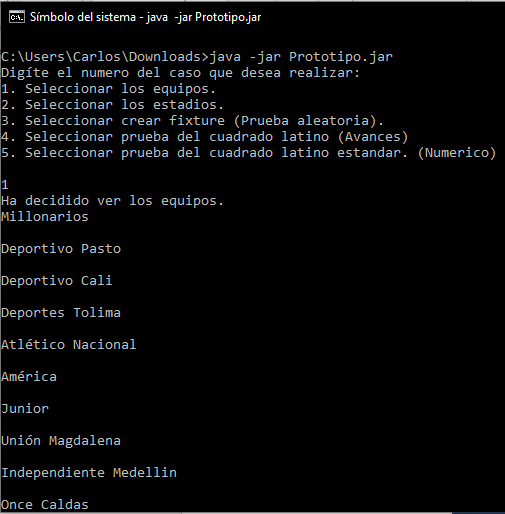
Seguido de esto, debemos ubicarnos en la carpeta donde se encuentra el archivo .java, para poder ejecutarlo desde la ventana de comandos.



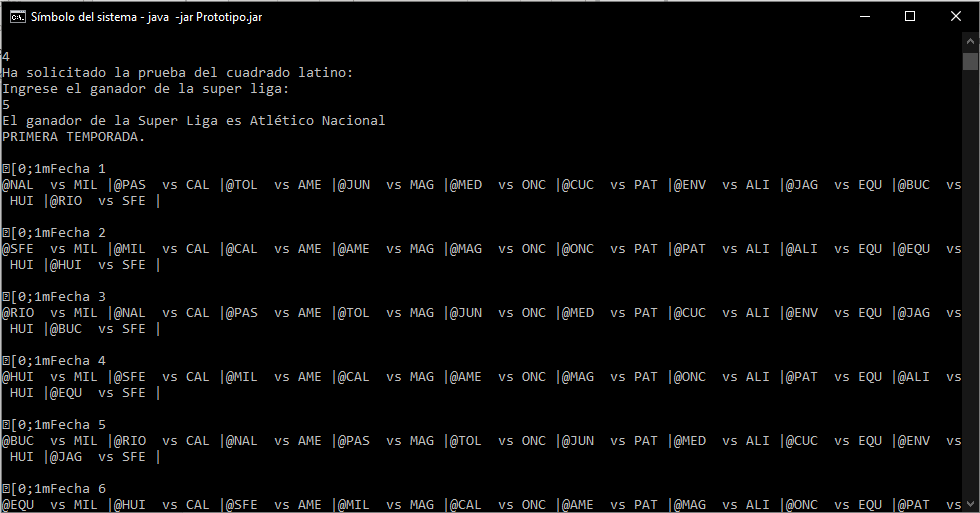
Luego cuando nos encontremos ubicados en la posición donde se encuentra la aplicación **“Prototipo.jar”** ejecutamos el siguiente comando: **“java -jar Prototipo.jar”** para ejecutar la aplicación y poder usarla.



Al ejecutarse **SPORTS SCHEDULING**, aparecerá en la pantalla de comandos, las opciones que actualmente tiene la aplicación. Presione **un número del menú** para ejecutar esa opción.

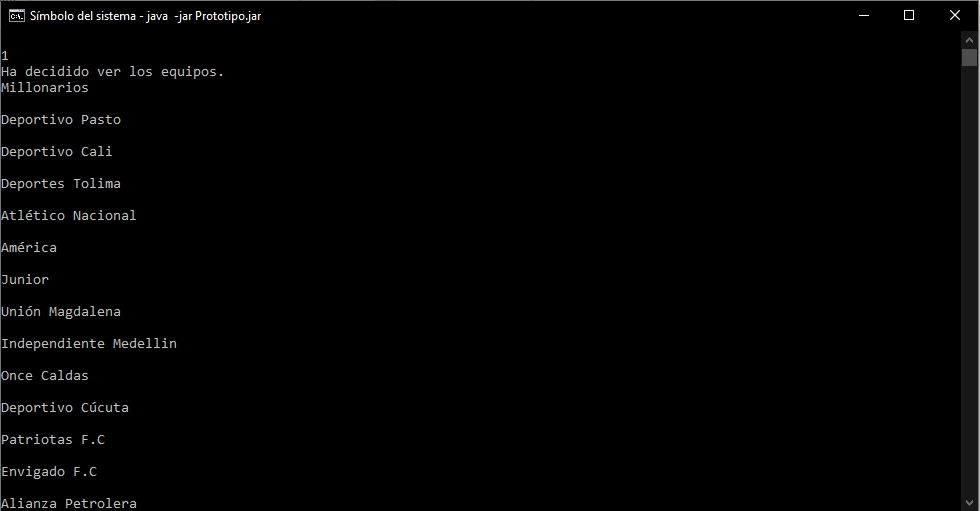


La aplicación ya se encuentra en una versión donde permite al usuario seguirla ejecutando hasta que este decida lo contrario, de igual forma se han corregido los errores encontrado para permitir una buena usabilidad por parte del usuario.

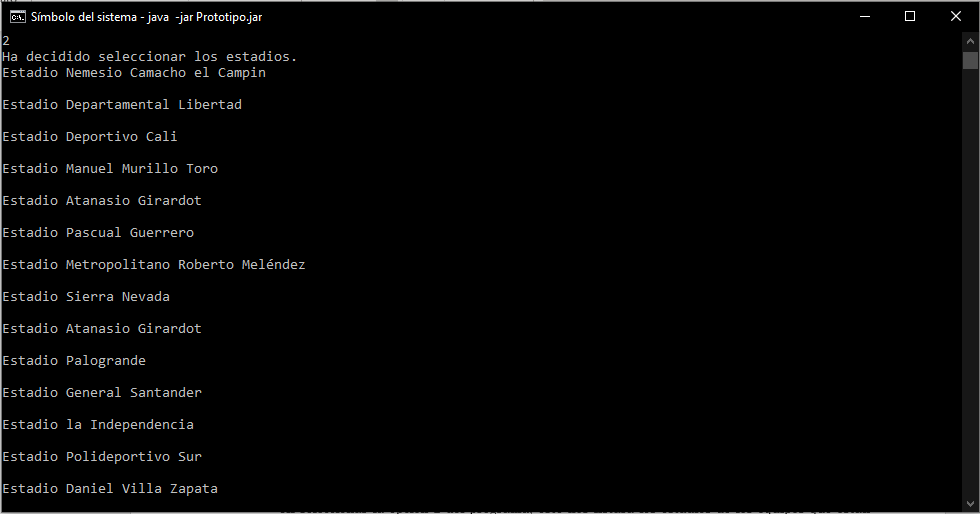


A continuación, se mostrará cómo se realizan las ejecuciones de la aplicación y lo que realiza cada opción seleccionada del programa SPORTS SCHEDULING.

Al seleccionar la opción 1 del programa, este nos listara los equipos que están actualmente en la liga águila.



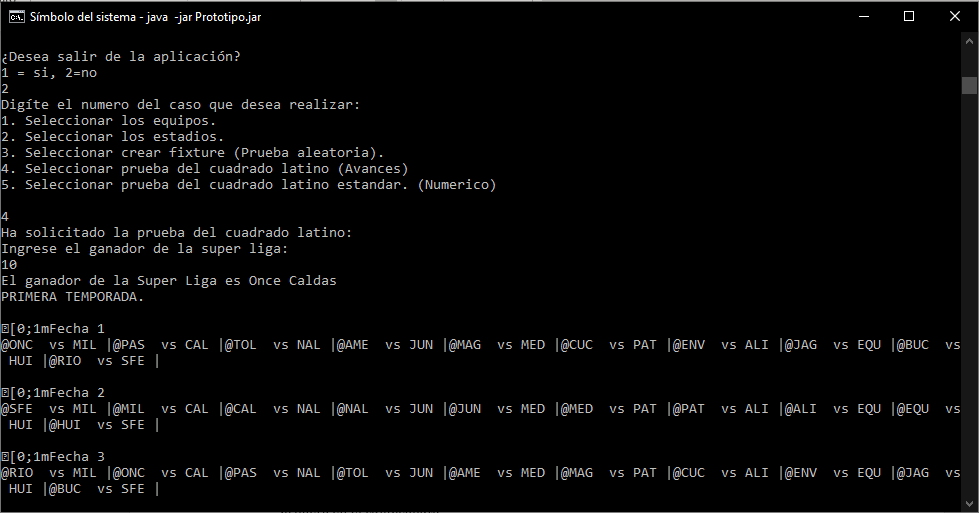
Al seleccionar la opción 2 del programa, este nos listara los estadios de los equipos que están participando en la liga águila:



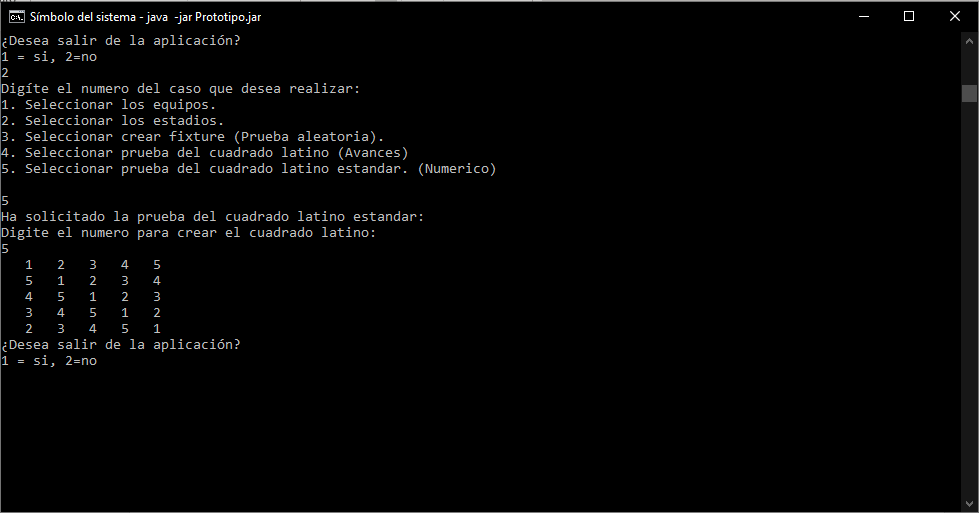
Al seleccionar la opción 3 del programa, este nos creara un fixture aleatorio entre los equipos participantes de la liga águila:



Al seleccionar la opción 4 del programa, este seleccionará crear el fixture de la liga águila, pero primero nos preguntará quien fue el ganador de la super liga para determinar quien jugará primero en el campeonato:

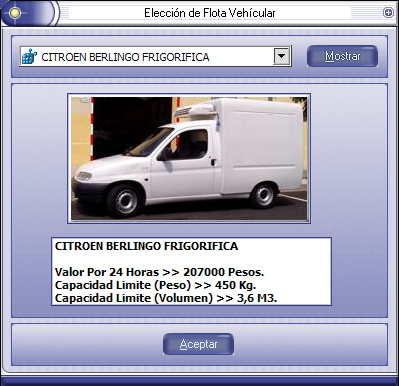


Al seleccionar la opción 5 del programa, este nos creara un cuadrado latino depende del numero que ingresemos al sistema:

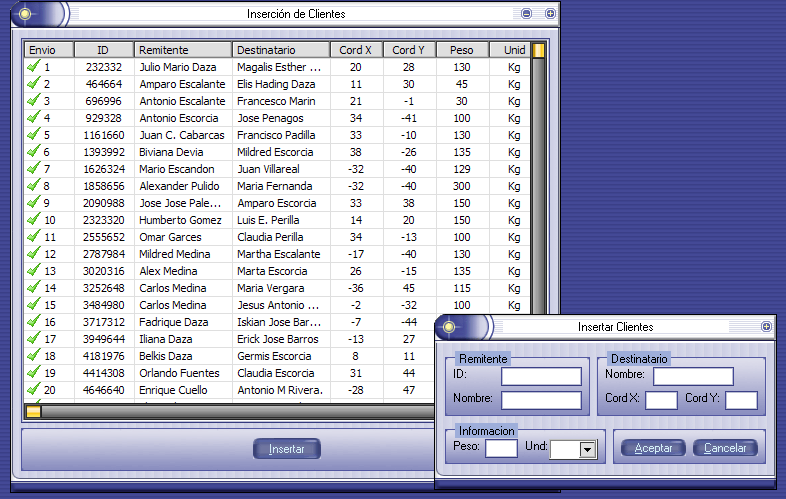


**2.2. Módulos de inicialización del sistema**

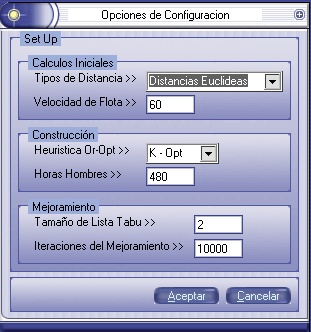
El primer módulo disponible, es la definición de la flota, para acceder a este, se debe seguir la siguiente ruta **Parámetros/Definir Flota.** En este módulo, se le suministra al software la información del tipo de la flota vehicular a utilizar (de carga general o frigorífica), su capacidad límite y su costo.



El segundo módulo disponible, es la definición de clientes, para acceder a este, se debe seguir la siguiente ruta **Parámetros/Insertar Clientes**. En este módulo, se le suministra al software la información de cantidad de clientes a rutear, sus direcciones y la carga a transportar.

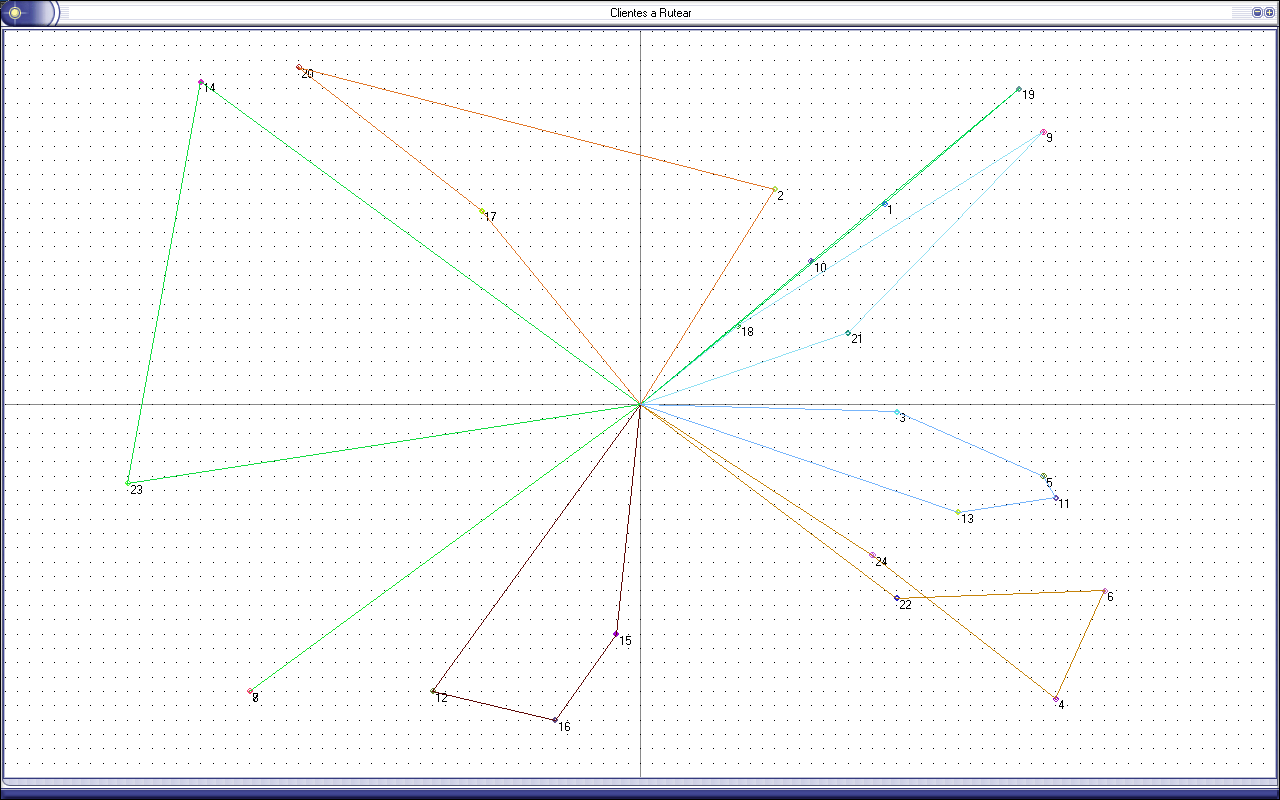
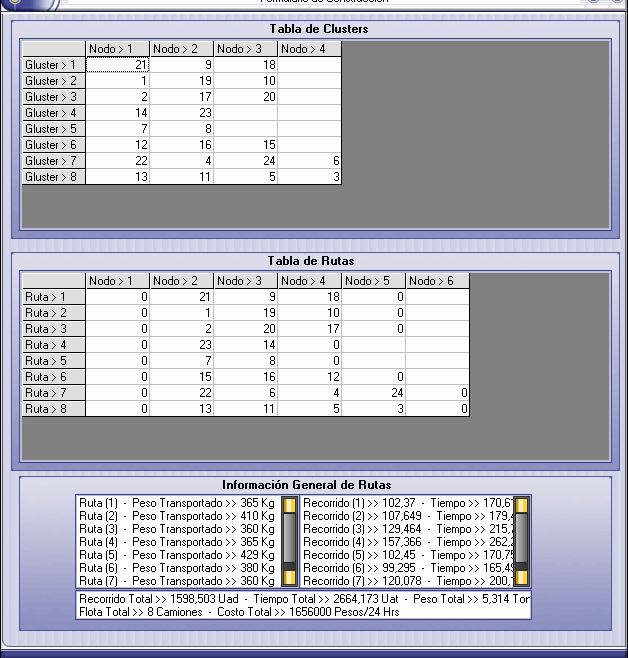


El tercer módulo disponible, es la configuración, para acceder a este, se debe seguir la siguiente ruta **Herramientas/Opciones de Configuración**. En este módulo, se le suministra al software la información de tipo de Heurística Or-Opt a utilizar, las HH que componen el jornal, la velocidad media de la flota, el tamaño de la lista tabú, y el número de iteraciones para el mejoramiento.

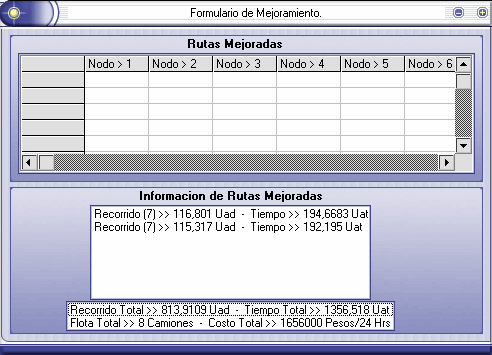


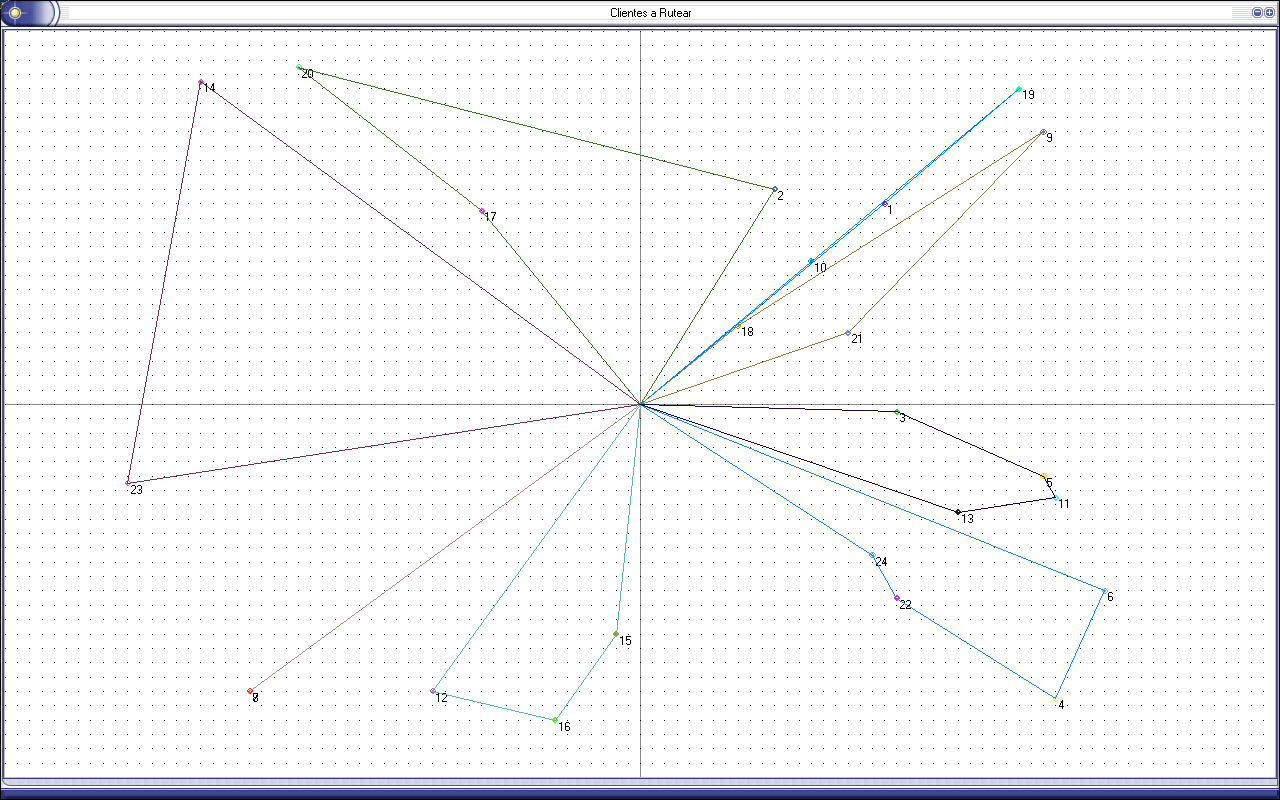
**2.3. Funcionamiento de RV-SmartCities**

Luego de completar los módulos anteriores se procede a ejecutar el ruteo y la posterior planificación, para esto seguimos la ruta, **Ejecutar/Ruteo/Construcción**, aquí el software muestra la construcción de las rutas con sus respectivos tiempos, distancia, capacidad y costo, de manera simultánea puede observar el resultado de la construcción gráficamente, para esto seguimos la ruta **Ver/Mapa**.



Posterior a esto se procede a ejecutar el mejoramiento de las rutas construidas, para esto seguimos la ruta, **Ejecutar/Ruteo/Mejoramiento**, aquí el software muestra el mejoramiento de las rutas previamente construidas con sus respectivos tiempo, distancia, capacidad y costo, de manera simultánea puede observar el resultado del mejoramiento gráficamente, para esto seguimos la ruta **Ver/Mapa**.

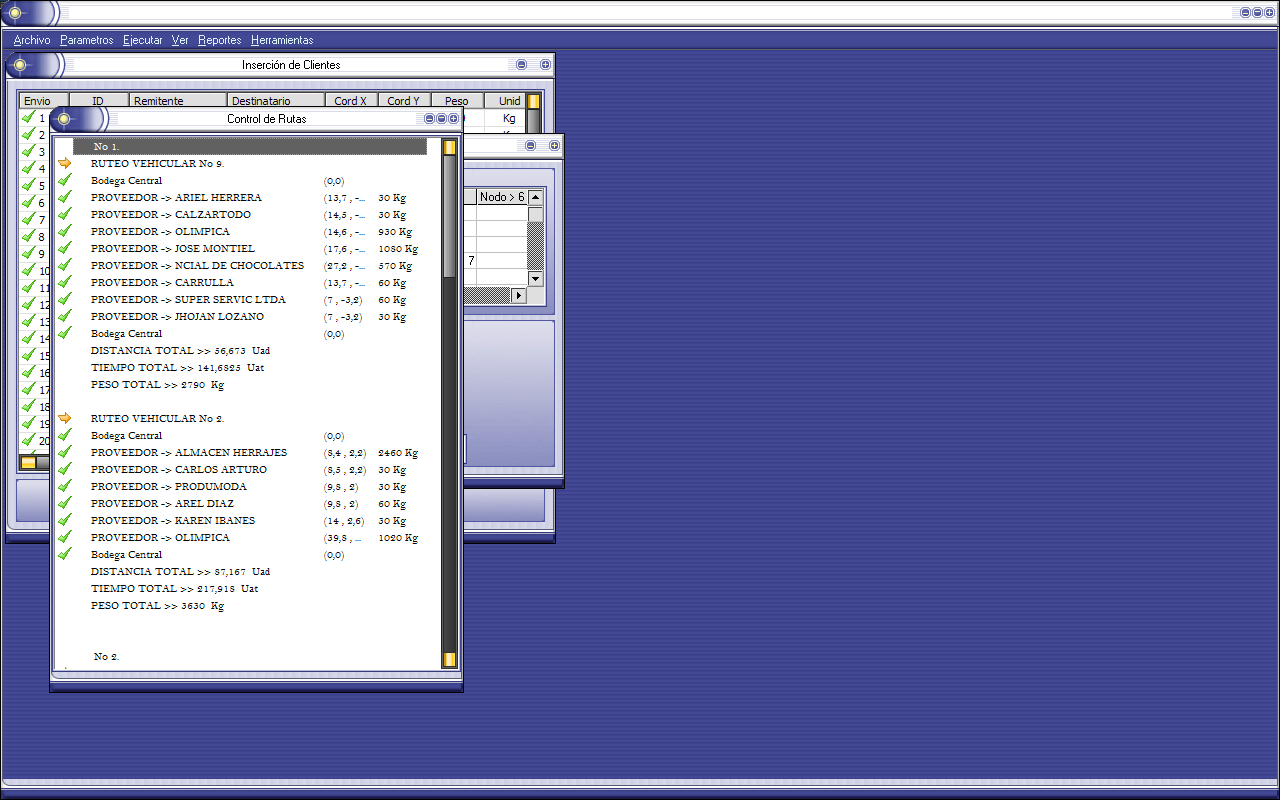




A continuación se procede a ejecutar la planificación de las rutas construidas y mejoradas en el ruteo, para esto seguimos la ruta, **Ejecutar/Planificación/Planificar**, aquí el software muestra gráficamente mediante un diagrama de Gantt los conjuntos de rutas que pueden abastecerse en un tiempo determinado para maximizar así la utilización de la capacidad instalada.



Por último, se puede observar esta planificación a manera de reporte para mayor aclaración de los conjuntos de rutas a abastecer, para esto seguimos la ruta, **Ver/Reporte/Control de Rutas**.



**3. IMPLEMENTACIÓN DEL SOFTWARE SPORTS SCHEDULING**

La implementación del software se realizó a través de programación orientada a objetos, utilizando el lenguaje de programación Java, utilizando la herramienta de NetBeans, haciendo iteraciones conforme a la metodología adoptada, permitiendo que se desarrollara el prototipo de manera incremental e iterativa.

A continuación, se muestra el código fuente para alternativa algorítmica para problemas de creación de fixtures.

El proyecto se divide por el momento en 2 clases, que son, Prototipo.java y Equipos.java

**Prototipo.java:**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package prototipo;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.InputMismatchException;

import java.util.Random;

import java.util.Scanner;

import java.util.Vector;

/\*\*

\*

\* @author Carlos David Quijano

\*/

public class Prototipo {

/\*\*

\* @param args the command line arguments

\*/

static Scanner sc = new Scanner(System.in);

public static void main(String[] args) {

boolean salir = false;

Prototipo menuDespliegue = new Prototipo();

do{

menuDespliegue.Menu();

System.out.println("¿Do you want to exit the application?");

System.out.println("1 = Yes, 2 = No");

int resp;

resp = sc.nextInt();

if(resp == 1){

salir = true;

}

}while(salir == false);

}

public static Vector<Equipos> Equipos() {

Vector<Equipos> vectorEquipos = new Vector();

Equipos equipo1 = new Equipos(1, "Millonarios", "Estadio Nemesio Camacho el Campin", "Bogotá", "MIL");

Equipos equipo2 = new Equipos(2, "Deportivo Pasto", "Estadio Departamental Libertad", "Pasto", "PAS");

Equipos equipo3 = new Equipos(3, "Deportivo Cali", "Estadio Deportivo Cali", "Cali", "CAL");

Equipos equipo4 = new Equipos(4, "Deportes Tolima", "Estadio Manuel Murillo Toro", "Ibagué", "TOL");

Equipos equipo5 = new Equipos(5, "Atlético Nacional", "Estadio Atanasio Girardot", "Medellin", "NAL");

Equipos equipo6 = new Equipos(6, "América", "Estadio Pascual Guerrero", "Cali", "AME");

Equipos equipo7 = new Equipos(7, "Junior", "Estadio Metropolitano Roberto Meléndez", "Barranquilla", "JUN");

Equipos equipo8 = new Equipos(8, "Unión Magdalena", "Estadio Sierra Nevada", "Santa Marta", "MAG");

Equipos equipo9 = new Equipos(9, "Independiente Medellin", "Estadio Atanasio Girardot", "Medellin", "MED");

Equipos equipo10 = new Equipos(10, "Once Caldas", "Estadio Palogrande", "Manizales", "ONC");

Equipos equipo11 = new Equipos(11, "Deportivo Cúcuta", "Estadio General Santander", "Cúcuta", "CUC");

Equipos equipo12 = new Equipos(12, "Patriotas F.C", "Estadio la Independencia", "Tunja", "PAT");

Equipos equipo13 = new Equipos(13, "Envigado F.C", "Estadio Polideportivo Sur", "Envigado", "ENV");

Equipos equipo14 = new Equipos(14, "Alianza Petrolera", "Estadio Daniel Villa Zapata", "Barrancabermeja", "ALI");

Equipos equipo15 = new Equipos(15, "Jaguares de Córdova", "Estadio Municipal de Monteria", "Monteria", "JAG");

Equipos equipo16 = new Equipos(16, "Equidad", "Estadio Metropolitano de Techo", "Bogotá", "EQU");

Equipos equipo17 = new Equipos(17, "Atlético Bucaramanga", "Estadio Alfonzo López", "Bucaramamga", "BUC");

Equipos equipo18 = new Equipos(18, "Atlético Huila", "Estadio Guillermo Plazas Alcid", "Neiva", "HUI");

Equipos equipo19 = new Equipos(19, "Rionegro Águilas", "Estadio Alberto Grisales", "Rionegro", "RIO");

Equipos equipo20 = new Equipos(20, "Santa Fe", "Estadio Nemesio Camacho el Campin", "Bogotá", "SFE");

vectorEquipos.add(equipo1);

vectorEquipos.add(equipo2);

vectorEquipos.add(equipo3);

vectorEquipos.add(equipo4);

vectorEquipos.add(equipo5);

vectorEquipos.add(equipo6);

vectorEquipos.add(equipo7);

vectorEquipos.add(equipo8);

vectorEquipos.add(equipo9);

vectorEquipos.add(equipo10);

vectorEquipos.add(equipo11);

vectorEquipos.add(equipo12);

vectorEquipos.add(equipo13);

vectorEquipos.add(equipo14);

vectorEquipos.add(equipo15);

vectorEquipos.add(equipo16);

vectorEquipos.add(equipo17);

vectorEquipos.add(equipo18);

vectorEquipos.add(equipo19);

vectorEquipos.add(equipo20);

return vectorEquipos;

}

public static void Estadios(Vector<Equipos> Equipos) {

for (int j = 0; j < Equipos.size(); j++) {

System.out.println(Equipos.elementAt(j).nombreCiudad);

}

}

public void Menu() {

try{

System.out.println("Enter the case number you wish to make: \n"

+ "1. See the teams. \n"

+ "2. See the stadiums. \n"

+ "3. Create fixture (Random test). \n"

+ "4. Create fixture (final advances) \n"

+ "5. Create latin square (Numeric) \n");

int caso = sc.nextInt();

Vector<Equipos> equipos = new Vector();

equipos = Equipos();

if((caso<6)&&(caso>0)){

switch (caso) {

case 1:

System.out.println("Has decided to see the teams: ");

for (int i = 0; i < equipos.size(); i++) {

System.out.println(equipos.elementAt(i).nombreEquipo + "\n");

}

break;

case 2:

System.out.println("Has decided to see the stadiums: ");

for (int i = 0; i < equipos.size(); i++) {

System.out.println(equipos.elementAt(i).nombreEstadio + "\n");

}

break;

case 3:

System.out.println("Has decided to create the fixture (Random test): ");

Fixture();

System.out.println("\n");

break;

case 4:

System.out.println("Has decided to create the fixture (final advances): ");

System.out.println("Please enter the winner of the Super Liga: ");

int ganador = sc.nextInt();

ejemploCuadrado2(GanadorSuperLiga(ganador));

System.out.println("\n");

ejemploCuadrado3(GanadorSuperLiga(ganador));

break;

case 5:

System.out.println("Has decided to create the latin square: ");

ejemploCuadrado();

}

}

else{

System.out.println("Wrong input, please try again.");

}

}catch(InputMismatchException e){

System.out.println("Wrong input, please try again.");

}

}

public static Vector<Equipos> Fixture() {

Vector<Equipos> equipos = new Vector();

equipos = Equipos();

int fecha = 1;

Random rnd = new Random();

System.out.println("Has decided to create the Fixture.");

for (int f = 0; f < 20; f++) {

System.out.println("Date " + fecha);

for (int i = 0; i < 10; i++) {

System.out.println("\n");

for (int j = 0; j < 2; j++) {

System.out.print(equipos.elementAt(rnd.nextInt(equipos.size())).acronimo);

if (j != 1) {

System.out.print(" vs ");

}

}

}

System.out.println("\n");

fecha++;

}

return equipos;

}

public void cuadradoLatino() {

System.out.println("Enter a number");

Cuadrado n = new Cuadrado(sc.nextInt());

int x;

int y = n.regresarValor();

for (x = 0; x < y; x++) {

if (x == 0) {

n.guardarValores(y);

} else {

n.recorrerValores();

}

}

}

public void ejemploCuadrado() {

System.out.println("Enter a number to create the latin square: ");

int q = sc.nextInt();

int FILAS = q;

int COLS = q;

int matriz[][] = new int[FILAS][COLS];

Vector<Equipos> equipos = new Vector();

for (int fil = 0; fil < FILAS; fil++) {

for (int col = 0; col < COLS; col++) {

//si es la primera fila

if (fil == 0) {

matriz[fil][col] = col + 1;

} //si no, si es la primera columna

else if (col == 0) {

matriz[fil][col] = matriz[fil - 1][COLS - 1];

} // para el resto de casos

else {

matriz[fil][col] = matriz[fil - 1][col - 1];

}

}

}

for (int i = 0; i < FILAS; i++) {

for (int j = 0; j < COLS; j++) {

if (matriz[i][j] < 10) {

System.out.printf(" %d", matriz[i][j]);

} else {

System.out.printf(" %d", matriz[i][j]);

}

}

System.out.println();

}

}

public void ejemploCuadrado2(Vector<Equipos> ganador) {

int FILAS = 19;

int COLS = 20;

Equipos matriz[][] = new Equipos[FILAS][COLS];

Random rnd = new Random();

int fecha = 1;

System.out.println("The winner of the Super Liga is "+ganador.elementAt(0).nombreEquipo);

for (int fil = 0; fil < FILAS; fil++) {

for (int col = 0; col < COLS; col++) {

//si es la primera fila

if (fil == 0){

matriz[fil][col] = ganador.elementAt(col);

} //si no, si es la primera columna

else if (col == 0) {

matriz[fil][col] = matriz[fil - 1][COLS - 1];

} // para el resto de casos

else {

matriz[fil][col] = matriz[fil-1][col-1];

}

}

}

System.out.println("FIRST SEASON.");

for (int i = 0; i < FILAS; i++) {

System.out.print("\n");

System.out.println("\033[0;1m" + "Date " + fecha);

for (int j = 0; j < COLS; j++) {

if (j%2!=1) {

System.out.print("@" + matriz[i][j].acronimo + " ");

} else{

System.out.print(matriz[0][j].acronimo + " ");

//System.out.print(equipos.elementAt(rnd.nextInt(equipos.size())).acronimo);

}

if (j % 2 != 1) {

System.out.print(" vs ");

}

if (j % 2 == 1) {

System.out.print("|");

}

}

System.out.println();

fecha++;

}

}

public void ejemploCuadrado3(Vector<Equipos> ganador) {

int FILAS = 19;

int COLS = 20;

Equipos matriz[][] = new Equipos[FILAS][COLS];

Random rnd = new Random();

int fecha = 1;

for (int fil = 0; fil < FILAS; fil++) {

for (int col = 0; col < COLS; col++) {

//si es la primera fila

if (fil == 0){

matriz[fil][col] = ganador.elementAt(col);

} //si no, si es la primera columna

else if (col == 0) {

matriz[fil][col] = matriz[fil - 1][COLS - 1];

} // para el resto de casos

else {

matriz[fil][col] = matriz[fil-1][col-1];

}

}

}

System.out.println("SECOND SEASON.");

for (int i = 0; i < FILAS; i++) {

System.out.print("\n");

System.out.println("\033[0;1m" + "Date " + fecha);

for (int j = 0; j < COLS; j++) {

if (j%2!=1) {

System.out.print(matriz[i][j].acronimo + " ");

} else{

System.out.print("@" + matriz[0][j].acronimo + " ");

//System.out.print("@" + equipos.elementAt(rnd.nextInt(equipos.size())).acronimo);

}

if (j % 2 != 1) {

System.out.print(" vs ");

}

if (j % 2 == 1) {

System.out.print("|");

}

}

System.out.println();

fecha++;

}

}

private Vector<Equipos> GanadorSuperLiga(int ganador) {

Vector<Equipos> equipos = new Vector();

Vector<Equipos> equiposOrganizados = new Vector();

equipos = Equipos();

for(int i=0; i<20;i++){

if(equipos.get(i).codigoEquipo == ganador){

equiposOrganizados.add(equipos.get(i));

}

}

for(int j=0; j<20; j++){

if(equipos.get(j).codigoEquipo != ganador){

equiposOrganizados.add(equipos.get(j));

}

}

return equiposOrganizados;

}

// private Vector<Equipos> Rivales(int ganador){

// Vector<Equipos> rivales = new Vector();

// Vector<Equipos> totales = new Vector();

// Vector<Equipos> locales = new Vector();

// totales = Equipos();

// locales = GanadorSuperLiga(ganador);

//

// for(int i=0;i<=10;i++){

// for(int j=0;j<10;j++){

// if(totales.elementAt(i).codigoEquipo != locales.elementAt(j).codigoEquipo){

// if(i>1){

// if((locales.elementAt(i).codigoEquipo == locales.elementAt(i-1).codigoEquipo)){

// rivales.add(totales.elementAt(i));

// }

// }

//

// }

// }

// }

// return rivales;

// }

private Vector<Equipos> clasicos(){

Vector<Equipos> equipos = new Vector();

Equipos equipo1 = new Equipos(1, "Millonarios", "Estadio Nemesio Camacho el Campin", "Bogotá", "MIL");

Equipos equipo2 = new Equipos(2, "Deportivo Pasto", "Estadio Departamental Libertad", "Pasto", "PAS");

Equipos equipo3 = new Equipos(3, "Deportivo Cali", "Estadio Deportivo Cali", "Cali", "CAL");

Equipos equipo4 = new Equipos(4, "Deportes Tolima", "Estadio Manuel Murillo Toro", "Ibagué", "TOL");

Equipos equipo5 = new Equipos(5, "Atlético Nacional", "Estadio Atanasio Girardot", "Medellin", "NAL");

Equipos equipo6 = new Equipos(6, "América", "Estadio Pascual Guerrero", "Cali", "AME");

Equipos equipo7 = new Equipos(7, "Junior", "Estadio Metropolitano Roberto Meléndez", "Barranquilla", "JUN");

Equipos equipo8 = new Equipos(8, "Unión Magdalena", "Estadio Sierra Nevada", "Santa Marta", "MAG");

Equipos equipo9 = new Equipos(9, "Independiente Medellin", "Estadio Atanasio Girardot", "Medellin", "MED");

Equipos equipo10 = new Equipos(10, "Once Caldas", "Estadio Palogrande", "Manizales", "ONC");

Equipos equipo11 = new Equipos(11, "Deportivo Cúcuta", "Estadio General Santander", "Cúcuta", "CUC");

Equipos equipo12 = new Equipos(12, "Patriotas F.C", "Estadio la Independencia", "Tunja", "PAT");

Equipos equipo13 = new Equipos(13, "Envigado F.C", "Estadio Polideportivo Sur", "Envigado", "ENV");

Equipos equipo14 = new Equipos(14, "Alianza Petrolera", "Estadio Daniel Villa Zapata", "Barrancabermeja", "ALI");

Equipos equipo15 = new Equipos(15, "Jaguares de Córdova", "Estadio Municipal de Monteria", "Monteria", "JAG");

Equipos equipo16 = new Equipos(16, "Equidad", "Estadio Metropolitano de Techo", "Bogotá", "EQU");

Equipos equipo17 = new Equipos(17, "Atlético Bucaramanga", "Estadio Alfonzo López", "Bucaramamga", "BUC");

Equipos equipo18 = new Equipos(18, "Atlético Huila", "Estadio Guillermo Plazas Alcid", "Neiva", "HUI");

Equipos equipo19 = new Equipos(19, "Rionegro Águilas", "Estadio Alberto Grisales", "Rionegro", "RIO");

Equipos equipo20 = new Equipos(20, "Santa Fe", "Estadio Nemesio Camacho el Campin", "Bogotá", "SFE");

equipos.add(equipo3);

equipos.add(equipo6);

equipos.add(equipo9);

equipos.add(equipo5);

equipos.add(equipo20);

equipos.add(equipo1);

equipos.add(equipo7);

equipos.add(equipo8);

equipos.add(equipo17);

equipos.add(equipo11);

equipos.add(equipo4);

equipos.add(equipo18);

return equipos;

}

}

**Equipos.java:**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package prototipo;

/\*\*

\*

\* @author Carlos David Quijano

\*/

public class Equipos {

//Atributos de los equipos.

int codigoEquipo;

String nombreEquipo;

String nombreEstadio;

String nombreCiudad;

String nombreAbreviación;

String acronimo;

//Metodos

public Equipos(int codigoEquipo, String nombreEquipo, String nombreEstadio, String nombreCiudad, String acronimo) {

this.codigoEquipo = codigoEquipo;

this.nombreEquipo = nombreEquipo;

this.nombreEstadio = nombreEstadio;

this.nombreCiudad = nombreCiudad;

this.acronimo = acronimo;

}

public String getAcronimo() {

return acronimo;

}

public void setAcronimo(String acronimo) {

this.acronimo = acronimo;

}

public int getCodigoEquipo() {

return codigoEquipo;

}

public void setCodigoEquipo(int codigoEquipo) {

this.codigoEquipo = codigoEquipo;

}

public String getNombreEquipo() {

return nombreEquipo;

}

public void setNombreEquipo(String nombreEquipo) {

this.nombreEquipo = nombreEquipo;

}

public String getNombreEstadio() {

return nombreEstadio;

}

public void setNombreEstadio(String nombreEstadio) {

this.nombreEstadio = nombreEstadio;

}

public String getNombreCiudad() {

return nombreCiudad;

}

public void setNombreCiudad(String nombreCiudad) {

this.nombreCiudad = nombreCiudad;

}

}

**4. VALIDACIÓN DEL SOFTWARE RV-SMARTCITIES**

La validación se realizó comprobando que el sistema cumpliera con los requisitos que se exponen en el proyecto de investigación ALTERNATIVA ALGORÍTMICA PARA PROBLEMAS DE ENRUTAMIENTO DE VEHÍCULOS EN SMART CITIES. Para esto se realizaron pruebas de aceptación en instancias conocidas y aleatorias, y en todos los casos mostrando buenos resultados.