Instituto Politécnico Nacional

Secretaría de Investigación y Posgrado

Centro de Investigación en Computación

OPTIMIZACIÓN DE PROCESOS DE ATENCIÓN A FUGAS HIDRÁULICAS PARA MINIMIZAR EL DESPERDICIO DE AGUA Y TIEMPOS DE REPARACIONES

T E S I S

QUE PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A

ING. MANUEL MARTÍNEZ ÁLVAREZ

Director de Tesis: M. en C. Sandra Dinora Orantes Jiménez

México, D.F. Mayo 2014

Resumen

La gestión de flotas es el eje central de los procedimientos de muchas de las empresas de transporte de viajeros o de servicios, una correcta administración de ellas incidirá en la cuenta de resultados de quienes las dirijan, por lo que será fundamental realizar tareas de optimización para lograr un aumento en los beneficios de sus operaciones.

Una completa gestión de flotas, inmersa en los problemas de análisis combinatorio, requiere de Sistemas de Toma de Decisiones sustentados por varios procesos de optimización, a partir de la complejidad de los problemas que se tengan, así serán los procedimientos que se planteen para su solución.

Como parte de los procedimientos para resolver los problemas de análisis combinatorio, ha destacado el uso de metaheurísticas, tal como la Optimización basada en Colonias de Hormigas ACO (*Ant Colony Optimization*) en la que, a partir de un comportamiento colaborativo entre los elementos que componen esta técnica y búsquedas aleatorias, es que permite arrojar sus resultados.

En este trabajo, se presenta una alternativa de solución para llevar el control y atención de fugas hidráulicas en una ciudad, a través del análisis combinatorio en la Programación de Tareas y Enrutamiento de Vehículos; para ello se propone, mediante el uso del algoritmo *ACO*, la solución a los dos problemas base involucrados en la presente investigación: SMTWTP (*Single-Machine Total Weighted Tardiness Problem*, Problema de la Tardanza Total Ponderada en una Sola Máquina) y TSP (*Traveling Salesman Problem*, Problema del Agente Viajero), mostrando así, que al implementarlo en un Organismo de Suministro de Agua Potable, en el área de gestión de flotas de mantenimiento, puede ayudar a minimizar el desperdicio de agua y tiempos de reparaciones de las fugas.

Por la necesidad de los problemas contenidos en este trabajo para ubicar puntos dentro de una ciudad, se utilizará la Interfaz de Programación de Aplicaciones *Google Maps API,* misma que servirá para proporcionar la mayoría de los parámetros de entrada de los algoritmos a través del uso de la matriz de distancias que contiene la herramienta.

Abstract

Fleet management is the backbone of procedures for many of the passenger or service companies, the proper administration of them will affect the income of those who lead, so it will be essential for optimization tasks achieve an increase in the profits of their operations.

A complete fleet management, immersed in the problems of combinatorial analysis requires Decision Making Systems supported by several optimization processes, from the complexity of the problems that have, so are the procedures for settlement arising.

As part of the procedures for solving problems of combinatorial analysis, highlighted the use of metaheuristics, such as optimization based on ACO (Ant Colony Optimization) in which, from a collaborative behavior between the elements this technique and random searches, is that you shed your results.

In this work, an alternative solution is presented to track and care of hydraulic leaks in a city through combinatorial analysis in Task Scheduling and Vehicle Routing, for it is proposed, using the ACO algorithm, the solution to two basic problems involved in this investigation: SMTWTP (Single-Machine Total Weighted Tardiness Problem) and TSP (Traveling Salesman Problem), showing thus that when implemented in a Supplier of Water, in the area of fleet management maintenance can help minimize water wastage and time repairs leaks.

For the need of the problems in this work to locate points within a city, we will be use the Application Programming Interface of *Google Maps*, it will serve to provide most of the input parameters of the algorithms through the use of will be used the distance matrix that contains the tool.

Agradecimientos

Ningún trabajo tendría éxito sin la gente a quienes se les debe el poder lograrlo. NO ES MÍO, ES DE USTEDES.

Dios mío, mi gran amigo; Virgensita, Madre nuestra; gracias por guiarme cada día, ayudarme a tomar las decisiones correctas y enseñarme el verdadero valor de las cosas. No sé si lo merezco, pero me han dado la mejor vida que alguien podría desear…

A mis padres, Héctor y Rosaura, me dieron la vida, su tiempo y dedicación. Con sus sabios consejos formaron mi carácter. A ellos y a mí nos dolió el momento de separarnos, pero sabíamos que era el punto de partida de una nueva historia. A pesar de mis desastres, les agradezco que confíen en mí. “Men”, un gran padre, pero un mejor maestro. “Jefa”, el pilar más grande de mí formación.

A mis hermanos, Gustavo (mi carnal) y Rosy (mi Güera). Carnal, me has enseñado a siempre dar lo mejor de mí a pesar de las adversidades; mi güera, tu alegría siempre me hizo saber que a la vida, hay que sonreírle.

Rosario, la mujer que amo, fueron dos años de distancias y un sinfín de ilusiones acumuladas, aun así, tu siempre estuviste a mi lado. Gracias por ayudarme a sacar siempre lo mejor de mí y a enseñarme lo que es el verdadero amor de pareja. Fuiste mi mayor motivación en todo este tiempo. Ya casi mi cielo, ya casi.

A mis amigos, Daniela, Naye, Ismael y Enrique, fueron grandes batallas, buenos debates y excelentes experiencias, pero lo más importante, una gran amistad. -Ay Daniela, tus regaños si me sirvieron-.

Maestra Dinora, además de ser una gran directora de tesis y una excelente profesora, siempre la consideré como una gran amiga, su apoyo incondicional y su calidez como persona me ayudaron a tener la confianza suficiente para poder llegar hasta este momento.

IPN, CIC y CONACYT, constituyeron la base del conocimiento adquirido y el apoyo para tener la formación adecuada sin preocupaciones durante mi estadía en la Maestría.

Me encanta mi pueblo, Villa Morelos, Mich. (Mi villa) algún día espero volver a vivir ahí. Aunque no lo crean, el sabor de cada mes poder regresar a mi hogar, siempre me ayudó a recordar que no estaba solo, que había un lugar al cual pertenecía y del cual me daba orgullo poder hablar.

Amigos, familiares y profesores, MUCHAS GRACIAS POR TODO !!!

Tabla de contenido

Lista de Figuras

Lista de Tablas

Glosario

Aleatoriedad	Se asocia a todo proceso cuyo resultado no es previsible más que en razón de la intervención del azar.
Algoritmo	Es un conjunto prescrito de instrucciones o reglas bien definidas, ordenadas y finitas que permite realizar una actividad mediante pasos sucesivos.
Bacheo	Acción de reparar los baches o deterioros de un camino.
Base de Datos	Es una colección de datos interrelacionados y almacenados conjuntamente en uno o más ficheros de computadora.
Caso de Uso	Es una funcionalidad o actividad a llevarse a cabo como algún proceso independiente de un Sistema.
Feromonas	Son sustancias químicas secretadas por los seres vivos con el fin de provocar comportamientos específicos en otros individuos.
Fuga hidráulica	Es un escape físico de agua en cualquier punto del sistema de agua potable; puede ocurrir en conducciones, tanques de almacenamiento, redes de distribución, conexiones domiciliarias o dentro de las casas de los usuarios.
Heurística	Técnica o procedimiento práctico o informal para resolver problemas a partir de la búsqueda de soluciones aproximadas o sin una regla estricta.
Implementación	Es la realización de una aplicación, instalación o la ejecución de un plan, idea, modelo científico, diseño, especificación, estándar, algoritmo o política.
Ingeniería de Sistemas	Es la actividad de especificar, diseñar, implementar, validar, distribuir y mantener sistemas.
Interrelación	Relación mutua entre dos o más entidades.
Metaheurística	Método heurístico de propósito general diseñado para guiar a una heurística específica hacia regiones de solución muy

prometedoras.

Módulo Es una parte repetitiva, autónoma e intercambiable de un diseño modular.

Procedimiento Es un conjunto de acciones u operaciones que tienen que realizarse de la misma forma, para obtener siempre el mismo resultado bajo las mismas circunstancias.

Regla de Sturges Es una regla práctica acerca del número de clases que deben considerar al elaborarse un histograma.

Sector Hace referencia a la parte seccionada o cortada de un todo, es decir, la división de un área determinada a partir de una línea delimitadora.

Sistema Es una colección de componentes interrelacionados que trabajan conjuntamente para cumplir algún objetivo.

Software Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

Glosario de Acrónimos

ACO Ant Colony Optimization, Optimización basada en Colonias de Hormigas.

API Application Programming Interface, Interfaz de Programación de Aplicaciones.

DSS Decision Support System, Sistemas de Apoyo a la Toma de Decisiones.

EIS Executive Information System, Sistemas de Apoyo a Ejecutivos.

GIS Geographic Information System, Sistema de Información Geográfica.

JSP Job Shop Problem.

MIS Management Information System, Sistemas de Información Gerencial.

OLAP On-Line Analytical Processing, Procesamiento analítico en línea.

OOAPAS Organismo Operador de Agua Potable Alcantarillado y Saneamiento.

OSP Open Shop Problem.

PSO Particle Swarm Optimization, Optimización de Enjambre de Partículas.

SMTWTP Single-Machine Total Weighted Tardiness Problem, Problema de la Tardanza Total Ponderada en una Sola Máquina.

SQL Structured Query Language, Lenguaje de Consulta Estructurado.

TPS Transaction Processing Systems, Sistemas de Procesamiento de Transacciones.

TSP Traveling Salesman Problem, Problema de Agente Viajero.

UML Unified Modeling Language, Lenguaje Unificado de Modelado.

VRP Vehicle Routing Problem, Problema de Enrutamiento de Vehículos.

1 Introducción

La demanda por tener mejores sistemas de análisis de información esta cimentada en la certeza y eficiencia de los procesos que sustentan su desarrollo, aunque el resultado final de un producto de software es el sistema que se presenta, son los procesos que hay detrás y sus implementaciones lo que hacen sobresalir a cualquier aplicación.

Las tecnologías de la información y los procesos de desarrollo que las envuelven están cada vez más al alcance de quienes las deseen utilizar, razón por la cual el software se ha convertido en una tecnología imprescindible en tareas de investigación, ingeniería, negocios, etc. Cada una de las labores que participan en estas funciones se ven envueltas en procesos de análisis de información para la toma de decisiones y a medida que las necesidades de una organización aumentan, la demanda en la calidad en el desarrollo de software también amerita un incremento.

Para poder llegar a un punto en el que un sistema se desempeñe correctamente, es necesario que sus procesos sean eficaces y eficientes, para ello, es necesario un arduo análisis para los problemas que se buscan solucionar. En el caso particular de este trabajo, se propone una alternativa de solución a procesos de análisis combinatorio con el fin de ver reflejados sus resultados en la atención adecuada de fugas hidráulicas en una ciudad.

1.1 Antecedentes

El problema del Agente viajero (*Traveling Salesman Problem*) es uno de los más estudiados dentro de los Problemas de Optimización Combinatoria, continúa vigente y podría decirse, que su gran aceptación se debe a que es fácil de plantear, pero difícil al momento de resolver.

Entre sus aplicaciones se encuentran: mejorar las rutas para la entrega de productos, optimizar la distribución de caminos para el transporte, minimizar en robótica el número de desplazamientos al realizar una serie de perforaciones en un circuito impreso, apoyar en los sistemas de empresas de Turismo y Agencias de viajes, ayudar en la estandarización de los horarios de transportes, etc., es así que puede tomarse como base para el manejo de secuencias, donde es importante el orden en el cual *n* trabajos tienen que ser procesados de tal forma que se minimice el costo total de producción.

Para el caso de estudio de esta investigación no se busca repartir un producto, sino resolver dos problemas reales como lo son: enviar personal para la solución de averías y minimizar los desperdicios ocasionados por ellas con base en su orden de reparación, lográndose mejoras y beneficios a partir de las decisiones involucradas con la correcta y

pronta atención de cada problema, permitiendo utilizar herramientas de optimización para el análisis de información, que conlleven a la elaboración de sistemas que puedan producir escenarios para lograr una satisfacción tanto para la empresa, como para los clientes o usuarios de la misma.

1.2 Planteamiento del problema

En la actualidad el problema de mejora de servicios de suministro y atención a desperfectos, juegan un papel importante en los Sistemas Operadores de Agua Potable, ya que planificar adecuadamente envíos de cuadrillas puede significar considerables ahorros logísticos y de costos como desperdicios de agua y horas hombre.

Son por estas causas que surge el problema de ruteo, ya que diariamente los Organismos Operadores de Agua se encuentran con fugas hidráulicas, mismas que generan un desperdicio de agua potable y como consecuencia, un problema al momento de buscar cómo atenderlas de manera eficiente y poder trasladar empleados a su reparación.

Este problema consiste en generar rutas para evitar el desperdicio de agua en atención a la demanda ciudadana como una prioridad, por lo que se espera la solución de los reportes de fugas a la mayor brevedad posible, así como manejar eficientemente los recursos humanos y materiales en el organismo.

Por ello, esta investigación se centra en solucionar el problema de atención rápida a las fugas hidráulicas que hay en una ciudad, disminuyendo las pérdidas de agua que generan y mejorando el manejo de los recursos que en ellas intervienen, permitiendo minimizar ciertos factores que ayuden a la empresa a obtener beneficios; estos pueden ser: minimizar tiempos y maximizar el ahorro de agua, lo cual lleva a obtener menores costos y por lo tanto obtener beneficios y una mejor calidad de servicio e imagen.

Este tipo de problemas, de manera intrínseca, conllevan a realizar cálculos computacionales muy elevados para su solución, debido a su intratabilidad es que son clasificados como problemas de clase *NP*, de manera específica *NP-Hard* ya que todo problema *NP* se puede reducir a él. Si bien se pueden resolver, sus soluciones no suelen ser exactas, sino aproximadas.

1.3 Objetivos

1.3.1 Objetivo general

Realizar mediante análisis combinatorio una optimización de procesos en la programación de tareas y enrutamiento de vehículos, buscando la disminución en el desperdicio de agua generado por fugas hidráulicas, presentando un procedimiento alternativo de solución para su control y atención, minimizando el gasto de traslado de recursos para su reparación, utilizando como caso de estudio el OOAPAS (Organismo Operador de Agua Potable Alcantarillado y Saneamiento de la ciudad de Morelia, Michoacán).

1.3.2 Objetivos específicos

- Recopilar la información necesaria para atender el mantenimiento correctivo que realiza el organismo para identificar y estructurar los datos que se integrarán en el sistema.
- Hacer un análisis de la información que permita realizar un programa cuyos resultados de decisión minimicen el tiempo de atención a las fugas hidráulicas para evitar los desperdicios excesivos de agua, utilizando el promedio de desperdicio de agua de cada una de las fugas hidráulicas y la cantidad de horas estimadas para su reparación.
- Estudiar la aplicabilidad, ventajas y desventajas de metaheurísticas para la solución de los problemas de ruteo y programación de tareas, para seleccionar la adecuada para este trabajo con base en la información recopilada y analizada.
- Utilizar algoritmos y técnicas de programación acordes a la toma de decisiones que se requerirá como procedimiento alternativo de solución para el sistema que se implementará.
- Seleccionar una ruta efectiva para recorridos de cuadrillas de reparación, con base en las distancias que hay entre los puntos en donde se encuentran las fugas hidráulicas de la ciudad y los sectores en los que se encuentra dividida y efectuar un control de fugas hidráulicas a través de un manejo adecuado de recursos materiales y recursos humanos, empleando los algoritmos seleccionados.

1.4 Justificación

Existe una gran variedad de programas especializados tales como los sistemas SCADA (*Supervisory Control and Data Acquisition)* que permiten el trabajo de las estaciones que operan bajo el control de los organismos operadores de agua potable, sin embargo el control se limita sólo al nivel operativo de las mismas.

La detección de fallas en pozos y tuberías, en el caso del *OOAPAS* se basa en inspecciones de rutina diarias por parte de los encargados del área de producción. Siempre que son detectadas fugas o que se necesitan hacer mantenimientos obligados, los empleados encargados de las inspecciones emiten notificaciones a través de un canal de radio hacia las oficinas de producción, ahí son registradas las fallas por quien se encuentra en esa terminal, enseguida son pasados los reportes a los encargados de mantenimiento que a su vez asignan los problemas que necesiten arreglar. Otra forma de emisión de fallas, es la realizada por los habitantes de la ciudad, a través de líneas telefónicas para que, una vez que cualquier usuario detecte una falla, pueda notificar al centro de atención a clientes con que cuenta el organismo.

El *OOAPAS* tiene como política atender todos los reportes de fuga en un plazo no mayor a 24 horas, cabe señalar que el proceso interno para el control de fugas resulta ineficiente, provocando así tiempos grandes para su atención y reparación, habiendo ocasiones que se pasan por alto las fugas hasta una segunda notificación del problema, es por ello, que se justifica esta investigación y considera al OOAPAS como su caso de estudio.

1.5 Beneficios esperados

- Conocer las ventajas y desventajas de las metaheurísticas para problemas de análisis combinatorio.
- Proponer un algoritmo que permita el análisis de la información proporcionada por el organismo y lograr así, de forma eficiente, la solución al problema de fugas que hay en la ciudad.
- Probar que los algoritmos seleccionados ofrecen buenas soluciones para el problema en cuestión.
- Implementar una solución factible para el problema de enrutamiento y de programación de tareas, considerando el problema de fugas hidráulicas.
- Automatizar la identificación, registro y atención de las fugas hidráulicas que son detectadas.
- Mantener un control histórico acerca de las fugas hidráulicas que ocurren y llevar el registro de sus reparaciones.
- A partir de los desperdicios promedio generados por las fugas hidráulicas y la cantidad de horas estimadas para sus reparaciones, se entregará un orden de solución de fugas en el que, a partir de su seguimiento, se disminuya considerablemente el desperdicio de agua generado por ellas.
- Buscar que la solución a las fugas hidráulicas de la ciudad y los mantenimientos necesarios sean atendidos de manera eficiente, disminuyendo los tiempos de

inactividad de los empleados y aumentando la rapidez de atención a cada problema.
- Permitir hacer la asignación de personal, herramientas y materiales necesarios para una brindar una atención adecuada a las reparaciones de las fugas hidráulicas que ocurren en la ciudad.
- Mostrar que las respuestas dadas por el sistema mejoran la situación actual del organismo.

1.6 Alcances y limitaciones

- Empleo de análisis combinatorio para la adecuación de un algoritmo capaz de resolver los problemas de enrutamiento de vehículos y programación de tareas.
- El procedimiento de solución, debido al tipo de problemas en cuestión, será fundamentado en el uso de metaheurísticas.
- Desarrollo de una alternativa de solución capaz de llevar el control de procesos de las ventanas diarias de mantenimiento dentro del OOAPAS
- La propuesta considerará el manejo de información referente a los recursos utilizados dentro del Organismo (recursos humanos y materiales).
- El sistema estará orientado a ser una aplicación Web.
- Las localizaciones con las que pretende trabajar el sistema estarán dadas mediante la ubicación de puntos dentro de un mapa, mismos que serán seleccionados desde un equipo de cómputo cualquiera, siempre y cuando tenga acceso a la aplicación y a internet.
- Las pruebas que se realicen al sistema se compondrán con dos tipos de información:
    - Información con datos reales basados en los recursos con los que cuenta el OOAPAS.
    - Información con instancias de prueba para evaluar las diferentes variantes que pueda entregar el sistema.

1.7 Organización de la tesis

En el primer capítulo se muestran los antecedentes y fundamentos del problema así como el rumbo que se busca seguir para la solución del tema en cuestión.

En el segundo capítulo se describirá el marco teórico, abarca la comparación que hay entre los tipos de sistemas y un análisis sobre los algoritmos y técnicas utilizadas

especificando el estado del arte que hay respecto a los algoritmos encargados al control de procesos, trazados de ruta y toma de decisiones.

En el tercer capítulo se detalla el análisis y diseño del sistema, mostrando el ciclo de vida de la aplicación, así como la arquitectura que fundamenta su elaboración.

En el cuarto capítulo se expone todo lo respectivo a la elaboración del sistema en cuanto a su desarrollo e implementación; abarca desde la configuración del entorno de desarrollo, hasta la puesta en marcha del sistema.

El capítulo quinto comprende las pruebas y resultados. Se muestran las evaluaciones hechas al sistema así como las respectivas pruebas de consistencia o integridad en la información dentro de la aplicación para mostrar su correcto funcionamiento.

En el capítulo seis se muestran los logros alcanzados por el sistema, además de las aportaciones del mismo, mostrando así, las conclusiones que permitan evaluar los beneficios alcanzados y los objetivos conseguidos, además de plantear los trabajos futuros.

2 Marco Teórico

2.1 Introducción

El software se ha convertido, no sólo en una tecnología para simplificar las actividades de los usuarios, sino en un completo modelo a seguir para el crecimiento de negocios, empresas, ingeniería e investigación.

Un sistema es una colección de componentes interrelacionados que trabajan conjuntamente para cumplir algún objetivo. Dado que cualquier software puede ser considerado como un tipo de sistema, la ingeniería de sistemas consistirá en la actividad de especificar, diseñar, implementar, validar, distribuir y mantener sistemas o software como un todo. [**1**]

Los sistemas de información dentro de las organizaciones son diseñados e implantados en ellas, no sólo para gestionar su información sino también como medio para automatizar y mejorar sus procesos. Dependiendo de las actividades de los usuarios o las organizaciones que operan los diferentes sistemas de información, es que se podrán encontrar mayores o menores beneficios al momento de su uso. La automatización de procesos, toma de decisiones y administración de datos, son sólo algunas de las ventajas competitivas en que converge el uso correcto de los sistemas de información. Así, los sistemas de información constituyen en la actualidad un elemento clave en las organizaciones, con un importante impacto en sus procesos, estructura y cultura [**2**].

2.2 Sistemas de Información

Hoy en día la información forma uno de los pilares más importantes para las empresas; cualquier organización por pequeña que sea, depende en gran parte de ella, incluyendo las tareas de almacenamiento, procesamiento, recuperación y gestión.

Los sistemas de información dentro de las organizaciones son diseñados e implantados en ellas no sólo para gestionar la información y el conocimiento sino también como medio para mejorar los procesos empresariales y en última instancia para crear valor. Un sistema será eficiente y mejor, cuanto sea capaz de mejorar los procesos de negocios y toma de decisiones, ya que así conducirá a una mayor rentabilidad en las organizaciones además de reducir sus costes costos. [**3**]

Ya sea por su estabilidad, grado de colaboración o según el objetivo reflejado, es que pueden existir diferentes clasificaciones de los sistemas de información. Atendiendo a la lógica evolutiva que han seguido los distintos sistemas es que se presentan las siguientes

dos perspectivas: una funcional, que identifica a los sistemas por sus funciones empresariales y otra perspectiva clasificada según sus usuarios, que identifica los sistemas en términos de los principales grupos de la organización a quienes dan servicio. [**4**] [**5**]

2.2.1 Los sistemas desde una perspectiva funcional

Los sistemas, desde una perspectiva funcional, se identifican de acuerdo con sus funciones empresariales, los cuales se clasifican en:

- **Sistemas de ventas y marketing.-** Sistema cuya función es vender los productos o servicios de la organización, así como su anuncio y promoción; también corresponde identificar clientes para la empresa e identificar lo que necesitan o desean.
- **Sistemas de manufactura y promoción.-** La ocupación de manufactura y producción corresponde a la producción de bienes y servicios de la empresa; estos sistemas están relacionados con la planeación, desarrollo y el mantenimiento del área de producción, establecimiento de metas y objetivos de la empresa, adquisición, almacenamiento y disponibilidad de los materiales que se utilizan, así como la programación de uso de equipo, instalaciones, materiales y mano de obra requeridos para la elaboración y distribución de sus productos.
- **Sistemas financieros y contables.-** Su principal desempeño se basa en administrar los activos financieros de la empresa como el efectivo, las acciones, bonos y otras inversiones, con el fin de maximizar su rendimiento; asimismo, se encarga de administrar la capitalización de la empresa, por ejemplo, revisando y buscando activos financieros en acciones.
- **Sistemas de información de recursos humanos.-** Su objetivo es atraer, desarrollar y mantener el grupo de trabajo de la empresa; estos sistemas apoyan actividades como identificar empleados potenciales, administrar su información personal, además de crear programas para contribuir al desarrollo de sus aptitudes y habilidades.

2.2.2 Los sistemas desde la perspectiva de los usuarios

Aunque la perspectiva funcional es útil para comprender la manera en que funcionan los sistemas de información, la clasificación desde la perspectiva de los usuarios ayuda a mostrar, en términos de niveles o jerarquías, los enfoques de los Sistemas (Figura 1).

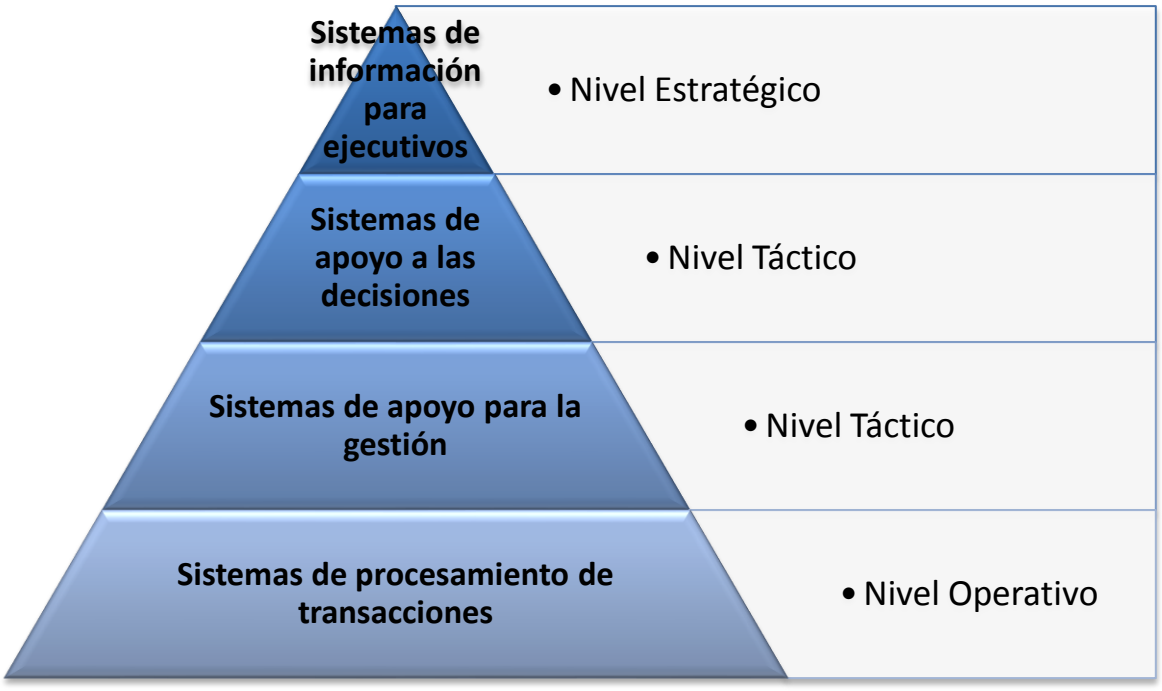


Figura 1. Clasificación de los sistemas de información [4]

Según las actividades que se manifiestan dentro de las organizaciones los Sistemas se clasifican como:

- **Sistema de Apoyo a Ejecutivos.-** Los Sistemas de Apoyo a Ejecutivos *(EIS, Executive Information System)* son sistemas diseñados para proveer análisis y evaluaciones exhaustivas de la empresa para poder generar una variedad de reportes requeridos por los directivos de la organización; así, se convierten en sistemas que abarcan aspectos estratégicos y tendencias de largo plazo para beneficio de la empresa, ofreciendo un seguimiento a los datos críticos para que los directivos puedan dar respuestas convenientes de solución.
- **Sistemas de Apoyo a la Toma de Decisiones.-** Los Sistemas de Apoyo a la Toma de Decisiones *(DSS, Decision Support System)*, siendo parte del nivel gerencial medio o administrativo colaboran en la toma de decisiones y se enfocan en problemas de naturaleza única y que cambian con rapidez, además de que por lo regular no existen procesos ni métodos predefinidos para su solución; estos sistemas se utilizan con frecuencia, para analizar la información existente y permitir a sus usuarios proyectar el efecto de sus decisiones a través del tiempo, así como sus repercusiones en el futuro.
- **Sistemas de Información Gerencial.-** En los Sistemas de Información Gerencial *(MIS, Management Information System)* se designa un servicio a la gerencia

inmediata o cargos de nivel administrativo, ya que permite generar informes acerca de la situación actual de la organización; esta información, se utiliza para supervisar y controlar la empresa, así como pronosticar su desempeño futuro. La mayoría de las veces este tipo de sistemas se apoya de los sistemas de procesamiento de transacciones para obtener datos y así usarlos en un posterior procesamiento.

- **Sistemas de Procesamiento de Transacciones.-** En los Sistemas de Procesamiento de Transacciones *(TPS, Transaction Processing Systems)* se proporciona el seguimiento a las actividades y transacciones elementales de la organización y son utilizados a nivel operativo; por lo general, son manejados por los trabajadores de más bajo nivel o personal de primera línea y participan en la recolección, almacenamiento, procesamiento y recuperación de datos; los sistemas transaccionales participan en tareas como ventas, recepciones, depósitos de efectivo, decisiones sobre créditos, entre otros, destacando así, su trabajo frecuente con operaciones en tiempo real.

2.2.3 Google Maps API

Partiendo de un servidor de mapas en la web, *Google Maps API,* es un servicio gratuito de la compañía *Google* que ofrece el manejo de imágenes de mapas, fotografías satelitales, además del manejo de rutas y ubicaciones dentro de ellas [**40**]. *Google Maps API* permite superponer datos propios sobre un mapa personalizado. Con esta plataforma se pueden crear potentes aplicaciones web y móviles manejando imágenes satelitales, vistas desde la superficies de las calles, perfiles de elevación, direcciones, mapas con estilos, demografía y una amplia base de datos de ubicaciones.

Siendo indispensable la conexión a internet es que *Google Maps API* permite el uso de una infraestructura organizacional que hace posible la adquisición y uso de datos geográficos con el fin de su análisis, síntesis y desplegado para facilitar la comprensión, planificación y gestión geográfica [**40**].

Las características frecuentes que se observan en un mapa como carreteras, lagos, montañas o construcciones son algunos de los elementos comunes que se incluyen en la herramienta *Google Maps,* la mayoría de las veces representadas por una combinación de puntos, líneas o polígonos [**40**] [**7**]. Gracias a la georreferenciación es que es posible poder manejar los puntos, líneas o polígonos en un mapa, que se refiere al posicionamiento con el que se define la localización de un objeto espacial en un sistema de coordenadas [**6**].

Google Maps API, trabaja en conjunto con herramientas tales como HTML (*HyperText Markup Language*), CSS (*Cascading Style Sheets*) y JavaScript para poder realizar su trabajo a través del uso de la web. Al igual como utilizar un mapa en la vida real, en donde

se observan imágenes, se figura una posición y enseguida se examinan rutas o áreas, *Google Maps API* envía y recibe información mediante el uso de servicios web para así poder plasmar puntos, líneas y áreas sobre imágenes de mapas en una computadora. Esta herramienta o infraestructura básicamente consiste en una serie de archivos que contienen clases, métodos y propiedades que permiten el uso de mapas para su manipulación y análisis.

Algunas de las funciones o utilerías principales de *Google Maps API* son:

- Imágenes en 45 grados.
- Autocompletado de direcciones.
- Búsqueda de lugares de interés.
- Visualización de capas demográficas.
- Matrices de distancia.
- Herramientas de dibujo.
- Perfiles de elevación.
- Codificación geográfica.
- Imágenes satelitales.
- Procesamiento KML (*Keyhole Markup Language*).
- Tráfico en tiempo real.
- Mapas estáticos
- Mapas con estilos.
- Compatibilidad con dispositivos.

2.2.4 Interrelación de los sistemas

Todos los sistemas de acuerdo con su clasificación se pueden contemplar de manera independiente, sin embargo, los sistemas de mayor jerarquía suelen usar sistemas inferiores, ya que mediante ellos, pueden recabar la información que necesitan para su trabajo.

Partiendo de la clasificación de los sistemas desde la perspectiva de los usuarios es que se presenta la siguiente interrelación entre los tipos existentes: (Figura 2)

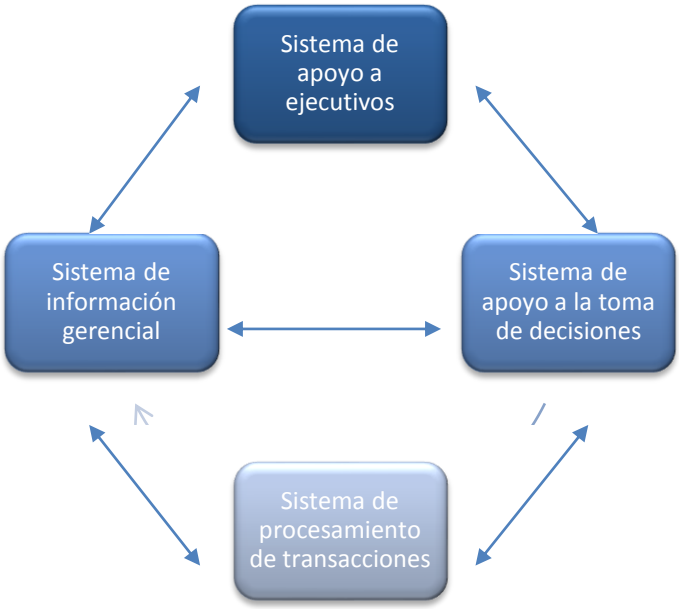


Figura 2. Interrelación de los sistemas de información [4]

2.2.5 Análisis en los Sistemas de Toma de Decisiones

Una gran parte de las organizaciones mediante el procesamiento de información, la toma de decisiones y la puesta en práctica de ellas que se realizan como parte de sus funciones cotidianas, ocasionan que día con día los *DSS* acaparen el mercado de los sistemas de información, ya que se manifiestan como una pieza fundamental para los procesos de evaluación de soluciones de las empresas.

Tradicionalmente se ha considerado que las organizaciones cuentan con un conjunto predefinido y estático de sus objetivos. Sin embargo con el fin de mantener su competitividad y sobrevivir en el campo que se especializan, es que en ellas se ha optado por mejora en la capacidad de respuesta y adaptación a los cambios que se presentan en su entorno de negocios. [**8**]

Hoy en día los *DSS* en lugar de centrarse en la información que los usuarios necesitan y sus factores de riesgo, tratan de mejorar los procesos que involucran a la decisión que se debe tomar y así reducir los sesgos en los resultados, además de ser más eficientes al momento de procesar la información que se necesita. [**9**]

La primera herramienta utilizada en los *DSS* fueron los almacenes de datos *(Data Warehouse)*, a partir de ahí se derivaron dos herramientas más que fueron el Procesamiento analítico en línea OLAP *(On-Line Analytical Processing)* y Minería de datos

(Data Mining) [**10**]. Todas estas herramientas coinciden en un mismo enfoque, el de proveer y procesar una visión histórica y unificada de los datos de una empresa.

Recientemente el campo de los *DSS* ha tenido que emplear nuevos métodos que se adapten a las necesidades del medio, así ha abarcado vertientes tales como Inteligencia Artificial, Sistemas Expertos y Sistemas Adaptativos entre otros. [**8**]

Un modelo básico para la toma de decisiones puede ser dividido en tres etapas: formulación, solución y análisis [**10**].

- **Formulación.-** Es la propuesta de una alternativa de solución. A partir de la adaptación del problema real hacia la propuesta de solución es que se esperan generar todos los puntos de optimización.
- **Solución.-** Es la etapa que comprende el algoritmo o método de la alternativa de solución previamente propuesta; aquí, es donde la fiabilidad, desempeño y certeza de los algoritmos y técnicas usadas impactarán en el éxito o fracaso de la solución.
- **Análisis.-** Etapa en la que el resultado dado por la alternativa de solución planteada, deberá ser interpretado y analizado para conocer el impacto y beneficios del sistema elaborado.

2.2.5.1 Componentes de los Sistemas de Toma de Decisiones

En un panorama general sobre los componentes de los *DSS* se debe contar con elementos de datos, módulo de procesamiento y finalmente, con la interfaz de usuario [**4**].

- **Elementos de datos.-** Este componente está formado por la base de datos del *DSS* y es un conjunto de datos históricos o actuales de diferentes fuetes externas de información, mismas que pueden provenir de otros sistemas tales como los *TPS*.
- **Módulo de procesamiento.-** Este componente está conformado por todas las herramientas de software que se usan para el análisis de datos. Contiene herramientas *OLAP*, herramientas de minería de datos, herramientas para *Data Warehouse*, algoritmos diversos, etc.
- **Interfaz de usuario.-** Este componente comprende la presentación del sistema respecto al usuario que lo utilizará. Es el medio con el que el usuario podrá interactuar con el equipo de cómputo, ya que está destinado a entregar información acerca de los procesos y herramientas de control de las aplicaciones a través de lo que el usuario observa habitualmente en la pantalla.

En la Figura 3 se muestra la manera en que interactúan los componentes de un *DSS*.

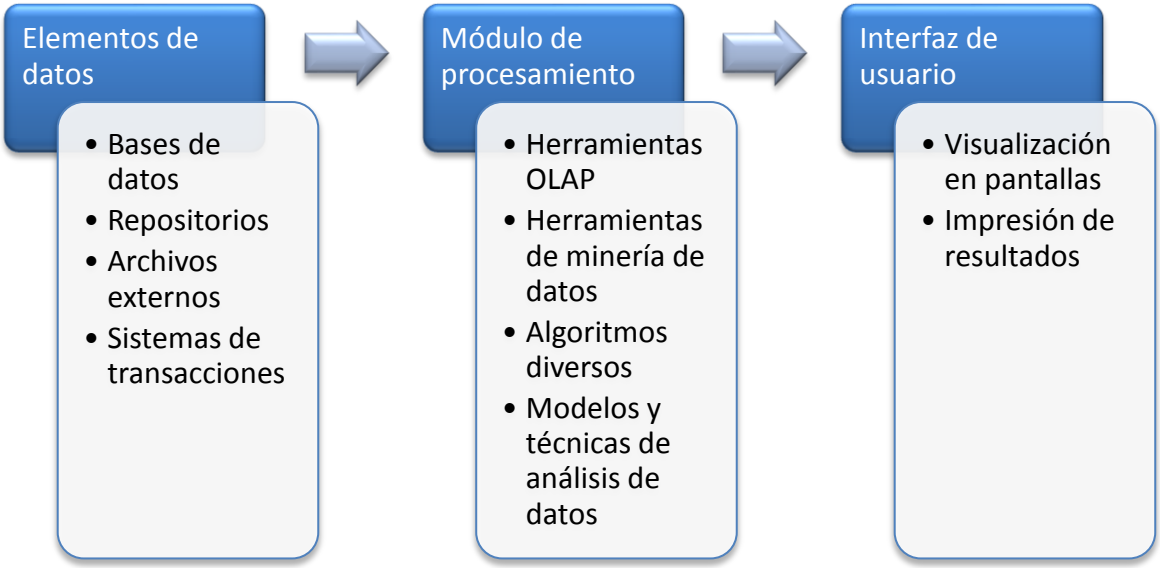


Figura 3. Interacción de los componentes de un DSS [4]

2.3 Procesos alternativos de solución

Fundamentados en la **etapa de solución** de los *DSS* surgen los algoritmos que se usan para resolver los diferentes problemas en este tipo de sistemas. Dentro de los problemas de análisis combinatorio hay dos clases de algoritmos disponibles para su solución: Algoritmos exactos y algoritmos aproximados. [**11**]

Los algoritmos exactos están garantizados para encontrar la solución óptima de un problema, además de la posibilidad de demostrarlo para cualquier instancia de tamaño finito. Por otro lado, los problemas NP-Difícil (*NP-Hard*), en el peor de los casos, el tiempo es considerado exponencial para poder encontrar la solución óptima.

Si la solución óptima obtenida no puede ser encontrada eficientemente en la práctica, la única posibilidad es tratar de buscar los mejores resultados de una manera eficiente. En otras palabras, la garantía de encontrar soluciones óptimas puede ser sacrificada en aras de conseguir buenas soluciones en tiempo polinomial. Los algoritmos aproximados a menudo llamados vagamente métodos heurísticos, son entonces aquellos que buscan obtener soluciones aceptables sin consumir tantos recursos ni gastar tanto tiempo en su ejecución. [**11**]

Los métodos heurísticos o simplemente llamados heurísticas, buscan y obtienen la mejor (casi óptima) solución a un costo computacional relativamente bajo. Basados en técnicas similares a las usadas por los algoritmos de aproximación, éste tipo de algoritmos pueden ser clasificados como métodos constructivos o de búsqueda local. Los algoritmos constructivos generan soluciones desde cero mediante la adición iterativa de componentes a una solución inicialmente vacía hasta que la solución se haya completado, es decir, construyen una solución a un problema de optimización combinatoria de manera incremental. Así, estos algoritmos paso a paso y sin dar marcha atrás agregan componentes de la solución hasta que se genera una solución completa. Por otro lado, los algoritmos de búsqueda local empiezan desde una solución inicial y repetidamente tratan de mejorar la solución actual mediante cambios locales, es decir, mediante la exploración iterativa de sectores de soluciones tratan de mejorar los resultados actuales mediante cambios de forma local. Aunque los algoritmos constructivos son típicamente los más rápidos de entre los métodos heurísticos, la calidad de las soluciones que generan la mayoría de las veces es inferior a la calidad de las encontradas por los algoritmos de búsqueda local [**11**].

Los algoritmos heurísticos han tenido un gran impacto en la solución de problemas de asignación, de programación de tareas, de transporte y de producción [**12**] [**13**]. Cuando estos algoritmos heurísticos requieren de otros métodos para la creación y evaluación de alternativas de solución son denominados métodos metaheurísticos [**14**] [**15**].

Una metaheurística puede ser vista como un método heurístico de propósito general diseñado para guiar a una heurística específica hacia regiones de solución prometedoras, es decir, espacios de solución de alta calidad [**11**]. Así, los métodos metaheurísticos hacen uso de procesos iterativos a partir de los cuales navegan en espacios de solución en donde, al aplicar reglas específicas de los métodos, convergen hacia soluciones cada vez mejores. Algunos ejemplos de metaheurísticas son: Recocido simulado, Búsqueda tabú, Búsqueda local iterativa, Algoritmos genéticos, Optimización basada en colonias de hormigas, etc.

Las metaheurísticas clasificadas como algoritmos evolutivos [**12**] o como Inteligencia de enjambre (*Swarm Intelligence*) [**16**] [**17**] son métodos de búsqueda estocásticos que imitan la metáfora de la evolución biológica natural o el comportamiento social de las especies. Ejemplos de ello incluyen como es que las hormigas buscan la mejor ruta para llegar hasta su alimento o como las aves encuentran su destino durante la migración en donde el comportamiento de cada una de las especies es guiado por su aprendizaje, adaptación u evolución. La imitación eficiente de estos comportamientos de las especies han sido estudiados e implementados en sistemas que buscan rápidas y robustas soluciones para problemas complejos de optimización combinatoria [**12**].

Existe un sinnúmero de problemas de Optimización combinatoria del tipo NP en los cuales se pueden aplicar las metaheurísticas [**18**] [**19**] [**20**] [**21**]. Muchos de este tipo de problemas pueden ser considerados dentro de las siguientes categorías [**11**]: Enrutamiento (*Routing*), Asignaciones (*Assignment*), Programación de tareas (*Scheduling*) y problemas de subconjuntos (*Subset problems*).

Dentro de cada una de estas categorías existe una amplia variedad de problemas en cada una de ellas, mismos que al ser abordados por alguna metaheurística puede ser que generen mejores o peores resultados dependiendo la solución que se realice.

Los problemas de enrutamiento, por ejemplo, son aquellos en los que uno o más agentes deben de visitar un número predefinido de localidades y cuya función objetivo depende del orden en que estas son visitadas, dos de estos problemas considerados como clásicos son: El problema de agente viajero TSP (*Traveling Salesman Problem*) [**22**] o el problema de enrutamiento de vehículos VRP (*Vehicle Routing Problem*) [**23**].

La tarea de los problemas de asignaciones es la de establecer un conjunto de "ítems" (objetos, actividades, etc.) a un número de fuentes (locaciones, agentes, etc.) sujetos a algunas restricciones. Estas asignaciones pueden ser consideradas como mapeos entre las entidades. Ejemplos de este tipo de problemas son: Problemas de asignación cuadrática (*Quadratic Assignment Problem*), Problema de asignación generalizada (*Generalized Assignment Problem*), Asignación de frecuencia (*Frequency Assignment*) [**18**] [**11**].

Los problemas de programación de tareas en un sentido amplio tienen que ver con la asignación de recursos para tareas distribuidas en un tiempo determinado. Los problemas de programación se centran en las industrias en la producción y manufacturación. Dentro de este tipo de problemas encontramos: El problema de la Tardanza Total Ponderada en una Sola Máquina SMTWTP (*Single-Machine Total Weighted Tardiness Problem*) [**24**] [**25**], JSP (*Job Shop Problem*), OSP (*Open Shop Problem*) [**11**].

En los problemas de subconjuntos, la solución se encuentra inmersa en la representación de agrupaciones de elementos bajo un cierto número de restricciones; es decir, mediante la permutación o combinación de componentes de una instancia determinada y obedeciendo las restricciones involucradas, se podrá obtener una solución para éste tipo de problemas. Algunos de los problemas de este tipo son: *Set covering*, *Set Splitting*, *Set Packing* [**18**].

En general, los algoritmos evolutivos o *Swarm Intelligence* comparten un enfoque común en las aplicaciones al resolver algún problema, la optimización, para ello, primero el problema requiere ser representado para adaptarse a cada método, enseguida el algoritmo que se vaya a usar se debe aplicar iterativamente para llegar a una solución

aceptable, así al final solamente buscar la correcta interpretación de los resultados arrojados por cada método. Una breve descripción de algunos algoritmos de este tipo se presenta en las siguientes subsecciones.

2.3.1 Algoritmos Genéticos

Un algoritmo genético simple parte de tener una población "**P**" de "**p**" individuos en el que "$\mathbf{P_x}$" representan soluciones para problemas de optimización. Aquí un proceso evolutivo tiene lugar dentro de una población de soluciones candidatas.

Los algoritmos genéticos están inspirados en una condición física de los sistemas biológicos a través de su evolución. La solución dada para el problema es representada en forma de una cadena llamada "Cromosoma", que consiste en un conjunto de elementos llamados "Genes", los cuales, mantienen al conjunto de valores para las variables de optimización. Los algoritmos genéticos trabajan con una población aleatoria de posibles soluciones (Cromosomas). La aptitud de cada cromosoma se determina mediante su evaluación de resultados en frente a la función objetivo. El fin de estos algoritmos es simular la supervivencia natural de las entidades a través de un intercambio de información a través de los cromosomas más fuertes (mediante un cruce o mutación) para producir descendencia de más cromosomas con mejores soluciones [**12**] [**16**]. Así las soluciones hijo (soluciones a partir de los "descendientes") son evaluados y utilizados para desarrollar la nueva población siempre y cuando ofrezcan mejores soluciones que los miembros débiles de la población. Por lo general, es un proceso continuo de mejora durante un gran número de generaciones para obtener el mejor ajuste como resultado [**26**].

El pseudocódigo básico de los algoritmos genéticos se muestra en la Figura 4, en el que cuatro parámetros principales afectan el rendimiento de los algoritmos genéticos: el tamaño de la población, el número de generaciones, la operación de cruza y la operación de mutación.

```
AlgoritmoGenetico(){
      Generar una población aleatoria de P soluciones (cromosomas)
      Para cada individuo i∈P calcular su Aptitud(i)
      Para cada generación
            Aleatoriamente seleccionar una operación de cruza o mutación
            Si es Cruza
                  Seleccionar dos padres aleatorios i_a y i_b
                  Generar un hijo i_c a partir de Cruza(i_a y i_b)
            Si es Mutación
                  Seleccionar un cromosoma i de manera aleatoria
                  Generar un hijo i_c a partir de Mutación(i)
            Calcular la Aptitud de i_c
            Si i_c es mejor que el peor cromosoma que se tiene
                  Reemplazar el peor cromosoma que se tiene por i_c
      Revisar si Terminación()=verdadero
}
```

Figura 4. Pseudocódigo de los algoritmos genéticos [12]

Ventajas y desventajas de Algoritmos Genéticos:

- ✓ Permite realizar una búsqueda con mayor intensidad en un espacio en particular caminando a través de las soluciones vecinas [**26**].
- ✓ Aunque las primeras soluciones se realizan de manera aleatoria, las siguientes soluciones irán produciendo mejores resultados hasta completar la evaluación de las generaciones de la población [**27**].
- × Necesita un bastante número de iteraciones para poder encontrar buenos resultados. En caso de no haber seleccionado un número correcto de generaciones entonces caerá en óptimos parciales.

2.3.2 Optimización de Enjambre de Partículas

Los algoritmos de Optimización de Enjambre de Partículas PSO (*Particle Swarm Optimization*) se originaron como una simulación de un sistema social simplificado de la coreografía elegante pero impredecible de una parvada de aves [**28**]. Estos algoritmos imitan la comunicación que hay entre la parvada al momento de su vuelo. Cada ave mira en alguna dirección en específica, al comunicarse con las otras aves identifican a la que se encuentra en una mejor ubicación. En consecuencia cada ave acelera hacia la mejor ave usando una velocidad que depende de su posición actual. Cada ave, a continuación, investiga el espacio de búsqueda a partir de su nueva posición, finalmente el proceso se debe repetir hasta que se alcanza el destino deseado.

En *PSO* cada solución es un "pájaro" de la parvada y es conocido como una "partícula". Una partícula sería el análogo a un cromosoma (miembro de una población) en los algoritmos genéticos. A diferencia de los algoritmos genéticos en *PSO* no se crean nuevas

aves de los padres, sino que las aves de una población sólo evolucionan sobre su comportamiento social y en consecuencia su movimiento hacia un destino [**12**].

El pseudocódigo del *PSO* se muestra en la Figura 5.

```
MetaheurísticaPSO(){
      Generar una población aleatoria de N soluciones (partículas)
      Para cada individuo i∈N calcular la Aptitud(i)
      Inicializar un valor de factor de peso w
      Por cada partícula
            Asignar pMejor como la mejor posición de la partícula i
            Si Aptitud(i) es mejor que pMejor
                  pMejor(i)=fitness(i)
      Asignar gMejor como la mejor aptitud de todas la partículas
      Por cada Partícula
      Calcular la velocidad de la partícula
      Actualizar la posición de la partícula
      Actualizar el valor del factor de peso w
      Revisar si Terminación()=verdadero
}
```

Figura 5. Pseudocódigo de la metaheurística *PSO* [12]

Ventajas y desventajas de Optimización de Enjambre de Partículas:

- ✓ En PSO aunque se requiere de una cierta "evolución", a diferencia de los algoritmos genéticos no necesita cálculos de mutación ni cruza en sus elementos, sino que la evolución de la cual trata sólo se refiere a una evolución en el comportamiento de sus elementos y en consecuencia su movimiento hacia un destino [**17**].
- ✓ Tiene muy bien desempeño en áreas multi-objetivo, optimización dinámica y manejo de restricciones.
- × El método sufre fácilmente la caída a óptimos parciales, lo que lo hace menos exacto en la regulación de la velocidad y la dirección de los elementos del algoritmo.
- × El método no puede resolver problemas de dispersión (*Scattering problems*), ni tampoco puede resolver problemas de sistemas sin coordenadas tales como el campo de energía (*Energy Field*) ni las reglas del movimiento de partículas en un campo de energía (*The moving rules of the particles in the energy field*) [**17**].

2.3.3 Búsqueda Tabú

La búsqueda tabú es un procedimiento metaheurístico utilizado para manejar un algoritmo heurístico de búsqueda local y así evitar que el proceso se detenga en un

óptimo local. Por lo tanto, la búsqueda tabú realiza un exploración a través de un espacio de configuraciones delimitando adecuadamente los óptimos locales [**26**].

Este tipo de algoritmo lo que hace es mantener una lista de puntos en el espacio del problema que han sido evaluados más mal, a partir de ello alejarse lo más que se pueda de ellos para poder hacer la búsqueda de su solución [**16**]. Así mientras se realiza la búsqueda de la solución gracias a mantener la lista de malos resultados es que se evita que el proceso regrese a óptimos locales y entre en ciclos repetitivos. El algoritmo considera a estos movimientos como "movimientos tabú" y así prohíben que una configuración sea visitada nuevamente.

La idea es ir haciendo movimientos sobre el espacio de soluciones, cuando un movimiento ha sido clasificado como tabú y después de ser analizado produce una función objetivo mejor que un valor de referencia escogido (que puede ser una incertidumbre u otra buena solución previamente encontrada), entonces se aplica un criterio llamado "regla de aspiración" que consiste en cancelar la prohibición de un estado y volver a aceptar el movimiento para andar sobre ese espacio de soluciones [**26**].

Más en concreto, la búsqueda tabú se basa en la premisa de la solución de un problema a partir de una memoria adaptativa y exploración sensible. La función de memoria adaptativa de la búsqueda tabú permite la aplicación de procedimientos capaces de buscar en un espacio de solución de manera económica y efectiva dado que las opciones locales de búsqueda se rigen por la información recopilada durante la búsqueda en curso. El énfasis en la exploración de una solución usando la búsqueda tabú ya sea con una implementación determinista o probabilista se deriva de la superposición de que una mala elección estratégica puede proporcionar más información que una buena elección al azar. En un sistema que utiliza memoria, una mala elección basada en estrategia puede proporcionar pistas muy útiles acerca de que tan rentable será cambiar de estrategia [**29**].

Ventajas y desventajas de Búsqueda tabú:

- ✓ Evita las malas soluciones mediante una lista de "Movimientos tabú" [**26**].
- ✓ Puede conseguir soluciones en muy poco tiempo [**29**].
- × A pesar de que opera en muy poco tiempo no siempre encuentra las mejores soluciones [**29**].

2.3.4 Optimización basada en Colonias de Hormigas

La Optimización basada en Colonias de Hormigas ACO (*Ant Colony Optimization*) es una de las más recientes técnicas para la solución de problemas de optimización. Esta técnica se fundamenta en el comportamiento de las colonias de hormigas. La base de este comportamiento es la comunicación indirecta entre las hormigas cuando realizan

aleatoriamente la búsqueda de sus alimentos, dejando por su paso sustancias químicas llamadas feromonas que posteriormente utilizarán para hacer el marcado de rutas desde su nido hasta el lugar donde puedan recolectar su comida.

ACO en una metaheurística en la cual una colonia de hormigas artificiales colabora para encontrar buenas soluciones a los problemas de optimización. La colaboración es el componente clave del diseño de los algoritmos *ACO*. La idea es asignar los recursos computacionales para establecer agentes relativamente simples (hormigas artificiales) que se comuniquen indirectamente a través de su entorno. Así es como *ACO* permite obtener buenas soluciones como una propiedad emergente de la interacción cooperativa de sus agentes [**11**] [**30**].

Primero se debe generar un conjunto finito de componentes de solución. Enseguida hay que definir un conjunto de valores de "feromonas" que no son más que valores del modelo que permitirán usarse como parámetros probabilísticos. El modelo de las "feromonas" se usará para generar probabilísticamente soluciones para el problema. En general, el enfoque *ACO* intenta resolver los problemas de optimización mediante la iteración de los siguientes pasos [**31**]:

- Las soluciones candidatas se construyen utilizando un modelo de feromonas, es decir, una distribución de probabilidad parametrizada sobre el espacio de soluciones.
- Las soluciones candidatas se utilizan para modificar los valores de feromonas de una manera que se considerará para las soluciones futuras de alta calidad.

Informalmente, un algoritmo ACO puede ser imaginado como una interacción de tres etapas: Construcción de soluciones, Actualización de Feromonas y Acciones [**11**]. (Figura 6)

- **Construcción de soluciones.-** Etapa en la que, de manera concurrente y asíncrona, las "hormigas artificiales" visitan el espacio de solución a través de los nodos vecinos dentro del grafo inicial del problema. Las hormigas se mueven por el grafo mediante decisiones probabilísticas en las que interfieren los valores de los nodos y las aristas, haciendo uso también de los rastros de feromona depositados en los diferentes nodos del grafo y por la información heurística que se tiene. Una vez que la hormiga ha construido una solución, las hormigas deberán evaluar las soluciones parciales para posteriormente ser usadas por el procedimiento "Actualización de Feromonas" para saber cuál será la cantidad de feromonas a depositar.
- **Actualización de Feromonas.-** Etapa en la que el rastro de feromonas es modificado. El valor de feromonas aumenta conforme las soluciones de las hormigas son buenas, pero también puede disminuir debido al proceso llamado

"Evaporación de la feromona" que es un decremento en la cantidad de feromonas en las aristas del grafo y cuyo objetivo es evitar la caída en óptimos locales.

- **Acciones.-** Son los procedimientos que no tienen que ver con el desempeño de las hormigas, sino con las optimizaciones locales y la manipulación de la información relacionada con la naturaleza del problema en cuestión.

```
MetaheurísticaACO {
        ConstrucciónDeSoluciones
        AcatualizaciónDeFeromonas
        Acciones
}
```

Figura 6. Pseudocódigo de la metaheurística *ACO* [11].

Ventajas y desventajas de Optimización basada en Colonia de Hormigas:

- ✓ Por la naturaleza que compone al problema, dado que utiliza a agentes (hormigas artificiales) de manera independiente, es fácil de paralelizar.
- ✓ Retroalimentación positiva para la búsqueda de mejores soluciones [**17**].
- ✓ Se adapta a los cambios en el planteamiento del problema de una manera dinámica, es decir, en tiempo real si la función objetivo cambiara, el algoritmo se podría adaptar fácilmente.
- ✓ Debido a que el algoritmo está basado en cómo las hormigas parten de su nido en busca de un destino, su alimento es que se tiene una mayor ventaja cuando los problemas presentan una fuente y un destino previamente definidos [**17**].
- × El análisis teórico es muy difícil [**17**].
- × Utiliza secuencias de decisiones basadas en funciones aleatorias

En la Tabla 1 se muestran los resultados sobre un grafo de ejemplo con 8 nodos:

Tabla 1. Comparación entre metaheurísticas para un problema de 8 nodos [27].

Algoritmo	Iteraciones	Resultado (Valor a minimizar)
Optimización basada en Colonia de Hormigas	10 Ciclos	1490
Búsqueda tabú	14 Iteraciones	1490
Algoritmos genéticos	125 Generaciones	1490
Optimización de Enjambre de Partículas	625 Propuestas	1490
GRASP (Procedimiento de búsqueda adaptativa aleatoria Greedy)	1000 Iteraciones	1490

Como se puede observar todas las técnicas obtuvieron los mismos **resultados**, en lo único que difirió el uso de cada algoritmo fue en la cantidad de iteraciones, ya que el que requirió menor esfuerzo fue la metaheurística *ACO*. Cabe señalar que, en esencia, cada algoritmo realiza "iteraciones"**,** pero de manera particular, cada uno de ellos, las nombra de manera diferente debido al contexto que manejan.

2.4 Análisis combinatorio

Una entrada para un problema computacional será codificada como una cadena finita binaria **s,** cuyo tamaño estará dado por **|s|.** Por su parte un problema de decisión será aquel que con un número determinado de cadenas responderá con un "Si" o un "No" dependiendo de la cadena de entrada que se tenga. Se dirá que un algoritmo **A** para resolver un problema de decisión correrá en tiempo polinomial si hay una función polinomial **p( )** que para cualquier cadena de entrada **s** el algoritmo **A** terminará sobre **s** a lo más en **p(|s|)** pasos.

Por otro lado un algoritmo puede ser verificado eficientemente independientemente si puede ser resuelto eficientemente. Un algoritmo verificador para un determinado problema tiene diferente estructura sobre un algoritmo que busca resolver un problema, ya que con el fin de revisar una solución, sólo se necesitará una cadena de entrada que contenga evidencia de que el algoritmo responderá "Si" al momento de su solución [**20**].

Se dice que un **verificador** eficiente para un problema será aquel que al conocer una cadena **s** podrá determinar en tiempo polinomial si esta pertenece al espacio de soluciones del problema en cuestión. Así definiremos que un problema es **NP** si perteneces al conjunto de problemas para los cuales existe un **verificador** eficiente.

Un problema ***NP-Hard*** se definirá a partir de lo siguiente: si **X ∈ NP**, para todo **Y ∈ NP** y $Y \leqslant_p X$ (**Y** es reducible polinomialmente a **X**), en otras palabras, si cualquier problema en NP puede ser reducido a X, entonces llamaremos a X como *NP-Hard*.

Un problema de tipo *NP-Hard* es el *TSP* que se enmarca en el conjunto de problemas de optimización de análisis combinatorio y al cual, los problemas NP pueden ser reducidos polinomialmente, demostrador en [**20**].

Un problema cualquiera de optimización consiste en minimizar o maximizar el valor de una variable, es decir, es un problema en el que se busca determinar o calcular el valor mínimo o máximo en una función.

En ciencias de computación, principalmente en grafos, podemos encontrar problemas de optimización, que sin importar el problema que se desee solucionar se pueden usar técnicas de búsqueda exhaustiva para su solución, mismas que por su naturaleza tienden a poseer una complejidad temporal enorme. Este tipo de problemas utilizan las metaheurísticas como su solución, que aunque no aseguran entregar el óptimo resultado, las soluciones que brindan siempre se acercan a la solución esperada.

2.4.1 El Problema de la Tardanza Total Ponderada en una Sola Máquina

En el Problema de la Tardanza Total Ponderada en una Sola Máquina SMTWTP (*Single-Machine Total Weighted Tardiness Problem*) "**n**" trabajos deben ser procesados de manera secuencial en una sola máquina. Cada trabajo tiene asociado un tiempo de procesamiento "$\mathbf{p_j}$" un peso "$\mathbf{w_j}$" y una fecha de vencimiento "$\mathbf{d_j}$" y todos los trabajos están disponibles para ser procesados desde el tiempo cero. La tardanza para un trabajo "**j**" está definida como $\mathbf{T_j = max\{0, CT_j - d_j\}}$, en donde "$\mathbf{CT_j}$" es el plazo de ejecución en la secuencia de trabajos actual.

El objetivo de *SMTWTP* es encontrar una secuencia de trabajos, es decir, una permutación de los índices de los trabajos tal que minimice la suma de la tardanza ponderada, dada por la fórmula [**11**]:

$$\sum_{j=1}^{n} w_j T_i$$

El problema *SMTWTP* sin ningún tipo de restricción es NP-Difícil (*NP-Hard*) en el sentido estricto [**24**]. Desde la perspectiva experimental este problema de manera tajante ha tenido algunas aproximaciones a su solución utilizando diversas técnicas metaheurísticas tales como: Búsqueda tabú [**32**], Algoritmos genéticos [**33**], *Particle Swarm Optimization* [**34**], Sistemas de hormigas [**11**].

2.4.2 Problema del Agente Viajero

Perteneciendo a la clase de los problemas NP-Difícil (*NP-Hard*), el problema del Agente Viajero TSP (*Traveling Salesman Problem*) menciona que, dado un conjunto de "**n**" ciudades, todas conectadas a cierta distancia entre ellas, la tarea consiste en encontrar la ruta más corta posible capaz de visitar a todas las ciudades sólo una vez y regresar a la ciudad de origen. Asumiendo entonces que el agente comienza y termina en la ciudad de la cual partió, la encrucijada se encuentra en la búsqueda de todos los órdenes restantes de "**n-1**" ciudades, que conduce a un espacio de búsqueda de tamaño "**(n-1)!**".

Formalizando esto, se puede considerar la posibilidad de que un agente de ventas debe visitar "**n**" ciudades etiquetadas como $\mathbf{v_1, v_2, \ldots v_n}$. El agente de ventas comienza en "$\mathbf{v_1}$" (su inicio) y quiere encontrar una ruta en la cual visite todas las demás ciudades para posteriormente regresar a su inicio "$\mathbf{v_1}$". Su objetivo es encontrar una ruta que lo lleve a recorrer todas las ciudades en la menor distancia posible asegurándose de no visitar ninguna ciudad más de una vez. Para cada par ordenado de ciudades $\mathbf{(v_i, v_j)}$ se especificará un número no negativo $\mathbf{d(v_i, v_j)}$ como la distancia que hay desde la ciudad "$\mathbf{v_i}$" hasta la ciudad "$\mathbf{v_j}$". Así, el resultado esperado esta dado en función de minimizar el total de las sumatoria de las distancias que hagan posible recorrer todas las ciudades, siempre y cuando se cubran las restricciones ya mencionadas [**20**].

Dentro de este problema existe una restricción natural llamada "métrica" que está dada por las distancias entre los nodos (ciudades) en cuestión. Las longitudes de las aristas forman entonces una métrica sobre el conjunto de los vértices.

Cuando las ciudades se consideran como puntos en el plano muchas funciones de distancia son métricas. Las siguientes son tres de las métricas más utilizadas:

- ***Metric TSP.-*** La distancia entre dos puntos está dada por valores no negativos asignados sin ningún orden, referencia o especificación entre los nodos.
- ***Euclidean TSP.-*** La distancia entre dos puntos es la distancia ordinaria entre los puntos correspondientes.
- ***Rectilinear TSP.-*** La distancia entre dos puntos es la suma de las diferencias de sus coordenadas **(x, y)**.

TSP ha sido un problema típico de optimización, desde el ámbito científico ha habido varias técnicas heurísticas de solución, dentro de ellas han sobresalido: Algoritmos genéticos, Redes neuronales, Sistemas de hormigas [**14**].

2.5 Resumen

En éste capítulo se describieron los tipos de sistemas a partir de dos clasificaciones, desde una perspectiva funcional y desde una perspectiva por parte de los usuarios, se mostraron las relaciones que hay entre ellos y de manera específica, se explicaron los *DSS*.

Por otro lado, se expusieron algunos de los principales algoritmos usados en el área de análisis combinatorio y por consecuente toma de decisiones; asimismo, se detallaron dos problemas que presentan este tipo de análisis y en los cuales se sustenta esta investigación: *SMTWTP* y *TSP*.

3 Análisis y Diseño

En esta sección se describen las especificaciones del sistema a partir de los requerimientos funcionales; asimismo, se proponen los diagramas UML (Lenguaje Unificado de Modelado, *Unified Modeling Language*) para mostrar la estructura y comportamiento que deberá tener el Sistema.

3.1 Requerimientos funcionales

Dado que el objetivo general del Sistema es llevar el control de las fallas hidráulicas de una ciudad para su pronta atención, se ha tomado la decisión de dividirlo en 4 módulos: Emitir fallas, Administrar recursos, Distribuir recursos y Consultar información.

3.1.1 Emitir fallas

1. El sistema llevará el registro de las fallas hidráulicas de la ciudad.
    a. El usuario tendrá la posibilidad de hacer la petición para un reporte de falla, en ese momento recibirá en pantalla un formulario para registrar los detalles para llenar el reporte.
    b. Dado que se necesita la posición de las fallas, dentro del formulario existirá un campo, en el cual, se recogerán las coordenadas para su ubicación; asimismo, se incluirá la posibilidad de situar dentro de un mapa la ubicación de la falla.
    c. Cada falla reportada podrá identificarse de manera única.
    d. Cada reporte que se genere deberá tener una clasificación de acuerdo con el tipo de falla que se haya manifestado.
    e. El sistema almacenará la información de cada falla registrada.
    f. Conforme las fallas vayan siendo reparadas se aceptará hacer la actualización del avance y el estatus en el que se encuentra su reparación.
2. Para cada una de las fallas existentes se permitirán realizar las reparaciones pertinentes para su solución, permitiendo programar la fecha de inicio y fin de su atención.
    a. Cada reparación admitirá la asignación de materiales, herramientas y empleados para su atención.

3.1.2 Administrar recursos

El sistema controlará los recursos materiales y humanos con los que cuenta el organismo, es decir, el material con que se cuenta para las reparaciones de fallas, así como la maquinaria y herramientas para su reparación:

a. Se efectuará el ingreso de nuevos recursos, permitiendo realizar una clasificación de acuerdo con las categorías: Materiales, Herramientas y Empleados.
b. Se realizará la consulta de los elementos con los que se cuenta en el organismo.
c. El sistema hará el actualizado de las cantidades de los recursos ya existentes cada vez que se creen nuevos recursos.

3.1.3 Distribuir recursos

1. Mediante una evaluación de las fallas hidráulicas de la ciudad, la cantidad de desperdicio promedio que generan y el estimado de tiempos para su reparación, el sistema mostrará un listado con el orden para la solución de las fallas:
    a. El listado será apoyado con el uso de un mapa para la ubicación de las fugas.
2. Una vez que la fuga haya sido reparada, el sistema permitirá, mediante la evaluación de las distancias entre cada una de las fugas ya reparadas, entregar una ruta para proceder a realizar el bacheo de cada una de ellas.

3.1.4 Consultar información

El sistema podrá desplegar el historial de las reparaciones ya realizadas; además, proporcionará consultas de reportes de fallas con base en el tipo de falla que se desee buscar y al seleccionar cualquier registro de la consulta realizada, el sistema mostrará los detalles del registro de falla.

3.2 Casos de Uso

Partiendo de la decisión de dividir el sistema en 4 módulos diferentes, se tiene el diagrama general de casos de uso que se muestra en la Figura 7.

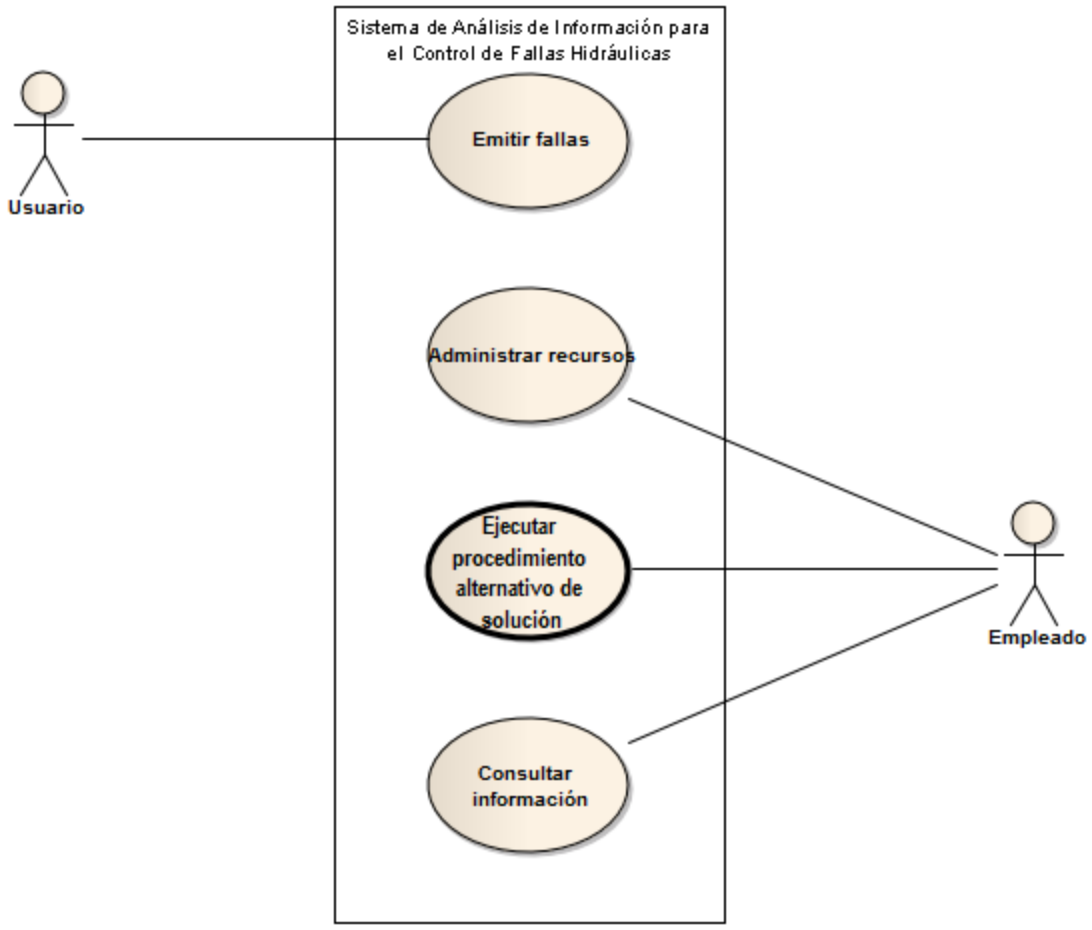


Figura 7. Diagrama de casos de uso: "Sistema de Análisis de Información para el Control de Fallas Hidráulicas".

Con base en el diagrama mostrado en la Figura 7 se mostrarán a continuación, los desgloses de cada uno de los casos de uso.

3.2.1 Caso de uso: Emitir fallas

Como se puede observar en la Figura 8, el caso de uso "Emitir fallas" se encargará de llevar el registro de las fallas hidráulicas de la ciudad.

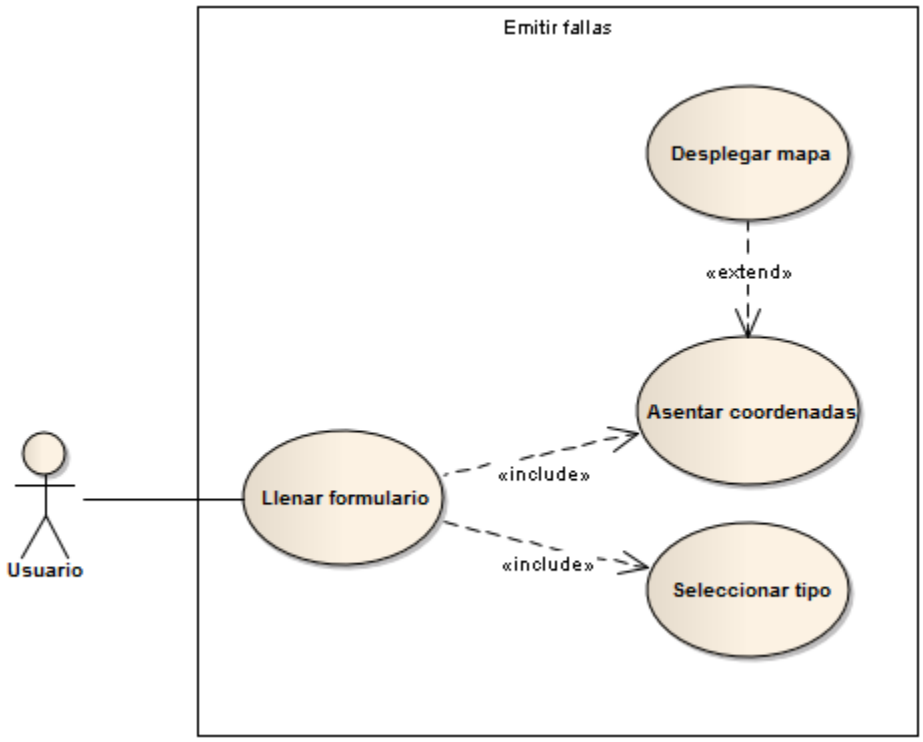


Figura 8. Caso de uso "Emitir fallas".

3.2.2 Caso de uso: Administrar recursos

En la Figura 9 puede observarse, que el sistema llevará el control de los recursos materiales y humanos con los que cuenta el organismo.

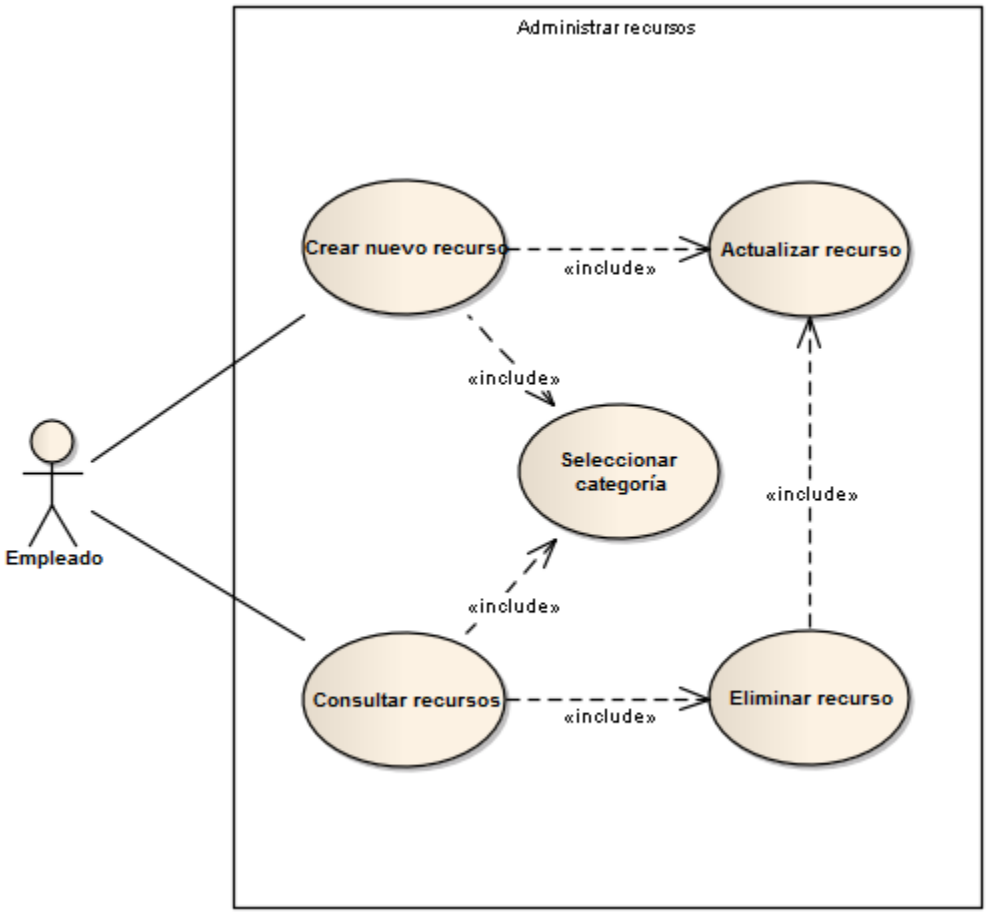


Figura 9. Caso de uso "Administrar recursos".

3.2.3 Caso de uso: Distribuir recursos

En la Figura 10 se concentra la parte medular del sistema, aquí será en donde las fases de reparación y de bacheo realicen la solución de los problemas referentes al desperdicio de agua causado por fugas.

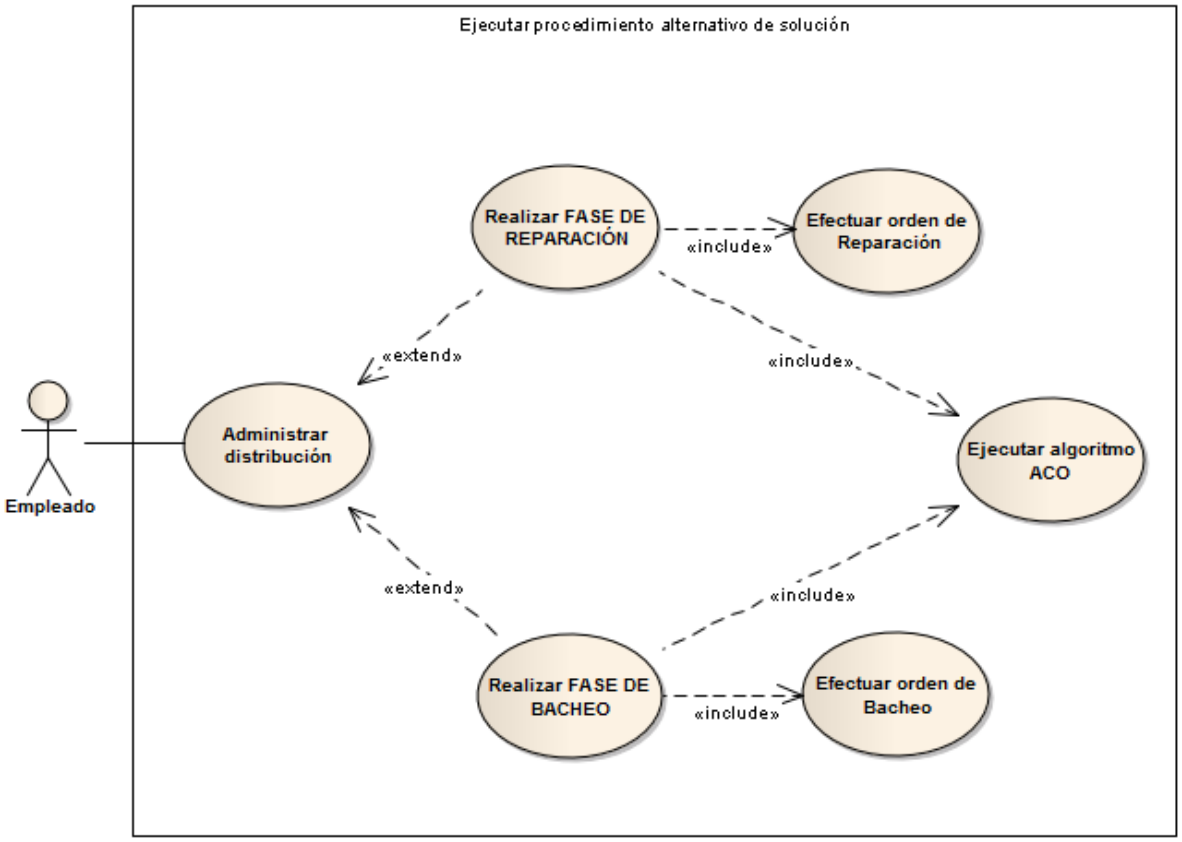


Figura 10. Caso de uso "Ejecutar procedimiento alternativo de solución".

3.2.4 Caso de uso: Consultar información

En la Figura 11 se puede observar el desglose del caso de uso "Consultar información", en el cual, se ve reflejada la administración de las reparaciones que se llevan a cabo en el sistema.

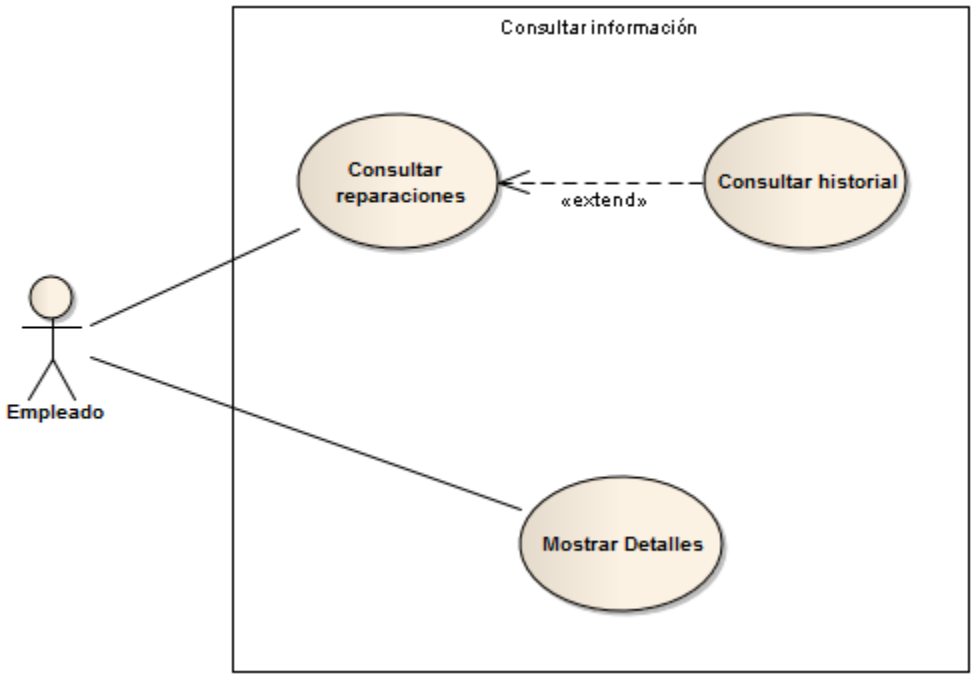


Figura 11. Caso de uso "Consultar información".

3.3 Diagrama de clases

Como se observa en la Figura 12, la aplicación estará conformada por dos clases: la parte pública que corresponderá a la emisión de las fallas por parte de los usuarios y la pantalla que se mostrará al administrador conformada por la administración y distribución de recursos del organismo; asimismo, la aplicación hará uso constante de una clase para el manejo y correcto desempeño de la base de datos.

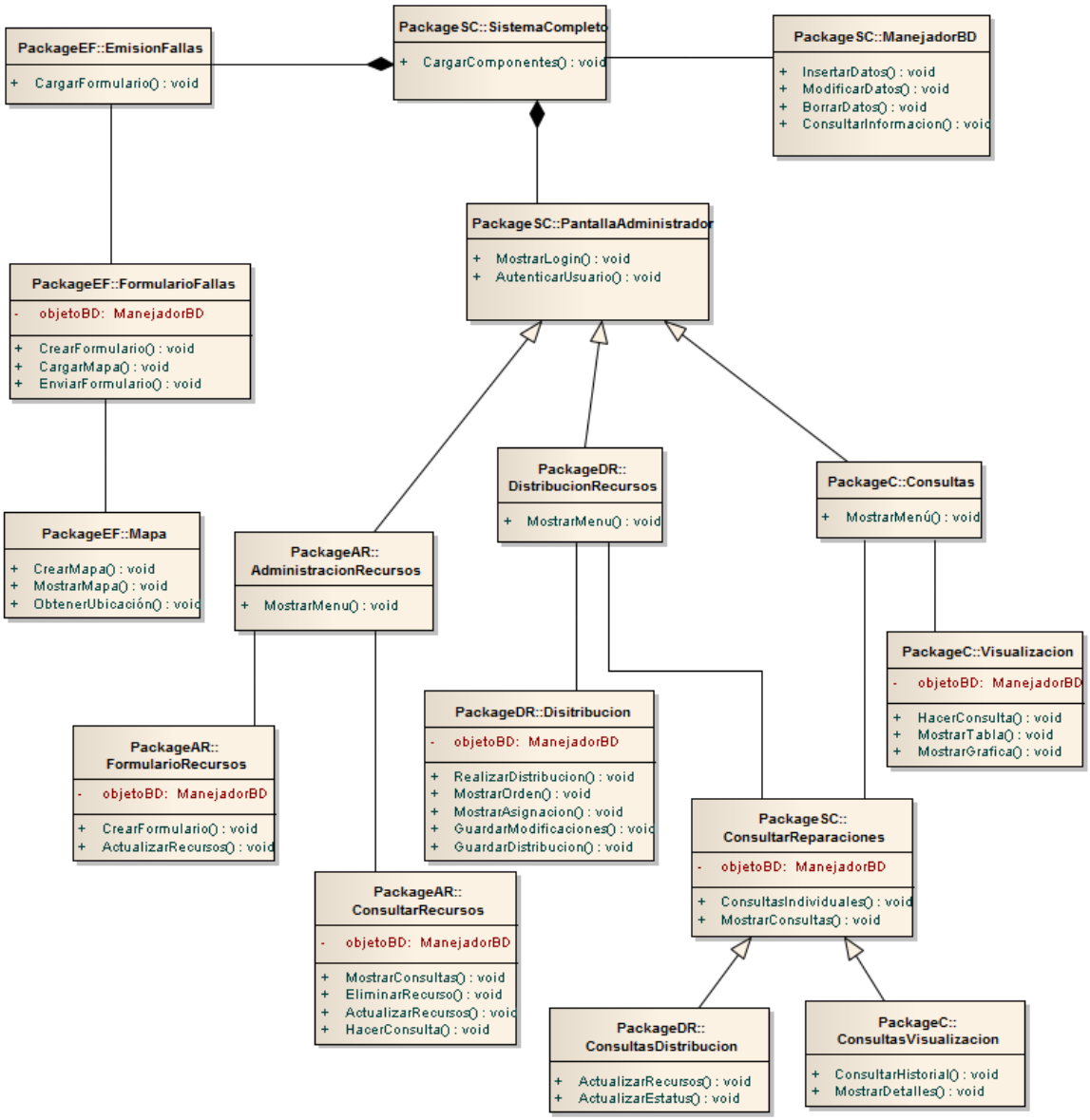


Figura 12. Diagrama de clases de la aplicación.

3.4 Diagramas de Paquetes

En el Diagrama de Paquetes que puede observarse en la Figura 13, se muestra la relación entre los cuatro módulos principales de la aplicación con el Sistema completo.

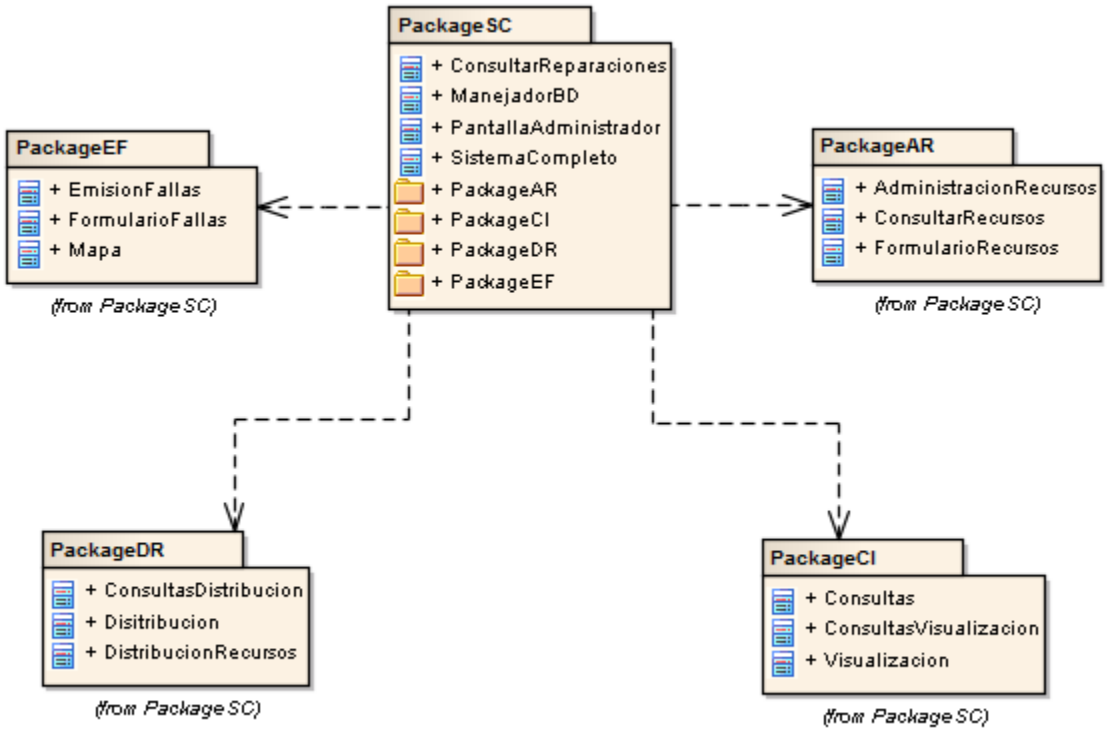


Figura 13. Diagrama de paquetes de la aplicación.

Cada paquete corresponde a la siguiente descripción:

- PackageSC.- Paquete del Sistema Completo
- PackageEF.- Paquete para Emitir Fallas
- PackageAR.- Paquete para Administrar Recursos
- PackageDR.- Paquete para Distribuir Recursos
- PackageCI.- Paquete para Consultar Información

3.5 Diagrama de Despliegue

En la Figura 14 se observa que la aplicación constará de tres partes para poder ser ejecutada: Servidor de Base de datos, Servidor web y equipo de trabajo.

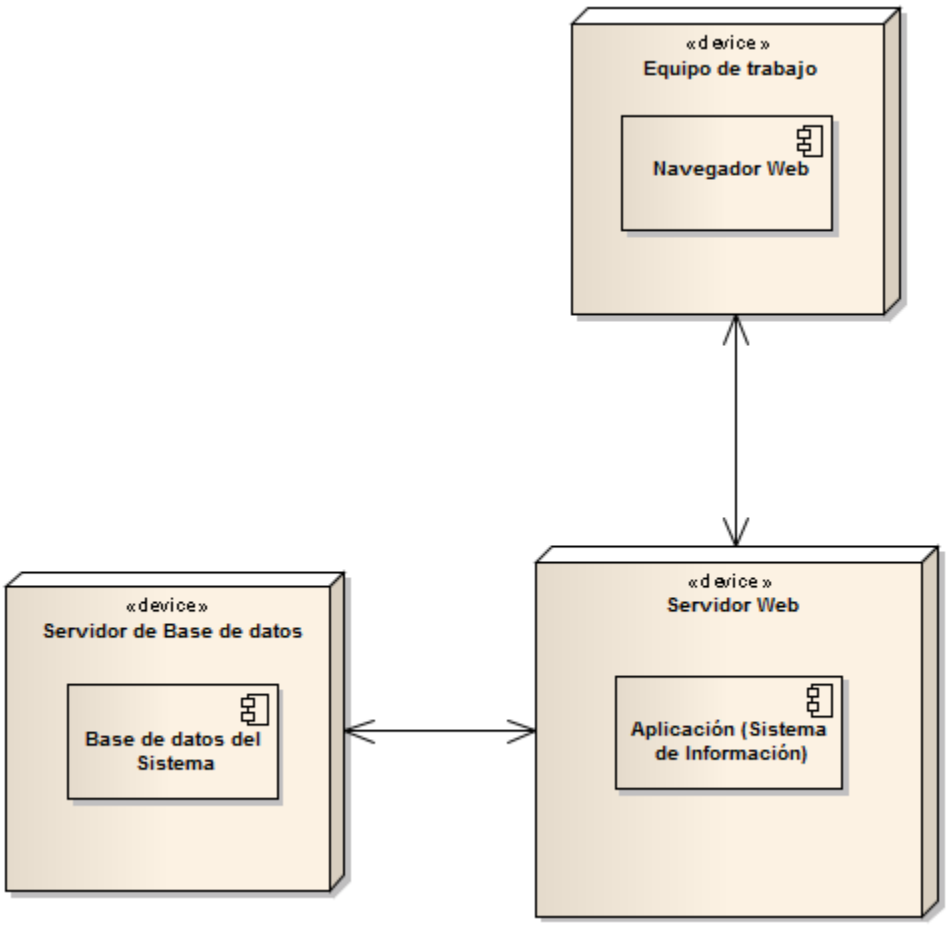


Figura 14. Diagrama de despliegue de la aplicación.

3.6 Diagramas de interacción

3.6.1 Emitir Fallas

En la Figura 15 se muestra el diagrama correspondiente al caso de uso “Emitir fallas”, es decir, la parte correspondiente al registro de fallas hidráulicas de la ciudad.

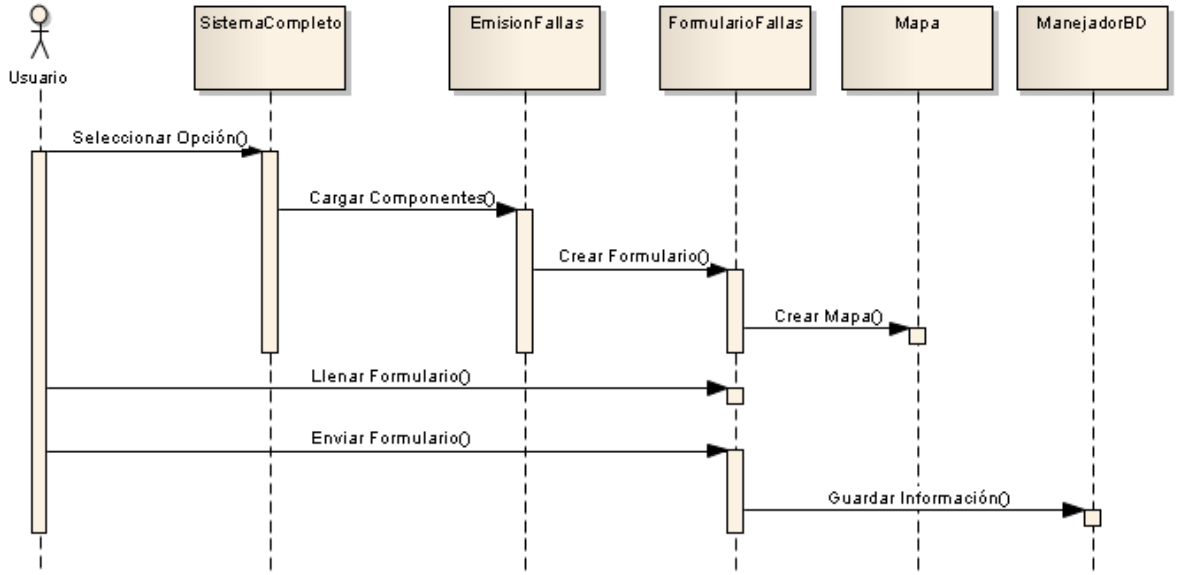


Figura 15. Diagrama de interacción del caso de uso “Emitir fallas”.

3.6.2 Crear nuevo recurso

Dentro de la sección “Administrar recursos” se mostrará cómo se llevará el control de los recursos con los que contará el organismo. (Véase Figura 16)

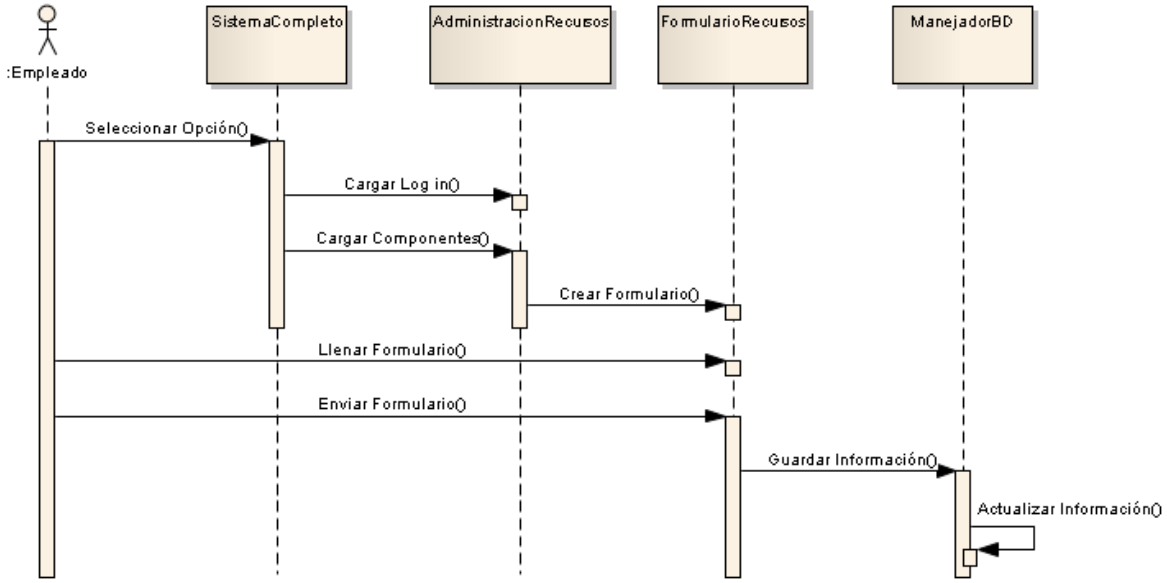


Figura 16. Diagrama de interacción del caso de uso "Crear nuevo recurso".

3.6.3 Distribuir recursos

La parte principal del Sistema corresponde al diagrama para realizar la distribución de recursos (listado de fugas y trazado de rutas) correspondientes a la reparación de cada una de las fugas hidráulicas existentes. (Véase Figura 17)

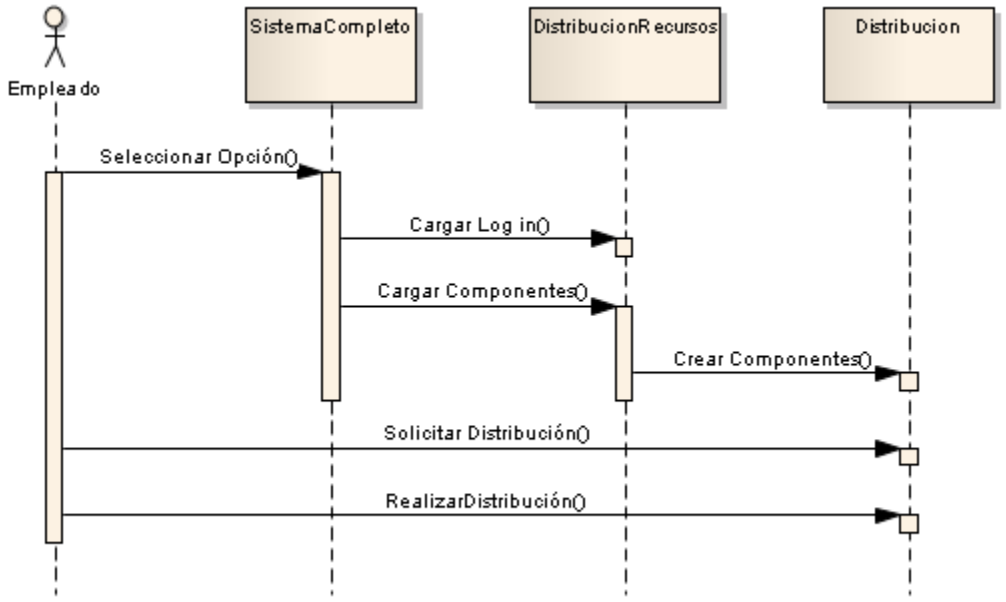


Figura 17. Diagrama de interacción del caso de uso "Realizar distribución".

3.6.4 Consultar información

En los diagramas correspondientes a "Consultar información" se mostrarán las alternativas para poder visualizar los datos dentro de la aplicación: consultar el historial de los registros y permitir opciones de detalle sobre ellos. (Véase Figuras 18 y 19)

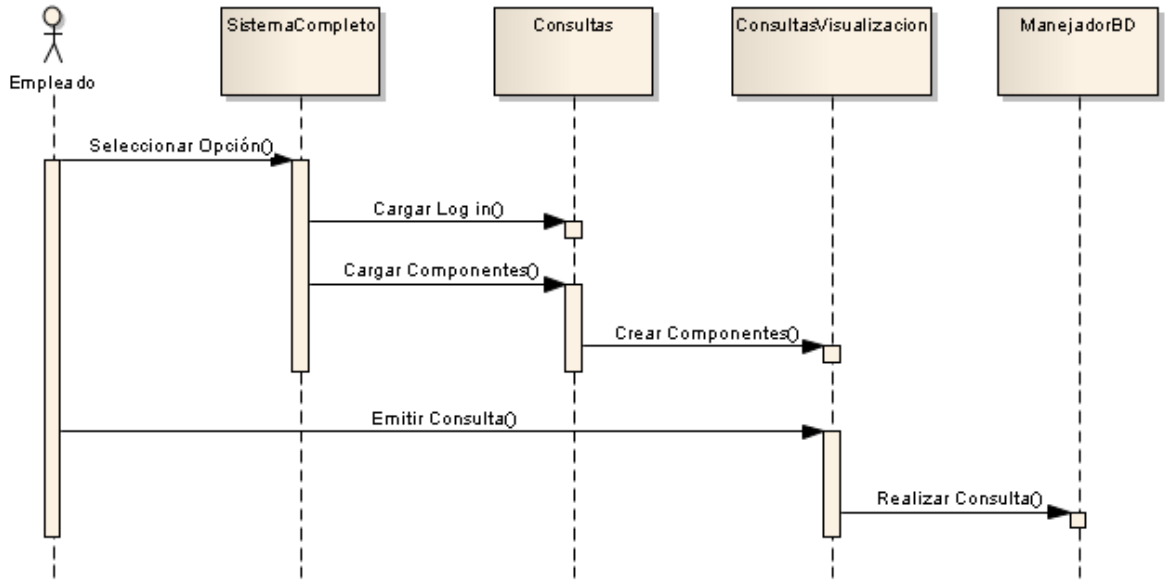


Figura 18. Diagrama de interacción del caso de uso "Consultar reparaciones".

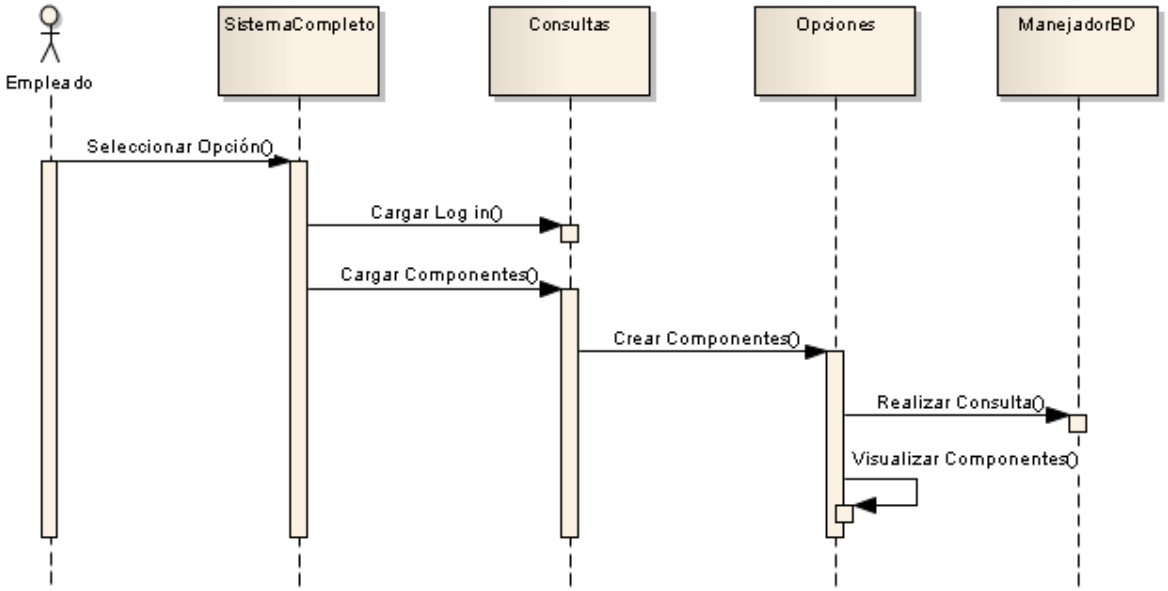


Figura 19. Diagrama de interacción del caso de uso "Mostrar detalles".

3.7 Base de datos

La Base de Datos que se utilizará estará compuesta de cinco entidades principales, cuatro de ellas para abastecer a la aplicación y una para llevar el registro de los datos una vez procesados por el algoritmo.

Las Entidades de abastecimiento del algoritmo son:

- **Materiales.-** Aquí se guardará todo el inventario de los materiales con que cuenta el organismo de agua potable.
- **Herramientas.-** Aquí se guardará todo el inventario de las herramientas que tiene el organismo
- **Empleados.-** Será la entidad que almacene el registro de todos los empleados que pertenece al área de flotillas de mantenimiento del organismo.
- **Fallas.-** Compondrá la entidad clave de alimentación del algoritmo, ya que en ella se guardará el registro de cada una de las fugas que se presenten en la ciudad, los datos indispensables de la tabla como parámetros al momento de ejecutar el algoritmo de solución de este trabajo serán:
    - Id de fuga.
    - Fecha de registro.
    - Tipo de Fuga.
    - Cantidad de desperdicio de agua generada por la fuga.
    - Tiempo estimado de reparación de la fuga.
    - Coordenadas de localización de la fuga.

La entidad que llevará el registro de datos una vez que haya sido procesada la información por el algoritmo será:

- **Reparaciones.-** Cada vez que sea ejecutado el algoritmo de solución del sistema y sea aceptada su respuesta entregada por parte del jefe de flotillas de mantenimiento, se guardará en esta entidad cada una de las reparaciones que se vayan realizando. Los datos principales que compondrán esta tabla serán:
    - Id de la fuga a reparar.
    - Fecha de inicio de la reparación.
    - Fecha de fin de la reparación.
    - Porcentaje de avance de la reparación.

En la Figura 20 se muestra el diagrama Entidad-Relación del Sistema. Asimismo, el **Anexo A** contiene el Diagrama del Modelo Relacional de la base de datos utilizada, misma que se encuentra normalizada hasta la forma normal de **Boyce-Codd**.

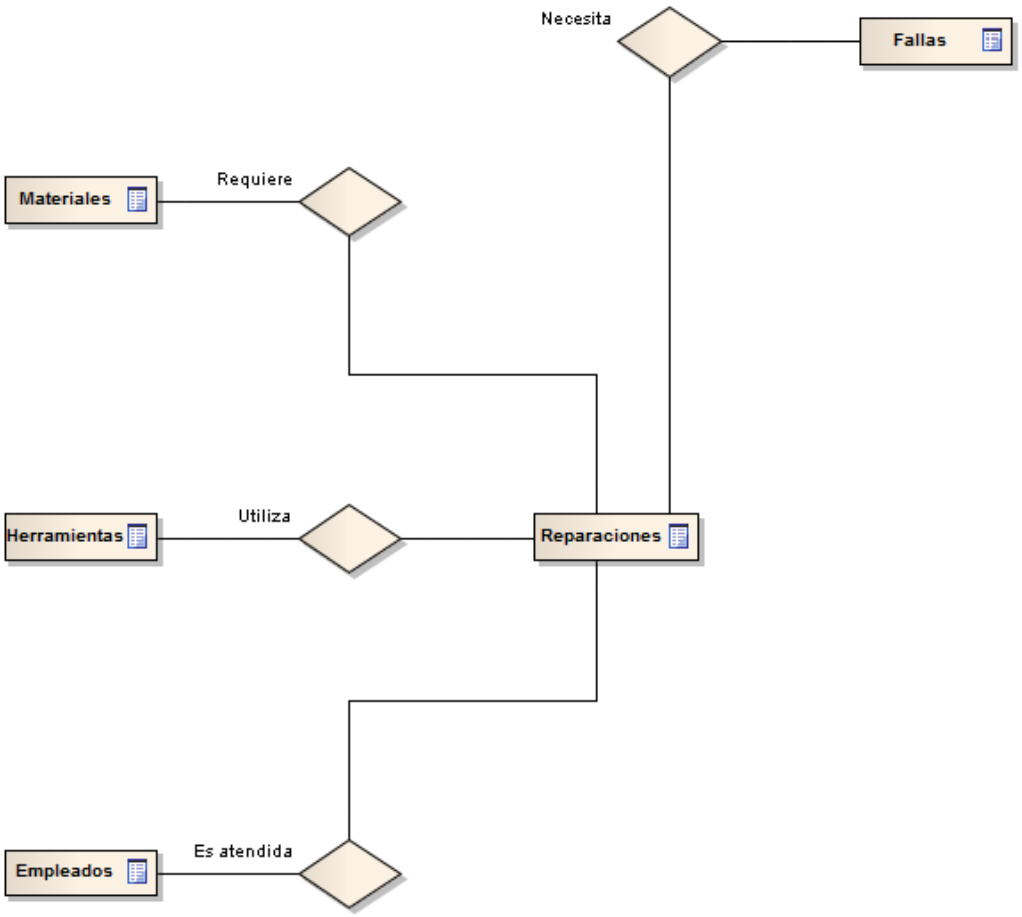


Figura 20. Diagrama Entidad-Relación de la Base de datos.

3.8 Resumen

Cualquier tipo de desarrollo de software requiere un plan de trabajo, así se permite saber sus alcances y limitaciones; a partir de requerimientos y un buen diseño, puede un Sistema tener un correcto seguimiento. En este capítulo se mostró la estructura del Sistema, detallando las partes en que fue construido.

4 Implementación

4.1 Naturaleza del problema

Es común tener reportes de pérdidas de agua en cualquier momento de un día cotidiano, pero en ocasiones no son detectables ya que los problemas que las generan se dan dentro del sistema de distribución mismo. Las pérdidas de agua son un problema que día con día ocupan más la atención de estudiantes, investigadores, trabajadores, políticos y personas en general, ya que la escasez de tan vital líquido obliga a la moderación de su consumo y por consecuencia, al cuidado en su desperdicio.

Una fuga es considerada como un escape físico y no controlado de agua en cualquier punto del sistema de agua potable, puede ocurrir en conducciones, uniones de tuberías, codos, conductos, válvulas, tanques de almacenamiento, redes de distribución, conexiones domiciliarias o dentro de las casas de los usuarios [**35**] [**36**].

En la actualidad los Sistemas Operadores de Agua Potable con los que cuenta el país se enfrentan a grandes retos tratando de ofrecer calidad a los usuarios que atienden, lo cual les obliga a mejorar los servicios de suministro, así como la atención de los desperfectos que en ellos suceden.

Una forma efectiva de conservar el agua y ahorrar dinero es mediante la reducción de pérdidas de agua potable a través de la reparación de fugas en los sistemas hidráulicos [**36**]. En cuanto a reparación de fugas se refiere, el hecho de evitar el desperdicio de agua en atención a la demanda ciudadana se convierte en una prioridad, intentando así, dar atención y solución a los reportes de fugas a la mayor brevedad posible.

De lo anterior, se puede establecer que una gran área de oportunidad para mejorar la atención de las fugas hidráulicas que ocurren en las ciudades es a través de la programación de las reparaciones de fugas, del control de rutas de atención para cada reparación, así como el manejo eficiente de recursos humanos y materiales para la reparación de los deterioros.

El propósito general en que se centra esta investigación, es la optimización en el control de fugas hidráulicas mediante el análisis combinatorio, cuya meta será la de proporcionar una programación eficiente para llevar a cabo las reparaciones de cada una de las fugas y entregar una ruta para el traslado del personal hacia las mismas; esto con el fin de poder servir como un Sistema de Toma de Decisiones de los Organismos Operadores de Agua Potable.

4.1.1 Etapas del problema

El Organismo Operador de Agua Potable, Alcantarillado y Saneamiento de la ciudad de Morelia (OOAPAS) maneja un procedimiento para atención a fugas en el que hay la posibilidad de automatizar y optimizar la toma de decisiones que en él se involucran. El departamento de reparaciones y mantenimiento define como una reparación total a aquella fuga que ha cumplido las etapas de reparación y bacheo con éxito, definiendo: *Reparación,* como la etapa que comprende detectar, perforar y corregir la fuga y *Bacheo*, la etapa que consiste en reparar los baches ocasionados durante la reparación de la fuga.

El proceso de reparación de una fuga se basa en primer lugar, en que para el OOAPAS, la ciudad está dividida en sectores, cada sector está contemplado para una brigada de reparación y una de bacheo. Cada una de las brigadas está formada por un número de empleados, materiales y herramientas previamente contemplados para las reparaciones; el responsable, es el jefe de mantenimiento, ya que se requieren dos etapas de planificación: la primera involucra al orden en que deberán ser atendidas las fugas con el objetivo de que el desperdicio de agua que surja en ellas sea el menor posible y la segunda, involucra el orden en que las fugas una vez reparadas, deberán ser bacheadas; el objetivo en ambos casos, es que la distancia que se recorra para cada una de las reparaciones para su bacheo sea la mínima.

4.2 Propuesta de solución

Construir una alternativa de solución para la optimización en el control y atención de fugas hidráulicas mediante el análisis combinatorio situada en el contexto de los Sistemas de Toma de Decisiones y basada en las etapas que involucran una reparación hidráulica en su totalidad a través de una labor dividida en dos fases: Reparación y Bacheo.

4.2.1 Fase de reparación

La primera fase involucra la programación de la atención a cada una de las fugas existentes y se realizará a partir de que la ciudad ha sido dividida en sectores por el organismo operador de agua potable en turno y cada uno de ellos, delimitará el área de cobertura de fugas, además poseerá una brigada de personal para atenderlas.

Cada fuga que se registre dentro de la ciudad será compuesta de un identificador, un tiempo de reparación estimado y una cantidad de desperdicio de agua de litros por hora.

El procedimiento que se incluye para esta fase busca que el sistema arroje una programación de reparaciones que logre disminuir el desperdicio de agua; para ello, se

propone el uso del algoritmo ACO (*Ant Colony Optimization*) como modo de solución de éste problema.

Utilizando como base el problema SMTWTP (*Single-Machine Total Weighted Tardiness Problem*) se tratará de aplicar *ACO* sobre él para su solución, haciendo una analogía con los elementos que involucran a *SMTWTP* se tiene que los trabajos **j** serán representados por cada una de las fugas a reparar, los tiempos de procesamiento $\mathbf{p_j}$ serían los tiempos que tarda en repararse cada fuga, el peso $\mathbf{w_j}$ sería la cantidad de agua que se desperdicia por cada fuga y finalmente, la fecha de vencimiento $\mathbf{d_j}$ sería la fecha máxima en consideración permitida por el organismo para hacer las reparaciones.

El algoritmo para la fase de reparación deberá tener las siguientes características:

Construcción del grafo. El conjunto de componentes **C** del grafo consistirá **n** fugas y **n** posiciones a las cuales cada fuga será asignada. Además de contener el conjunto **L** de aristas que conectarán de manera completa al grafo.

Restricciones. La única restricción que se tendrá será que todas las fugas deberán ser programadas para su atención.

Rastro de feromona. Los rastros de feromona se referirán a la conveniencia de programar una fuga para ser reparada en la posición "**i**" en el orden de reparación.

4.2.2 Fase de bacheo

Dado que la fase de bacheo comprende la corrección de los baches originados por las fugas previamente atendidas en la fase de reparación, éste es un proceso más rápido de solucionar y además, ya hay un área previa de solución.

El procedimiento a seguir programa cada uno de los bacheos de tal manera que el recorrido de la brigada sea uno de los más cortos, generando así menos gastos en gasolina o disminución de tiempo en el recorrido.

Al igual que en la fase de reparación también aquí se propone como solución del problema el uso de un algoritmo *ACO*.

Intuitivamente, el problema *TSP* es el problema en el que un agente viajero, empezando desde un punto inicial en la ciudad, quiere encontrar la ruta más corta a través de la cual visite todos los puntos especificados previamente y regrese al punto inicial [**11**]. Así, a partir del problema *TSP* y mediante el algoritmo *ACO* se buscará solucionar el problema de bacheo de esta fase del motor de toma de decisiones, ya que puede ser aplicado a *TSP* haciendo una analogía, el agente viajero se interpretará como la brigada de bacheo del

organismo, el conjunto de ciudades a visitar será el conjunto de fugas ya reparadas y finalmente, cada una de las aristas junto con su tamaño serán los caminos existentes entre cada una de las localizaciones de las fugas ya reparadas que hubiera.

El algoritmo para la fase de bacheo debe tener las siguientes características:

Construcción del grafo. El conjunto de componentes **C** del grafo consistirá **n** fugas ya reparadas y **n** posiciones a las cuales cada fuga ya reparada será asignada. Además de contener el conjunto **L** de aristas que conectarán de manera completa al grafo.

Restricciones. La única restricción que se tendrá será que todas las fugas, ahora ya reparadas, deberán ser visitadas para recibir el bacheo.

Rastro de feromona. Los rastros de feromona se referirán a la conveniencia de visitar una fuga ya reparada para ser bacheada en la posición "**i**" del orden de bacheo.

4.2.3 Proceso para realizar la Reparación y Bacheo

El proceso para realizar las fases de Reparación y Bacheo en cuanto a la solución propuesta está formado por los siguientes pasos:

1. **Selección problema (*SMTWTP y TSP*).-** El motor de la aplicación permitirá la solución de los problemas *SMTWTP y TSP*, según sea el caso, se deberá elegir una opción al momento de la ejecución del sistema.
2. **Consultar fugas por sector.-** A partir de las fugas registradas y con base en un sector de la ciudad (previamente definido por el organismo) se hará un filtrado de las fugas que participarán en el algoritmo.
3. **Generar grafo.-** Partiendo de las fugas previamente filtradas se generará un grafo totalmente conectado para poder servir como objeto base para el algoritmo.
4. **Asignar parámetros de entrada.-** Se asignarán como parámetros de entrada al algoritmo: el grafo generado a partir de las fugas existentes, la cantidad de ciclos con que trabajará el algoritmo, el número de hormigas artificiales (agentes) y el tipo de problema a resolver (*SMTWTP y TSP*).
5. **Ejecutar ACO.-** Se ejecutará el algoritmo ACO con los parámetros asignados.
6. **Interpretar solución.-** Una vez generada la solución se interpretará de la manera según convenga para cada fase, ya sea entregando un orden de ejecución para la fase de Reparación o entregando una ruta de reparación para la fase de Bacheo.
7. **Crear componentes de interfaz.-** Generar los componentes de interfaz web para ser presentados los resultados al usuario.

Cabe señalar que para ambas fases (Reparación y Bacheo) se utiliza una misma solución ya que la diferencia se encontrará en los parámetros de entrada para el algoritmo (**paso 4**) y la manera de interpretar la solución (**paso 6**).

El diagrama de la Figura 21 muestra el proceso que se tiene a seguir para la propuesta de solución que se plantea como parte medular de la investigación.

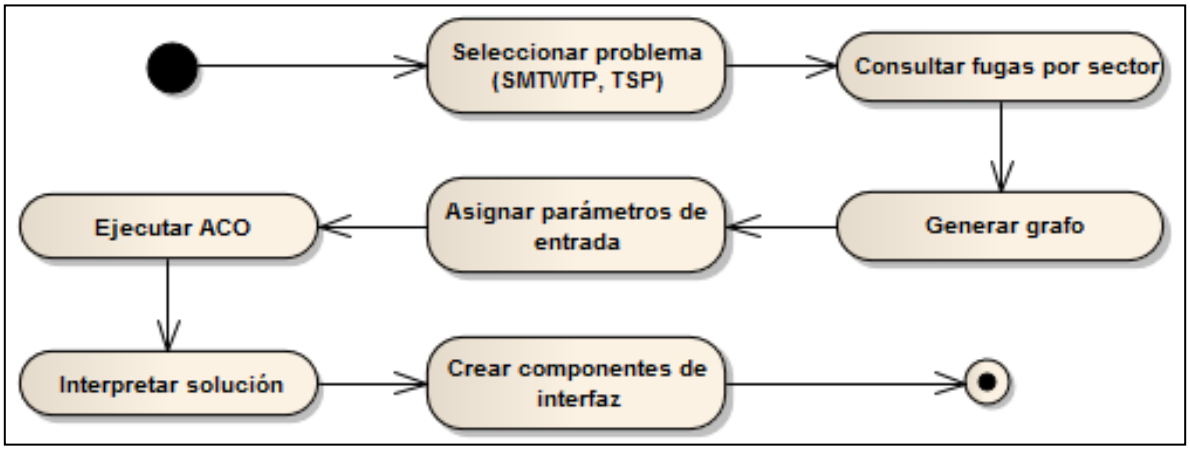


Figura 21. Diagrama de actividades referente al proceso de solución a las fases de Reparación y Bacheo.

4.2.4 Validez de ACO

Una cantidad de problemas importantes de optimización no han podido ser resueltos mediante el uso de algoritmos exactos, es decir, no ha sido posible encontrar su solución óptima con esfuerzos computacionales aceptables. Un problema de optimización ha sido el fenómeno llamado explosión combinatoria, el cual menciona que cuando crece el número de variables de decisión del problema, el número de decisiones factibles y el esfuerzo computacional crecen en forma exponencial [**26**]. Enmarcado dentro de estos problemas de análisis combinatorio se encuentran *TSP* y *SMTWTP* en los cuales muchos algoritmos han sido desarrollados para buscar su solución encontrando buenas resultados pero no siempre las soluciones óptimas. A partir de ello, las técnicas de solución se centraron en la aplicación de metaheurísticas de propósito general tales como la Optimización basada en Colonias de Hormigas.

Como ya fue mencionado en el Capítulo 2, la técnicas metaheurísticas siempre tienden a avanzar a la obtención de mejores soluciones, dado un espacio de búsqueda, siempre buscarán un mejor resultado. Tal es el caso de la metaheurística *ACO*, que aunque no se puede garantizar un óptimo resultado, se puede asegurar la búsqueda de una solución

certera en un tiempo polinomial partiendo del hecho intrínseco de que es una metaheurística.

La elección de *ACO* sobre las demás técnicas fue a partir del escenario presentado en la ciudad de Morelia, el número de fugas a analizar y los tipos de problemas que se deseaban resolver, ya que el algoritmo entregó los mejores resultados en cada una de las pruebas realizadas a los problemas de enrutamiento y programación de tareas de este trabajo. La metaheurística *ACO* cabe señalar, surgió y fue realizada justamente a partir del problema TSP, en el cual a partir de un conjunto de nodos a explorar, cada agente denominado "hormiga" haría un recorrido presentando al mejor la opción con el mejor resultado.

4.3 Implementación del Sistema

La parte esencial de este trabajo está sustentada con el análisis frente al problema de desperdicio de agua potable e inversión de tiempo para la reparación de las averías que las generan, así como la elaboración del algoritmo que pueda participar en la propuesta de la solución que más convenga para el escenario de pérdidas de agua potable de la ciudad de Morelia, Michoacán,

Una vez establecido el análisis y elección del algoritmo adecuado al problema de esta investigación, se procedió a la búsqueda de una herramienta para poder realizar la implementación.

Se determina que es necesario un Ambiente de Desarrollo que se pueda utilizar para generar aplicaciones web y un Manejador de Base de Datos que soporte tipos de datos espaciales; dos herramientas comerciales gratuitas, que cumplen las características necesarias para implementación de la solución, son: **Microsoft Visual Studio 2010** y para la base de datos, **Microsoft SQL Server 2012,** en donde ambas, al pertenecer a una misma infraestructura, permiten un acoplamiento eficaz entre ellas, además de un uso completo de sus propiedades y métodos sin la necesidad de componentes externos para el uso total de sus funcionalidades.

Por consiguiente, el Sistema fue desarrollado en el ambiente de Microsoft Visual Studio 2010 utilizando el lenguaje de programación C# bajo el Framework de Microsoft .Net Versión 4.0 (Véase Figura 22) y en cuanto a la base de datos, se utilizó Microsoft SQL Server 2012 (Véase Figura 23).

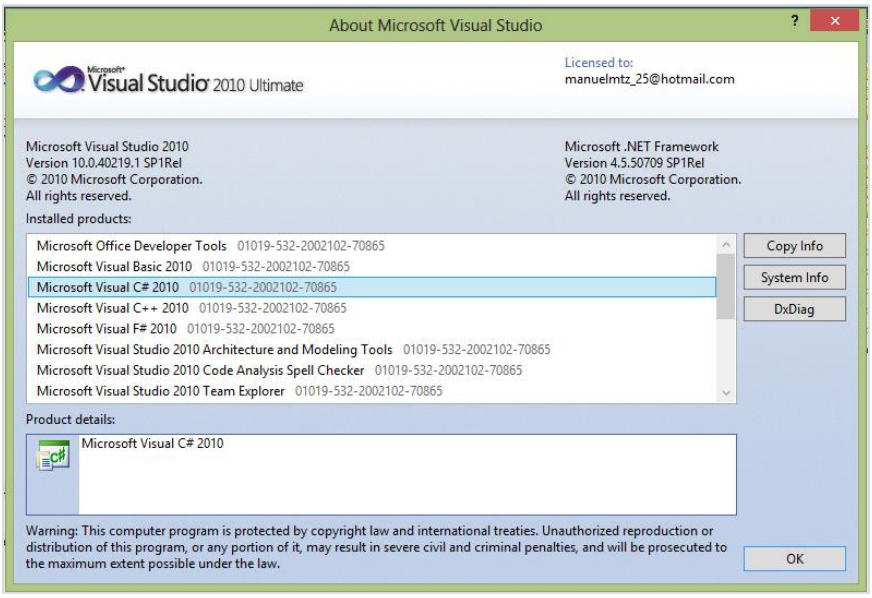


Figura 22. Ambiente de desarrollo utilizado para la elaboración del Sistema.

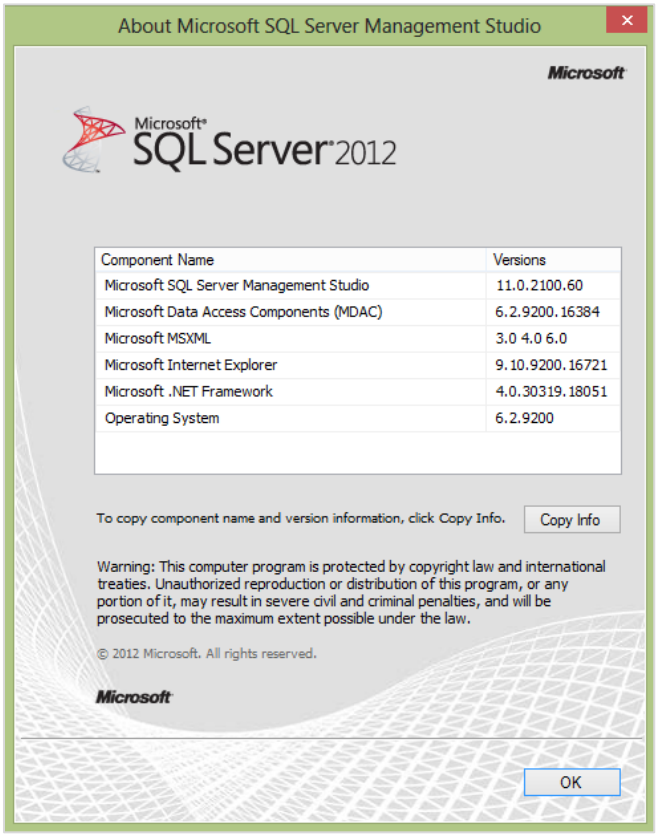


Figura 23. Manejador de Bases de datos utilizado para el desarrollo y ejecución del Sistema.

Partiendo de los requerimientos proporcionados por el organismo, el Sistema quedó conformado con tres módulos (Fallas, Recursos y Reparaciones) en donde cada uno de ellos, contiene la opción de consultas sobre la información respectiva al módulo en cuestión (Figura 24). Cabe destacar que para el módulo de "Fallas" el tipo de acceso es público, mientras que para los módulos de "Recursos" y "Reparaciones" el acceso es privado, en donde para poder acceder, previamente se debe seleccionar la opción de "Iniciar Sesión" (Véase Figura 25).

Figura 24. Pantalla de inicio del Sistema en la sesión de administrador.

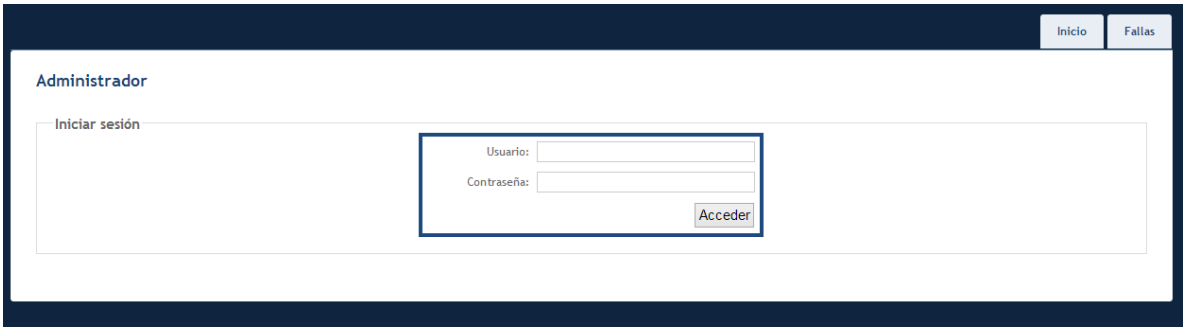


Figura 25. Pantalla de inicio de sesión.

Como parte extra de la implementación del Sistema, se cuenta con la siguiente información:

- Manual de Usuario (Anexo B).
- Manual de Instalación (Anexo C).
- Script para restaurar la Base de Datos (Anexo D).

4.3.1 Módulo "Fallas"

Módulo de nivel público, para poder acceder a él bastará con ingresar a la página web del organismo y seleccionar la opción "Fallas" y estará disponible para la sesión como Administrador; aquí, podrá llevarse el registro de todas las fallas que sean detectadas en la ciudad, además de poder consultar la información de manera individual de cada una.

Éste módulo está conformado por tres partes con las tareas de:

- Mostrar todas las fugas abiertas registradas en la ciudad sobre un mapa con base en su localización (Figura 26).
- Mostrar todas las fugas abiertas registradas en la ciudad mediante el uso de una lista con las opciones de (Figura 27):
  - Modificar.
  - Borrar.
  - Consultar detalles.
  - Asignar reparación.
- Crear registros de fugas hidráulicas una vez que éstas han sido detectadas, utilizando una ventana con campos de texto y con la opción de poder seleccionar sobre un mapa el punto de localización de ellas (Véase Figura 28).

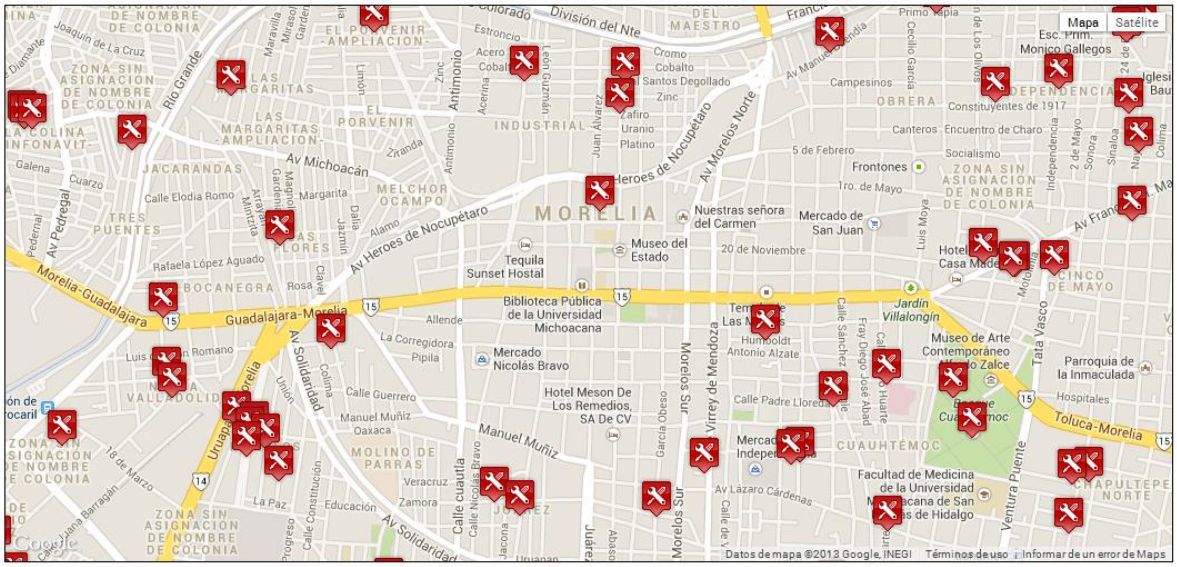


Figura 26. Sección del módulo "Fallas" con el mapa de fugas abiertas.

Mostrar 10 registros Buscar:

Falla	Emisor	Clasificacion	Gravedad	Ubicación de fuga en línea	Tipo fuga general	Estatus	Avance	Momento emisión
# 1	OOAPAS	En conexiones domiciliarias	Goteo	Válvulas	Visible	Bacheo	0	10/02/2013 12:00:00 a. m.
# 10	OOAPAS	En conexiones domiciliarias	Goteo	Válvulas	Visible	Abierto	0	01/10/2013 12:00:00 a. m.
# 100	OOAPAS	En conexiones domiciliarias	Goteo	Válvulas	Visible	Bacheo	0	03/10/2013 12:00:00 a. m.
# 101	OOAPAS	En conexiones domiciliarias	Goteo	Válvulas	Visible	Bacheo	0	03/10/2013 12:00:00 a. m.
# 102	OOAPAS	En conexiones domiciliarias	Goteo	Válvulas	Visible	Bacheo	0	03/10/2013 12:00:00 a. m.
# 103	OOAPAS	En conexiones domiciliarias	Goteo	Válvulas	Visible	Bacheo	0	03/10/2013 12:00:00 a. m.
# 104	OOAPAS	En conexiones domiciliarias	Goteo	Válvulas	Visible	Bacheo	0	03/10/2013 12:00:00 a. m.
# 105	OOAPAS	En líneas principales	Goteo	Válvulas	Visible	Abierto	0	03/10/2013 12:00:00 a. m.
# 106	OOAPAS	En conexiones domiciliarias	Goteo	Válvulas	Visible	Bacheo	0	03/10/2013 12:00:00 a. m.
# 107	OOAPAS	En conexiones domiciliarias	Goteo	Válvulas	Visible	Abierto	0	03/10/2013 12:00:00 a. m.

Mostrando del 1 al 10 de 270 registros Anterior Siguiente

Figura 27. Sección del módulo "Fallas" con la lista de fugas existentes.

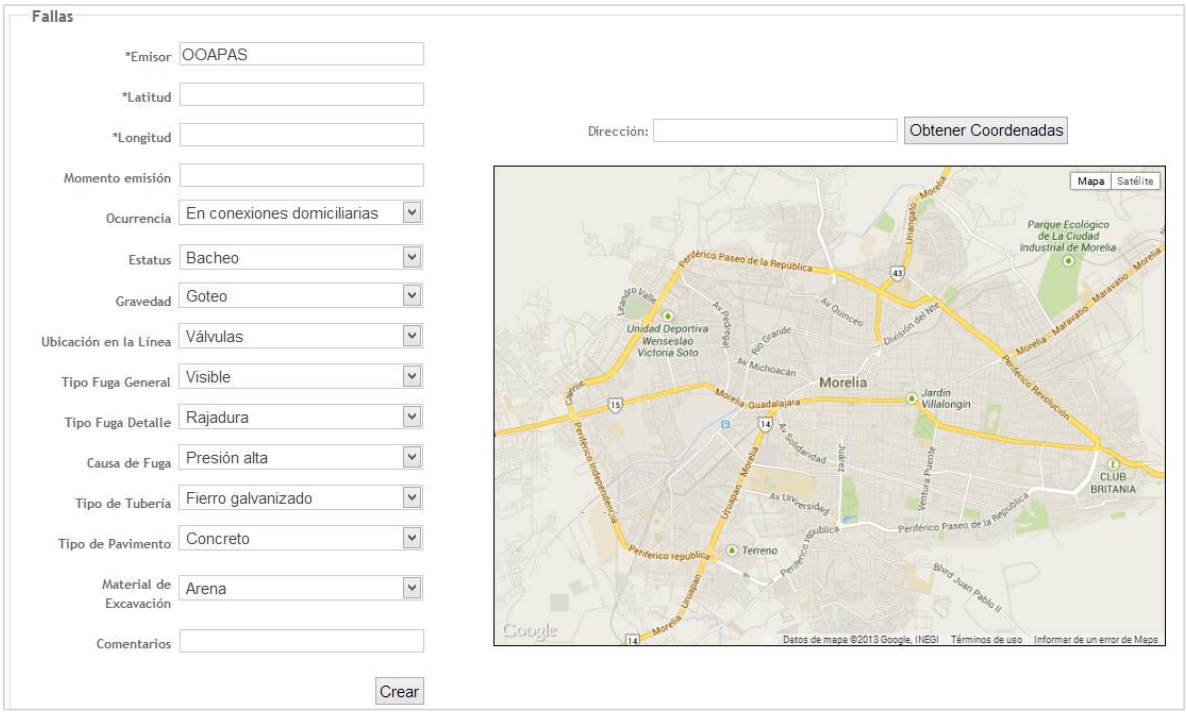


Figura 28. Sección del módulo "Fallas" para crear registros de fugas.

4.3.2 Módulo "Recursos"

Módulo privado, únicamente como administrador se puede ingresar en él y sirve para llevar el control acerca de los recursos con los que cuenta el organismo, con la capacidad de realizar altas, bajas, consultas y modificaciones de los datos involucrados. Estará compuesto por tres secciones:

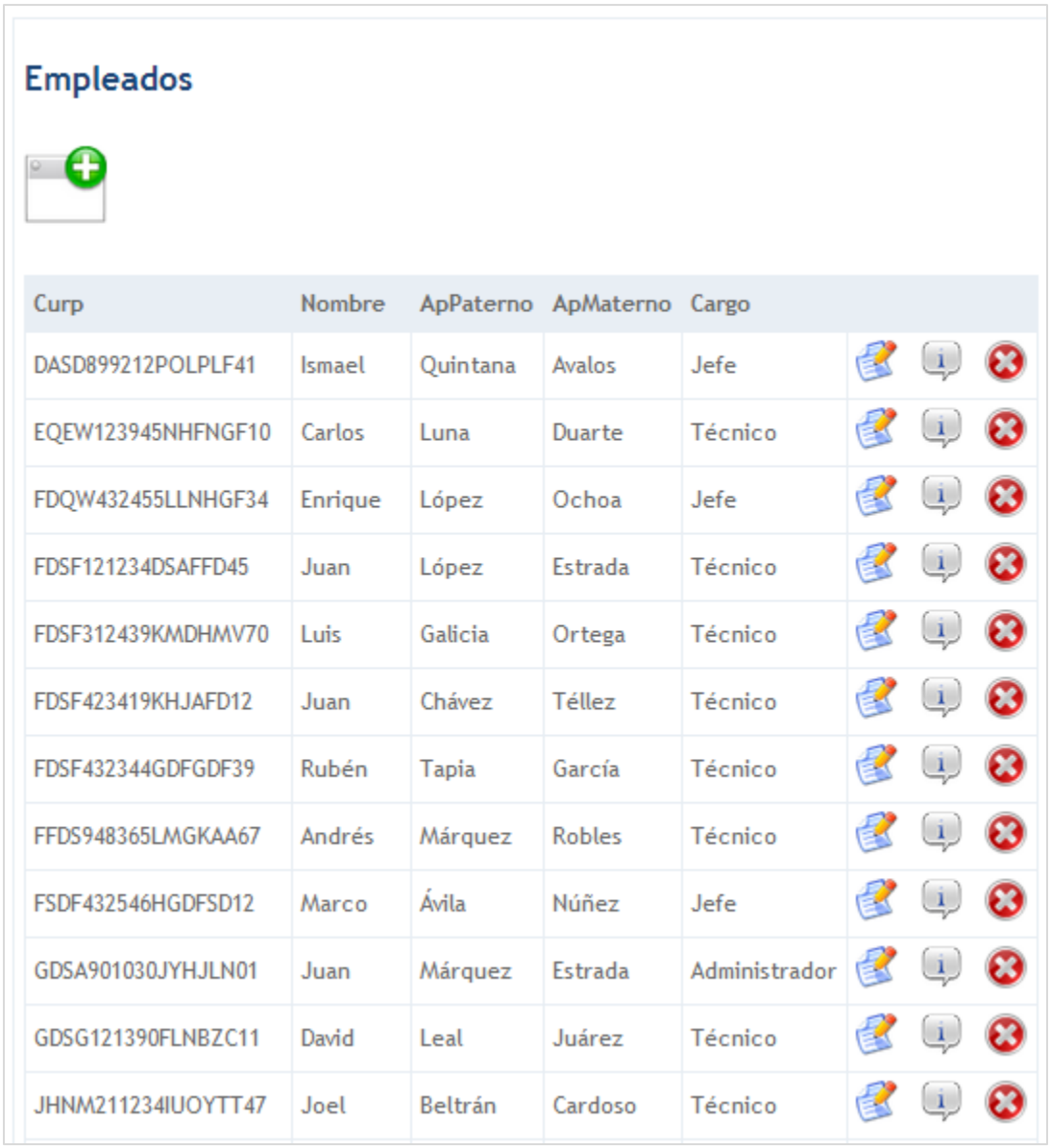
- Materiales (Véase Figura 29).
- Herramientas (Véase Figura 30).
- Empleados (Véase Figura 31).

Materiales

Nombre	Categoría	Cantidad
Anillos 10	Construcción	100
Anillos 20	Construcción	150
Cemento	Construcción	200
Mortero	Construcción	300
Pagamento	Plomería	484
Pegamento blanco	Plomería	100
Tubos cobre 10	Plomería	150
Tubos cobre 5	Plomería	100
Válvulas 10	Plomería	200
Válvulas 30	Plomería	249

Figura 29. Sección del módulo “Recursos” para el control de materiales.

Figura 30. Sección del módulo "Recursos" para el control de herramientas.



Empleados

Curp	Nombre	ApPaterno	ApMaterno	Cargo
DASD899212POLPLF41	Ismael	Quintana	Avalos	Jefe
EQEW123945NHFNGF10	Carlos	Luna	Duarte	Técnico
FDQW432455LLNHGF34	Enrique	López	Ochoa	Jefe
FDSF121234DSAFFD45	Juan	López	Estrada	Técnico
FDSF312439KMDHMV70	Luis	Galicia	Ortega	Técnico
FDSF423419KHJAFD12	Juan	Chávez	Téllez	Técnico
FDSF432344GDFGDF39	Rubén	Tapia	García	Técnico
FFDS948365LMGKAA67	Andrés	Márquez	Robles	Técnico
FSDF432546HGDFSD12	Marco	Ávila	Núñez	Jefe
GDSA901030JYHJLN01	Juan	Márquez	Estrada	Administrador
GDSG121390FLNBZC11	David	Leal	Juárez	Técnico
JHNM211234IUOYTT47	Joel	Beltrán	Cardoso	Técnico

Figura 31. Sección del módulo “Recursos” para el control de empleados.

4.3.3 Módulo “Reparaciones”

Módulo privado, únicamente como administrador se puede ingresar y en él se llevan a cabo las tareas primordiales del Sistema: Distribución y Bacheo; además se puede ver la lista con las reparaciones realizadas hasta la fecha actual incluyendo la posibilidad de agregar empleados, herramientas y materiales a cada reparación (Véase Figura 32).

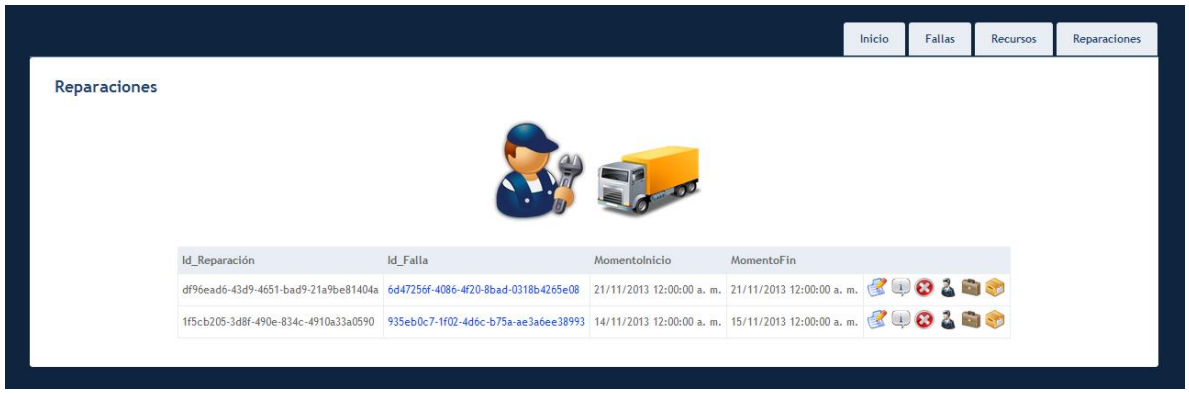


Figura 32. Pantalla principal del módulo "Reparaciones" con la lista de reparaciones realizadas.

En la sección de Distribución, utilizando como base una consulta en base de datos sobre el desperdicio de agua promedio generado por un tipo de fuga, así como la cantidad de horas estimadas para su reparación, el Sistema busca entregar una secuencia de solución de las fugas abiertas para un determinado sector de la ciudad, mostrando el orden de ejecución, características de las fugas abiertas y la cantidad de desperdicio de agua total estimado (Véase las Figuras 33 y 34).

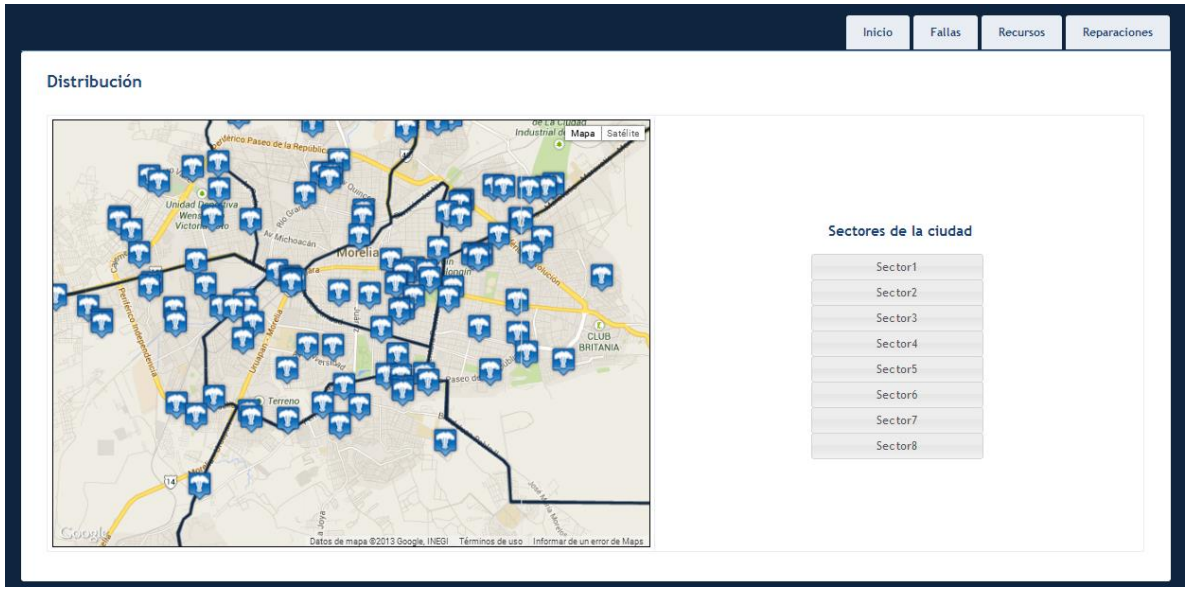


Figura 33. Pantalla inicial de la fase de Distribución del módulo de "Reparaciones".

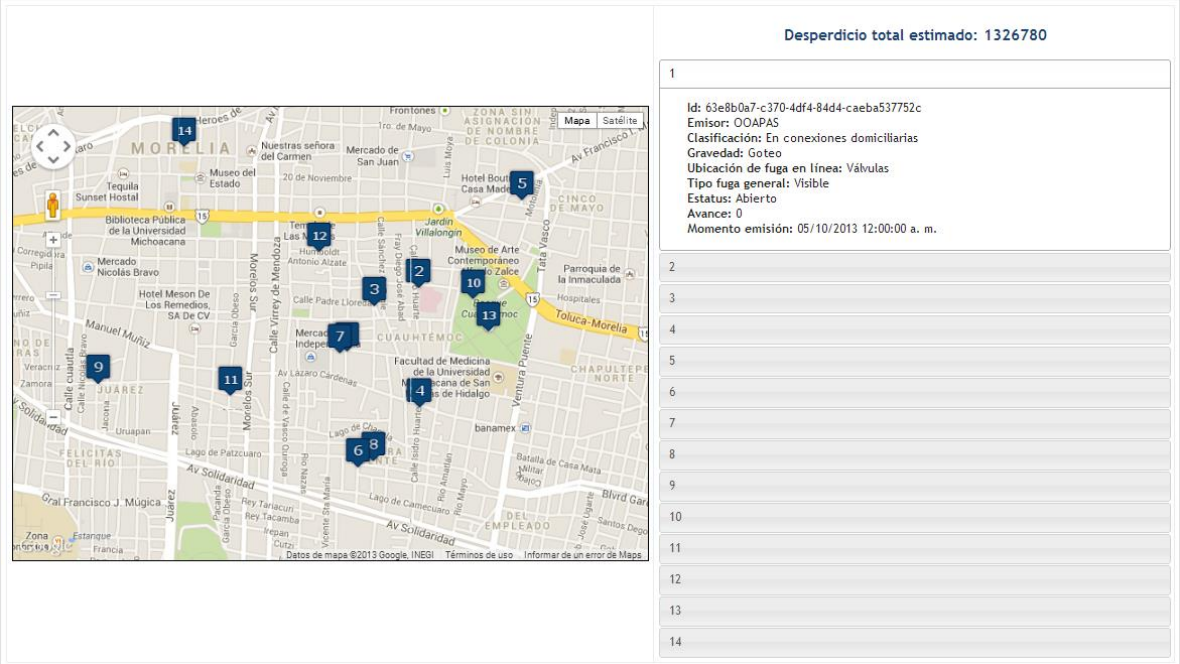


Figura 34. Pantalla secundaria de la fase de Distribución del módulo de "Reparaciones".

Después de seleccionar un sector para la para la tarea de Distribución mostrado en la figura 32, se mostrará la secuencia de atención de fugas a realizar y se indicará el desperdicio total estimado que generará es secuencia de actividades tal como se muestra en la figura 33.

En la sección de Bacheo, utilizando como base las distancias entre dos puntos dada a partir de la API (*Application Programming Interface*) de Google es que se propondrá una ruta para poder bachear cada una de las fugas una vez que fueron reparadas, la ruta mostrada también será proporcionada a partir de la API de Google (Véase las Figuras 35, 36 y 37).

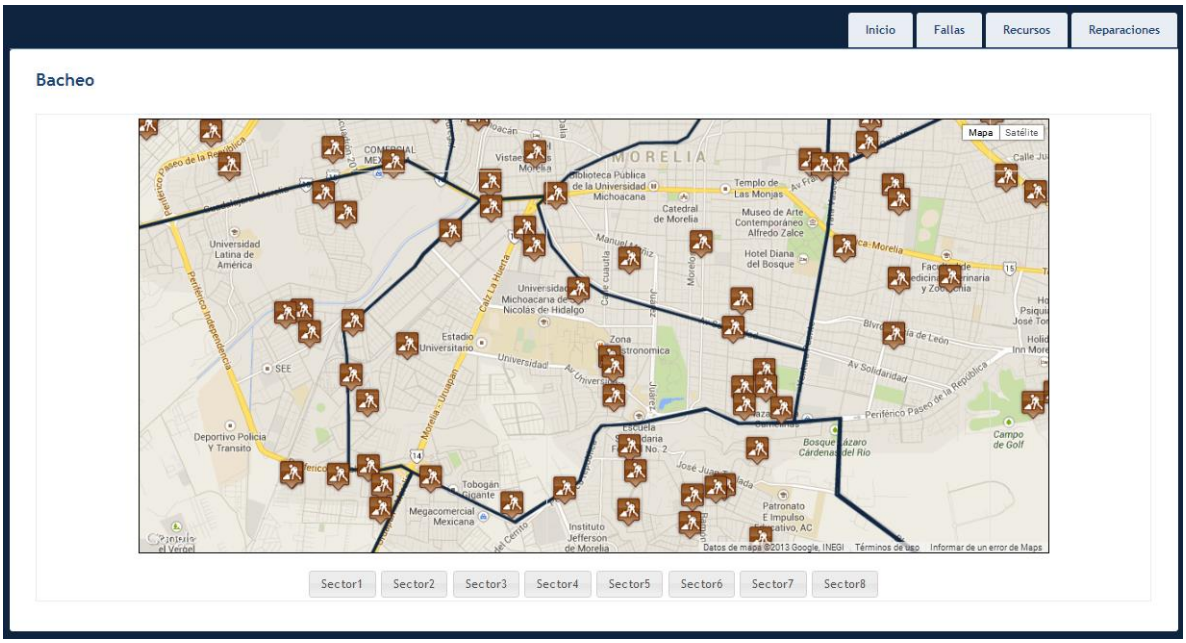


Figura 35. Pantalla inicial de la fase de Bacheo del módulo de "Reparaciones".

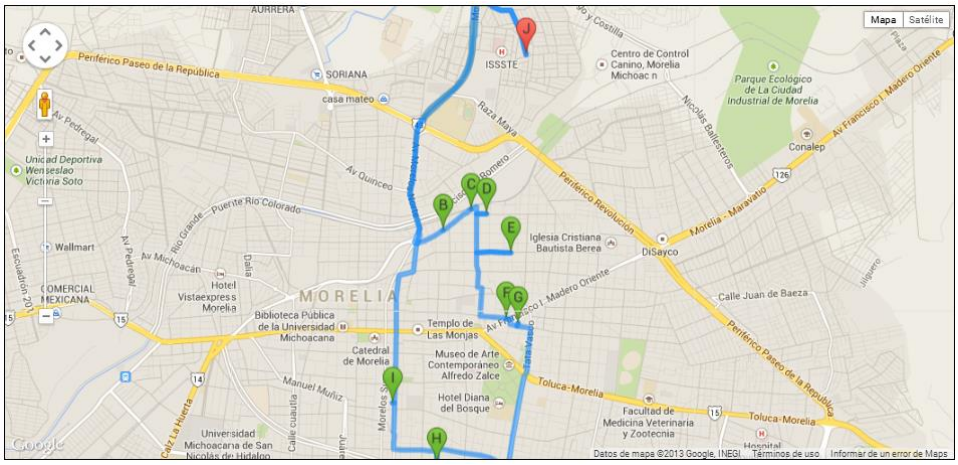


Figura 36. Pantalla secundaria de la fase de Bacheo del módulo de "Reparaciones".

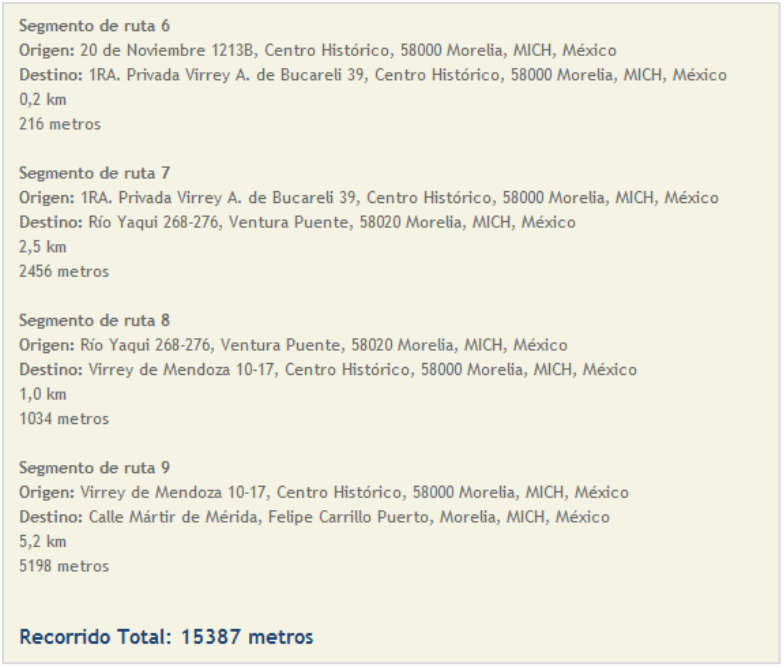


Figura 37. Lista secundaria de la fase de Bacheo del módulo de "Reparaciones".

Después de seleccionar un sector para la para la tarea de Bacheo mostrado en la figura 34, se entregará una ruta de atención de fugas ya reparadas para poder realizar su bacheo respectivo tal como se muestra en la figura 35; además, se mostrará una lista con los segmentos a recorrer por cada lugar que requiera ser bacheado, indicando las distancias de cada segmentos y la distancia total del recorrido así como se observa en la figura 36.

4.4 Resumen

En éste capítulo se expuso la naturaleza del problema, los procesos involucrados en el OOAPAS para generar los reportes de fugas y el protocolo para su reparación; a partir de ello, se planteó una propuesta de solución.

Con base en el análisis y diseño del Capítulo 3, se construyó el Sistema de ésta investigación, mismo que fue explicado también en el presente capítulo.

5 Pruebas y Resultados

Las pruebas de la aplicación del sistema, están orientadas hacia las características requeridas por el Organismo Operador de Agua Potable, Alcantarillado y Saneamiento de la ciudad de Morelia (OOAPAS).

Se toman como puntos de partida, los dos objetivos base del sistema que son: minimizar el desperdicio de agua generado por fugas en una ciudad y disminuir los gastos de traslado del personal para la atención de las fugas.

Por otra parte, como punto de comparación con soluciones óptimas, se realizaron evaluaciones al algoritmo ACO (*Ant Colony Optimization*) utilizando instancias de prueba y posteriormente, una vez comprobado que el algoritmo es útil para el tipo de análisis que se realiza en esta investigación, se ejecutaron las evaluaciones de las instancias con problemas de fugas reales obtenidas gracias al apoyo del OOAPAS de Morelia.

Las características del equipo de cómputo que se utilizó para las pruebas fue: procesador Intel Core i5 a 1.80 GHz, con 6 GB en RAM y el Sistema Operativo Windows 8 de 64 bits.

5.1 Pruebas al algoritmo ACO

La solución propuesta implica dos problemas, SMTWTP (*Single-Machine Total Weighted Tardiness Problem*) y TSP (*Traveling Salesman Problem*), por consiguiente el algoritmo del motor del DSS *(Decision Support System)* fue evaluado sobre instancias de prueba afines a los problemas mencionados.

5.1.1 Pruebas al problema SMTWTP

Para evaluar el desempeño del algoritmo *ACO* propuesto como solución al problema *SMTWTP* se contemplaron 3 pruebas con un tamaño de instancias de 7, 10 y 20 nodos (trabajos) respectivamente, de los cuales ya se tenían un análisis previo. Cada una de las pruebas fue realizada con dos variaciones del algoritmo: *Best* y *Random*. La variación *Best*, se basa en construir la solución a partir de los nodos cuya probabilidad de elegirlos es la más alta, mientras que en la *Random,* para construir la solución se hace a partir de una selección aleatoria que se apoya en las probabilidades de cada uno de los nodos a elegir. Por otra parte, se utilizaron 12 diferentes pares de ciclos y hormigas para su solución, tal como se muestra en la Tabla 2; donde para cada combinación, el algoritmo fue ejecutado 30 veces resultando un total de 720 ejecuciones por instancia de pruebas.

Tabla 2. Matriz de pruebas con la cantidad de ejecuciones que se realizaron para el problema *SMTWTP*.

		SMTWTP	
		Prueba	
Cycles	Ants	Best	Random
10	10	30	30
	20	30	30
40	10	30	30
	20	30	30
80	10	30	30
	20	30	30
120	10	30	30
	20	30	30
160	10	30	30
	20	30	30
200	10	30	30
	20	30	30

5.1.1.1 SMTWTP Prueba_7

En la Tabla 3, se registra una prueba obtenida de [**33**], tomando como muestra 7 trabajos, en donde la solución óptima es la secuencia con una tardanza total ponderada de 454 unidades.

Tabla 3. Instancia de Prueba_7 para el problema *SMTWTP* utilizando 7 trabajos [33].

j	1	2	3	4	5	6	7
p_j	12	13	14	16	26	31	32
d_j	42	33	51	48	63	88	146
w_j	7	9	5	14	10	11	8

En la Tabla 4 se muestran los mejores resultados obtenidos al haber ejecutado la instancia para la Prueba_7 con el algoritmo *ACO* utilizado, asimismo en la Tabla 5, se registra el error relativo de dichos resultados.

Tabla 4. Mejores resultados arrojados por el algoritmo *ACO* utilizado en la instancia Prueba_7.

		SMTWTP Prueba_7	
Cycles	Ants	Best	Random
10	10	536.00	535.00
	20	536.00	454.00
40	10	536.00	535.00
	20	536.00	454.00
80	10	536.00	464.00
	20	536.00	454.00
120	10	536.00	464.00
	20	536.00	454.00
160	10	536.00	454.00
	20	536.00	454.00
200	10	536.00	464.00
	20	536.00	454.00

Tabla 5. Error relativo de los mejores resultados en Prueba_7.

		SMTWTP Prueba_7 Error Relativo	
Cycles	Ants	Best	Random
10	10	18.06	17.84
	20	18.06	0.00
40	10	18.06	17.84
	20	18.06	0.00
80	10	18.06	2.20
	20	18.06	0.00
120	10	18.06	2.20
	20	18.06	0.00
160	10	18.06	0.00
	20	18.06	0.00
200	10	18.06	2.20
	20	18.06	0.00

5.1.1.2 SMTWTP Prueba_10

En la Tabla 6 se expone una prueba tomando como muestra 10 trabajos extraídos de [**33**], en donde la solución óptima es la secuencia con una tardanza total ponderada de 218 unidades.

Tabla 6. Instancia de Prueba_10 para el problema *SMTWTP* utilizando 10 trabajos (Liu, Abdelrahman, & Ramaswamy, 2005).

j	1	2	3	4	5	6	7	8	9	10
p_j	8	12	6	10	3	11	9	11	13	7
d_j	26	28	32	35	38	48	50	51	53	64
w_j	4	1	6	5	1	4	5	9	8	1

En la Tabla 7 se observan los mejores resultados obtenidos al haber ejecutado la instancia para la Prueba_10 con el algoritmo *ACO* utilizado, asimismo en la Tabla 8 se muestra el error relativo de dichos resultados.

Tabla 7. Mejores resultados arrojados por el algoritmo *ACO* utilizado en la instancia Prueba_10.

		SMTWTP Prueba_10	
Cycles	Ants	Best	Random
10	10	470.0	356.0
	20	470.0	381.0
40	10	413.0	341.0
	20	413.0	283.0
80	10	249.0	349.0
	20	271.0	260.0
120	10	260.0	283.0
	20	218.0	278.0
160	10	222.0	225.0
	20	218.0	222.0
200	10	234.0	283.0
	20	225.0	218.0

Tabla 8. Error relativo de los mejores resultados en Prueba_10.

		SMTWTP Prueba_10 Error Relativo	
Cycles	Ants	Best	Random
10	10	115.6	63.3
	20	115.6	74.8
40	10	89.4	56.4
	20	89.4	29.8
80	10	14.2	60.1
	20	24.3	19.3
120	10	19.3	29.8
	20	0.0	27.5
160	10	1.8	3.2
	20	0.0	1.8
200	10	7.3	29.8
	20	3.2	0.0

5.1.1.3 SMTWTP Prueba_20

En la Tabla 9 se observa una prueba con 20 trabajos tomada de [**37**] en donde la solución óptima es la secuencia con una tardanza total ponderada de 69 unidades.

Tabla 9. Instancia de Prueba_20 para el problema *SMTWTP* utilizando 20 trabajos

j	p_j	d_j	w_j
1	3	3	1
2	5	6	3
3	1	9	5
4	4	11	2
5	2	14	4
6	4	17	1
7	5	20	3
8	1	22	5
9	3	25	2
10	2	28	4
11	5	31	1

12	2	33	3
13	3	36	5
14	4	39	2
15	1	42	4
16	5	44	1
17	3	47	3
18	2	50	5
19	1	53	2
20	4	55	4

En la Tabla 10 se registran los mejores resultados obtenidos al haber ejecutado la instancia para la Prueba_20 con el algoritmo *ACO* utilizado, asimismo en la Tabla 11 se muestra el error relativo de dichos resultados.

Tabla 10. Mejores resultados arrojados por el algoritmo *ACO* utilizado en la instancia Prueba_20.

		SMTWTP	
		Prueba_20	
Cycles	**Ants**	Best	Random
10	10	95.00	80.00
	20	87.00	75.00
40	10	87.00	79.00
	20	87.00	80.00
80	10	87.00	78.00
	20	87.00	73.00
120	10	87.00	72.00
	20	87.00	69.00
160	10	87.00	77.00
	20	95.00	75.00
200	10	69.00	75.00
	20	72.00	69.00

Tabla 11. Error relativo de los mejores resultados en Prueba_20.

		SMTWTP Prueba_20 Error Relativo	
Cycles	Ants	Best	Random
10	10	37.68	15.94
	20	26.09	8.70
40	10	26.09	14.49
	20	26.09	15.94
80	10	26.09	13.04
	20	26.09	5.80
120	10	26.09	4.35
	20	26.09	0.00
160	10	26.09	11.59
	20	37.68	8.70
200	10	0.00	8.70
	20	4.35	0.00

5.1.1.4 Evaluación de los resultados en el problema SMTWTP

Como se pudo observar en las secciones de la 5.1.1.1, 5.1.1.2 y 5.1.1.3, en las pruebas del algoritmo respecto al problema *SMTWTP*, los resultados fueron favorables, en la mayoría de las instancias el error relativo no fue mayor al 10%, así que para la cantidad de nodos que se trataron se muestra buen desempeño y cabe destacar, que para la variación *Best* del algoritmo la convergencia fue más rápida que las otras variaciones, sin embargo, la mayoría de las ejecuciones mantuvo una tendencia sobre óptimos locales; por otro lado, la opción *Random*, aunque tarda en converger un poco más, acertó en mayor cantidad de ocasiones al óptimo resultado.

Utilizando la regla de Sturges [**38**] se determinó el número de clases a utilizar para evaluar la frecuencia de los resultados que se generaron, que en este caso, dado que fueron 30 las ejecuciones realizadas por cada combinación de variaciones, ciclos y hormigas, las clases resultantes fueron 6 para cada una de las instancias de prueba.

En las tablas 12, 13 y 14, se muestra la frecuencia con la que cada una de las ejecuciones fue registrada.

Tabla 12. Frecuencias de los resultados obtenidos por el algoritmo *ACO* para la instancia Prueba_7.

Prueba_7	454-710	711-966	967-1222	1223-1478	1479-1734	1735-
Best 10 10	21	6	1	1	0	1
Best 10 20	20	9	1	0	0	0
Best 40 10	17	13	0	0	0	0
Best 40 20	25	5	0	0	0	0
Best 80 10	20	10	0	0	0	0
Best 80 20	25	5	0	0	0	0
Best 120 10	19	11	0	0	0	0
Best 120 20	23	7	0	0	0	0
Best 160 10	19	11	0	0	0	0
Best 160 20	22	8	0	0	0	0
Best 200 10	21	9	0	0	0	0
Best 200 20	24	6	0	0	0	0
Random 10 10	24	4	0	1	1	0
Random 10 20	30	0	0	0	0	0
Random 40 10	30	0	0	0	0	0
Random 40 20	30	0	0	0	0	0
Random 80 10	30	0	0	0	0	0
Random 80 20	30	0	0	0	0	0
Random 120 10	30	0	0	0	0	0
Random 120 20	30	0	0	0	0	0
Random 160 10	30	0	0	0	0	0
Random 160 20	30	0	0	0	0	0
Random 200 10	30	0	0	0	0	0
Random 200 20	30	0	0	0	0	0

Tabla 13. Frecuencias de los resultados obtenidos por el algoritmo *ACO* para la instancia Prueba_10.

Prueba_10	218-294	295-370	371-446	447-522	523-598	599-
Best 10 10	0	0	0	21	7	2
Best 10 20	0	0	0	26	4	0
Best 40 10	0	0	1	28	1	0
Best 40 20	0	0	6	24	0	0
Best 80 10	3	0	3	24	0	0
Best 80 20	1	1	4	24	0	0

Best 120 10	2	0	4	24	0	0
Best 120 20	1	0	2	27	0	0
Best 160 10	2	0	4	24	0	0
Best 160 20	3	0	4	23	0	0
Best 200 10	5	1	2	22	0	0
Best 200 20	2	0	5	23	0	0
Random 10 10	0	1	12	16	1	0
Random 10 20	0	0	22	8	0	0
Random 40 10	0	11	18	1	0	0
Random 40 20	3	9	18	0	0	0
Random 80 10	0	13	17	0	0	0
Random 80 20	3	21	6	0	0	0
Random 120 10	2	18	10	0	0	0
Random 120 20	4	23	3	0	0	0
Random 160 10	2	19	9	0	0	0
Random 160 20	2	25	3	0	0	0
Random 200 10	4	22	4	0	0	0
Random 200 20	5	24	1	0	0	0

Tabla 14. Frecuencias de los resultados obtenidos por el algoritmo *ACO* para la instancia Prueba_20.

Prueba_20	69-95	96-121	122-147	148-173	174-199	200
Best 10 10	2	14	6	4	3	1
Best 10 20	3	21	3	1	2	0
Best 40 10	5	15	6	1	3	0
Best 40 20	6	17	4	2	1	0
Best 80 10	3	17	3	3	2	2
Best 80 20	5	21	2	2	0	0
Best 120 10	5	14	7	2	2	0
Best 120 20	4	19	4	2	0	1
Best 160 10	11	12	3	3	0	1
Best 160 20	1	22	1	2	3	1
Best 200 10	7	18	2	2	1	0
Best 200 20	4	22	1	2	1	0
Random 10 10	20	10	0	0	0	0
Random 10 20	14	16	0	0	0	0
Random 40 10	30	0	0	0	0	0
Random 40 20	23	7	0	0	0	0

Random 80 10	30	0	0	0	0	0
Random 80 20	23	7	0	0	0	0
Random 120 10	30	0	0	0	0	0
Random 120 20	26	4	0	0	0	0
Random 160 10	30	0	0	0	0	0
Random 160 20	28	2	0	0	0	0
Random 200 10	30	0	0	0	0	0
Random 200 20	27	3	0	0	0	0

Por otro lado, en las figuras 38, 39 y 40 se observan las gráficas de dispersión acerca del comportamiento del algoritmo en sus dos variantes: *Best* y *Random.*

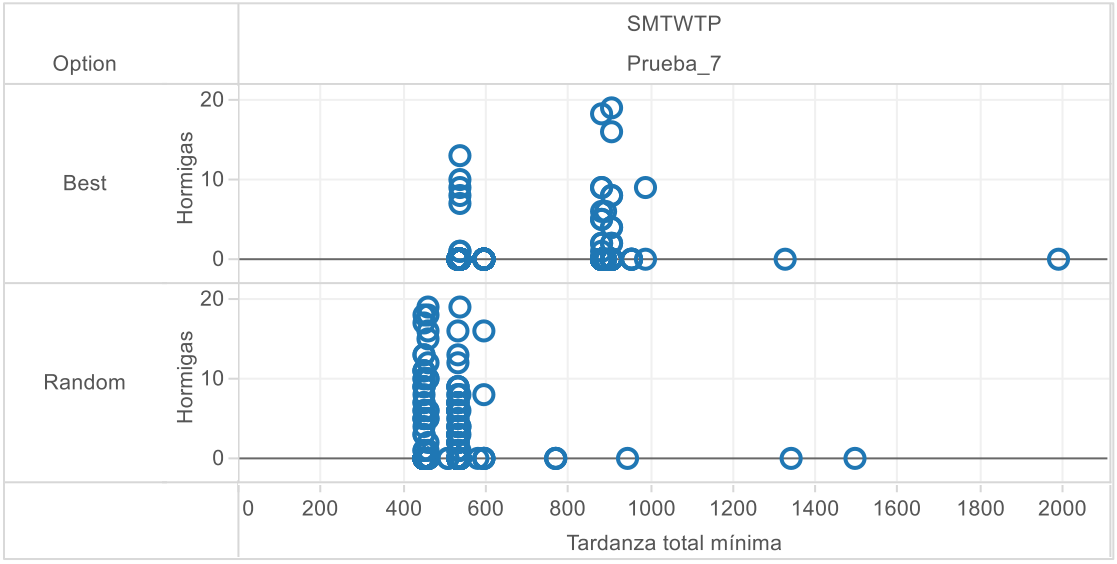


Figura 38. Comportamiento de las ejecuciones de la instancia Prueba_7 utilizando como referencia la tardanza total mínima.

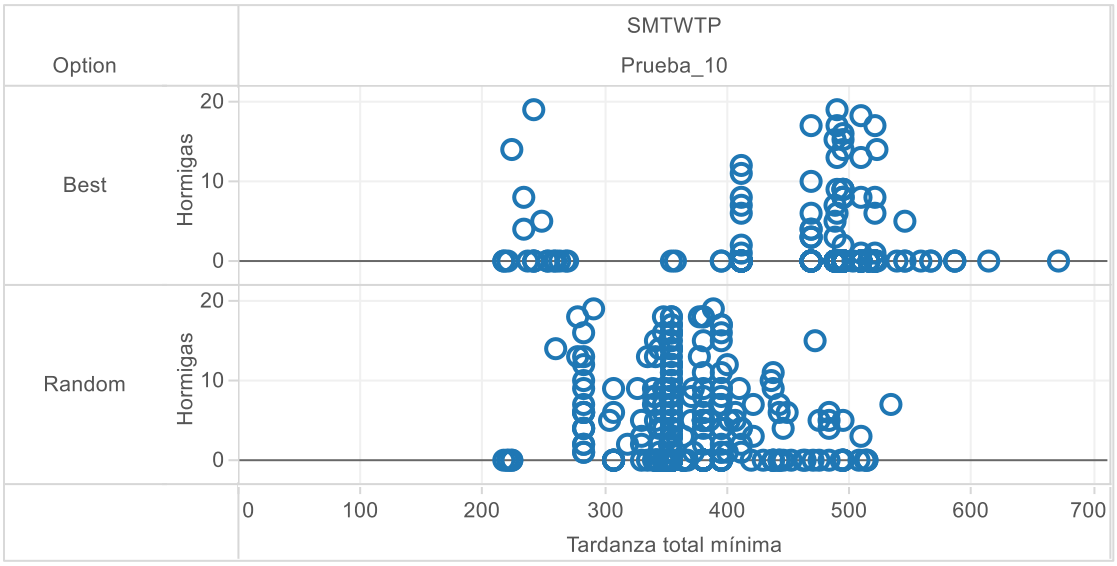


Figura 39. Comportamiento de las ejecuciones de la instancia Prueba_10 utilizando como referencia la tardanza total mínima.

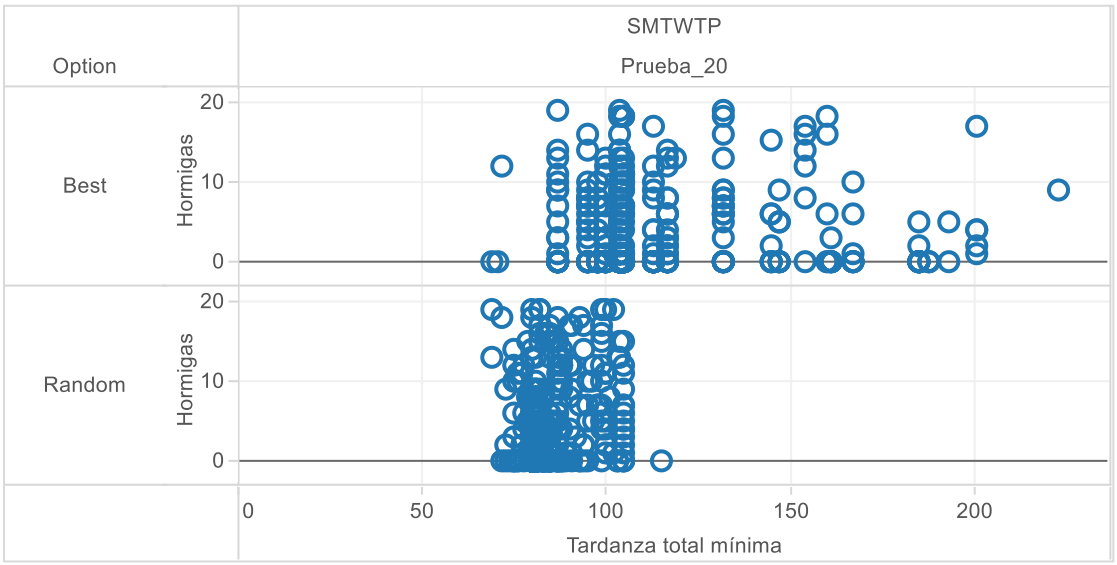


Figura 40. Comportamiento de las ejecuciones de la instancia Prueba_20 utilizando como referencia la tardanza total mínima.

5.1.2 Pruebas al problema TSP

Para evaluar el desempeño del algoritmo *ACO* propuesto como solución al problema TSP se contemplaron 3 pruebas con un tamaño de instancias de 16, 29 y 48 nodos (ciudades) respectivamente, de los cuales ya se tenía un análisis previo. Cada una de las pruebas fue realizada con dos variaciones del algoritmo: *Best* y *Random*. La variación *Best*, se basa en construir la solución a partir de los nodos cuya probabilidad de elegirlos es la más alta, mientras que en la variación *Random* para construir la solución lo hace a partir de una selección aleatoria con base en las probabilidades de cada uno de los nodos a elegir. Además se utilizaron 12 diferentes pares de ciclos y hormigas para su solución, tal como se muestra en la Tabla 15. Para cada combinación, el algoritmo fue ejecutado 5 veces resultando un total de 120 ejecuciones por instancia de pruebas.

Tabla 15. Matriz de pruebas con la cantidad de ejecuciones que se realizaron para el problema *TSP*.

		TSP	
		Prueba	
Cycles	Ants	Best	Random
10	10	5	5
	20	5	5
40	10	5	5
	20	5	5
80	10	5	5
	20	5	5
120	10	5	5
	20	5	5
160	10	5	5
	20	5	5
200	10	5	5
	20	5	5

5.1.2.1 TSP ulysses16

En la primer prueba se evaluó la instancia *ulysses16* con 16 nodos tomada de [**39**] en donde la solución óptima es la secuencia de ciudades con una distancia de 6859 unidades.

En la Tabla 16 se muestran los mejores resultados obtenidos al haber ejecutado la instancia de prueba *ulysses16* en el algoritmo *ACO* utilizado, por su lado en la Tabla 17 se observa el error relativo de dichos resultados.

Tabla 16. Mejores resultados arrojados por el algoritmo *ACO* utilizado en la instancia *ulysses16*.

		TSP ulysses16	
Cycles	Ants	Best	Random
10	10	6,870.0	7,108.0
	20	6,915.0	7,108.0
40	10	6,870.0	6,986.0
	20	6,909.0	7,025.0
80	10	6,870.0	6,932.0
	20	6,859.0	6,986.0
120	10	6,870.0	7,000.0
	20	6,915.0	6,986.0
160	10	6,915.0	7,025.0
	20	6,915.0	7,025.0
200	10	6,870.0	7,000.0
	20	6,915.0	6,986.0

Tabla 17. Error relativo de los mejores resultados en *ulysses16*.

		TSP ulysses16 Error Relativo	
Cycles	Ants	Best	Random
10	10	0.160	3.630
	20	0.816	3.630
40	10	0.160	1.852
	20	0.729	2.420
80	10	0.160	1.064
	20	0.000	1.852
120	10	0.160	2.056
	20	0.816	1.852
160	10	0.816	2.420
	20	0.816	2.420
200	10	0.160	2.056
	20	0.816	1.852

5.1.2.2 TSP bays29

En la segunda prueba se evaluó la instancia *bays29* con 29 nodos tomada de [**39**] en donde la solución óptima es la secuencia de ciudades con una distancia de 2020 unidades.

En la Tabla 18 se pueden observar los mejores resultados obtenidos al haber ejecutado la instancia de prueba *bays29* en el algoritmo *ACO* utilizado, por su parte en la Tabla 19 se puede ver el error relativo de los mismos resultados.

Tabla 18. Mejores resultados arrojados por el algoritmo *ACO* utilizado en la instancia *bays29*.

		TSP bays29	
Cycles	**Ants**	Best	Random
10	10	2,114.0	3,068.0
	20	2,134.0	2,742.0
40	10	2,134.0	2,724.0
	20	2,134.0	2,608.0
80	10	2,134.0	2,513.0
	20	2,134.0	2,493.0
120	10	2,134.0	2,705.0
	20	2,134.0	2,416.0
160	10	2,134.0	2,577.0
	20	2,134.0	2,440.0
200	10	2,134.0	2,616.0
	20	2,134.0	2,400.0

Tabla 19. Error relativo de los mejores resultados en *bays29*.

Cycles	Ants	TSP bays29 Error Relativo Best	TSP bays29 Error Relativo Random
10	10	4.65	51.88
	20	5.64	35.74
40	10	5.64	34.85
	20	5.64	29.11
80	10	5.64	24.41
	20	5.64	23.42
120	10	5.64	33.91
	20	5.64	19.60
160	10	5.64	27.57
	20	5.64	20.79
200	10	5.64	29.50
	20	5.64	18.81

5.1.2.3 TSP att48

En la tercera prueba se evaluó la instancia *att48* con 48 nodos tomada de [**39**] en donde la solución óptima es la secuencia de ciudades con una distancia de 10628 unidades.

En la Tabla 20 se muestran los mejores resultados obtenidos al haber ejecutado la instancia de prueba *att48* en el algoritmo *ACO* utilizado, por su lado en la Tabla 21 se registra el error relativo de dichos resultados.

Tabla 20. Mejores resultados arrojados por el algoritmo *ACO* utilizado en la instancia *att48*.

		TSP att48	
Cycles	Ants	Best	Random
10	10	12,020	19,796
	20	12,001	18,820
40	10	11,839	17,383
	20	11,839	17,202
80	10	12,012	20,432
	20	11,839	22,180
120	10	11,839	20,432
	20	11,847	18,820
160	10	11,753	20,189
	20	11,753	18,820
200	10	11,753	18,820
	20	11,839	18,820

Tabla 21. Error relativo de los mejores resultados en *att48*.

		TSP att48 Error Relativo	
Cycles	Ants	Best	Random
10	10	13.10	86.26
	20	12.92	77.08
40	10	11.39	63.56
	20	11.39	61.86
80	10	13.02	92.25
	20	11.39	108.69
120	10	11.39	92.25
	20	11.47	77.08
160	10	10.59	89.96
	20	10.59	77.08
200	10	10.59	77.08
	20	11.39	77.08

5.1.2.4 Evaluación de los resultados en el problema TSP

En las pruebas que se realizaron al problema *TSP* se obtuvieron buenos resultados, considerados mejores que los obtenidos en el problema *SMTWTP* y aunque no se obtuvo el óptimo resultado, la variación de error fue mínima, además el periodo de convergencia fue mucho más rápido, de hecho a partir de 40 ciclos la diferencia entre los valores obtenidos fue despreciable, por consiguiente, las tablas y gráficas sólo se presentarán en función de las variaciones *Best* y *Random,* ya que fue en donde se mostraron diferencias considerables en los resultados.

Para determinar el número de clases de la frecuencia de los resultados que se generaron, se emplea nuevamente la regla de Sturges [**38**], en éste caso fueron 60 ejecuciones en cada una de las variaciones (*Best* y *Random*), dando como resultado 7 clases para cada una de ellas.

En las tablas 22, 23 y 24 se muestra la frecuencia con la que cada una de las ejecuciones fue registrada.

Tabla 22. Frecuencias de los resultados obtenidos por el algoritmo *ACO* para la instancia *ulysses16*.

ulysses16	6859-7017	1018-7175	7176-7333	7334-7491	7492-7649	7650-7807	7808-
ulysses16 Best	27	1	7	2	6	14	3
ulysses16 Random	8	36	12	2	0	0	2

Tabla 23. Frecuencias de los resultados obtenidos por el algoritmo *ACO* para la instancia *bays29*.

bays29	2020-2218	2219-2416	2417-2614	2615-2812	2813-3010	3011-3208	3209-
bays29 Best	55	5	0	0	0	0	0
bays29 Random	0	2	13	26	11	5	3

Tabla 24. Frecuencias de los resultados obtenidos por el algoritmo *ACO* para la instancia *att48*.

att48	10628-12808	12809-14988	14989-17168	17169-19348	19349-21528	21529-23708	23709-
att48 Best	57	3	0	0	0	0	0
att48 Random	0	0	0	15	18	17	10

Por otro lado, en las figuras 41, 42 y 43 se muestran las gráficas de dispersión acerca del comportamiento del algoritmo en sus dos variantes: *Best* y *Random.*

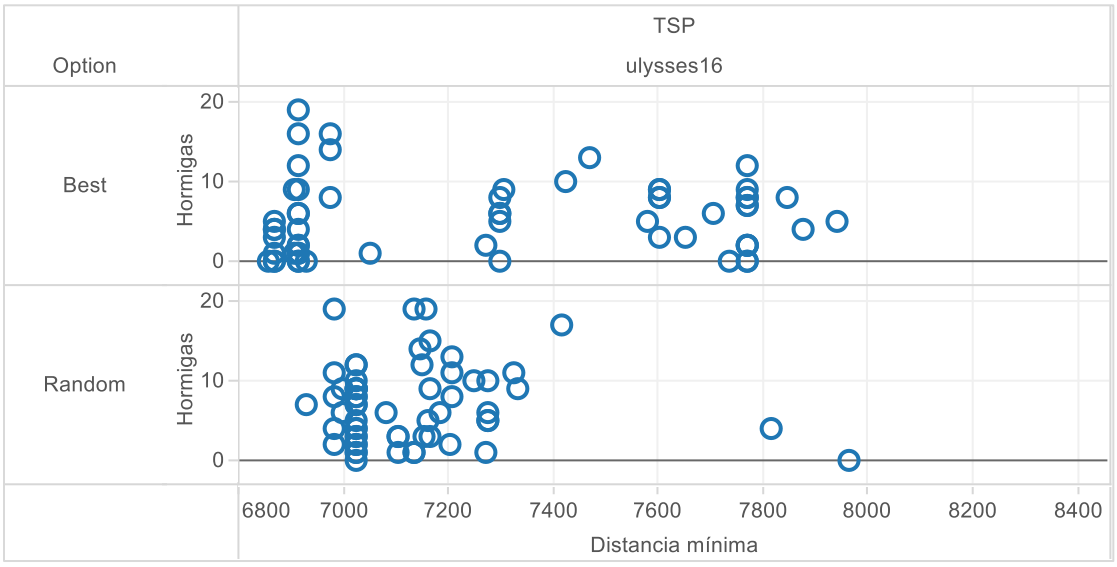


Figura 41. Comportamiento de las ejecuciones de la instancia ulysses16 utilizando como referencia la distancia mínima.

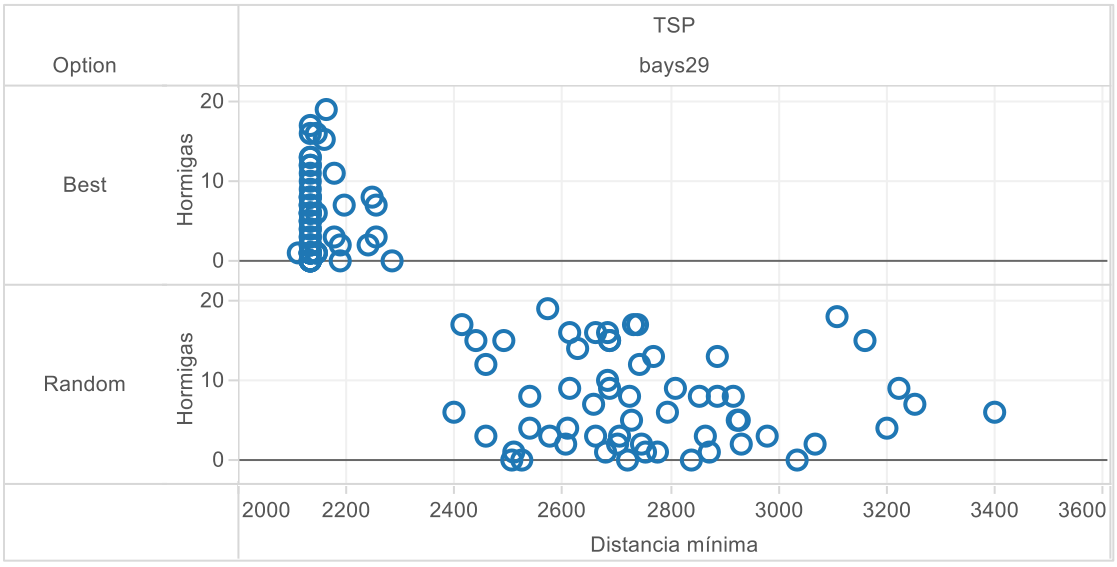


Figura 42. Comportamiento de las ejecuciones de la instancia bays29 utilizando como referencia la distancia mínima.

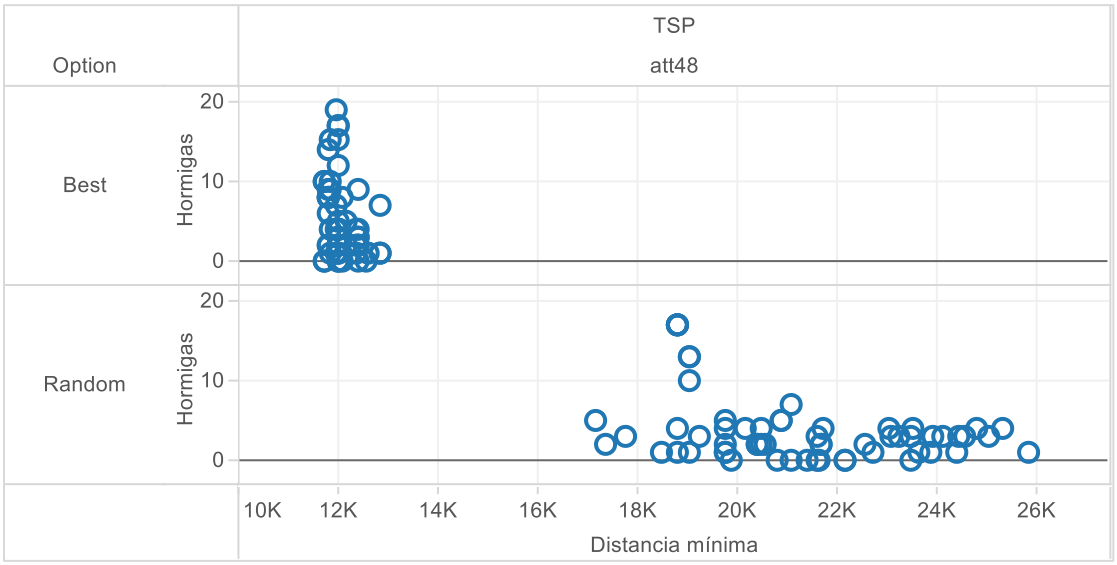


Figura 43. Comportamiento de las ejecuciones de la instancia *att48* utilizando como referencia la distancia mínima.

5.2 Pruebas al Sistema dentro del Entorno Real

Una vez demostrado que el algoritmo proporciona buenos resultados para el tipo de análisis que se requiere para el OOAPAS, se prosigue a probar el sistema frente al entorno real de la ciudad de Morelia, con base en los resultados obtenidos de las pruebas, se hicieron las ejecuciones del algoritmo utilizando la matriz de valores que se muestra en la Tabla 25.

Tabla 25. Matriz de pruebas con la cantidad de ejecuciones que se realizaron con los datos reales de la ciudad de Morelia.

		Problema SMTWTP / TSP	
		Instancia de fugas de la ciudad	
Cycles	**Ants**	Best	Random
10	20	5	5
40	20	5	5
80	20	5	5
120	20	5	5

El sistema se ejecutó con datos reales proporcionados por el OOAPAS y que corresponden a una semana común de trabajo en la ciudad de Morelia, tomando en cuenta la asignación

de actividades que a diario se hace para solución de fugas hidráulicas en la ciudad y con el fin de probar el sistema con la mayor cantidad de carga, se consideró la información de todas las fugas por atender de una semana completa.

En las tablas 26 y 27 se muestran los mejores resultados de las ejecuciones del algoritmo *ACO* para la fase de Distribución, en la cual, se aplicó la solución del problema *SMTWTP,* considerando la división en sectores de la ciudad como la maneja el OOAPAS, a partir de los cuales se generaron cada una de las soluciones. El valor de la columna “OOAPAS” registra el desperdicio de agua generado a partir de la solución propuesta por el Organismo de la ciudad, el valor de la columna “ACO” muestra el desperdicio de agua que se hubiera generado en caso de haber aplicado el análisis combinatorio propuesto en esta investigación, finalmente, las columnas “Ahorro” y “% Ahorro” señalan el ahorro en litros de agua y su porcentaje respectivo aplicando esta solución propuesta, notándose una diferencia considerable.

Tabla 26. Mejores resultados arrojados por el algoritmo *ACO* (variación *Best*) para la fase de "Distribución" respecto a los resultados del OOAPAS.

				Best			
Instance	Cycles	Ants	Nodes	ACO	OOAPAS	Ahorro	% Ahorro
Sector1_SMTWTP	10	20	14	1,326,780	1,326,780	0	0
	40	20	14	1,326,780	1,326,780	0	0
	80	20	14	1,326,780	1,326,780	0	0
	120	20	14	1,326,780	1,326,780	0	0
Sector2_SMTWTP	10	20	16	2,353,860	3,992,490	1,638,630	41
	40	20	16	2,353,860	3,992,490	1,638,630	41
	80	20	16	1,807,650	3,992,490	2,184,840	55
	120	20	16	1,807,650	3,992,490	2,184,840	55
Sector3_SMTWTP	10	20	13	2,797,470	3,343,680	546,210	16
	40	20	13	2,251,260	3,343,680	1,092,420	33
	80	20	13	1,705,050	3,343,680	1,638,630	49
	120	20	13	2,797,470	3,343,680	546,210	16
Sector4_SMTWTP	10	20	19	11,821,950	21,107,520	9,285,570	44
	40	20	19	10,729,530	21,107,520	10,377,990	49
	80	20	19	7,998,480	21,107,520	13,109,040	62
	120	20	19	4,721,220	21,107,520	16,386,300	78
Sector5_SMTWTP	10	20	15	2,601,180	18,441,270	15,840,090	86
	40	20	15	2,601,180	18,441,270	15,840,090	86
	80	20	15	2,601,180	18,441,270	15,840,090	86
	120	20	15	2,601,180	18,441,270	15,840,090	86
Sector6_SMTWTP	10	20	14	1,872,720	7,334,820	5,462,100	74
	40	20	14	1,872,720	7,334,820	5,462,100	74
	80	20	14	1,872,720	7,334,820	5,462,100	74
	120	20	14	1,872,720	7,334,820	5,462,100	74
Sector7_SMTWTP	10	20	22	3,884,760	5,523,390	1,638,630	30
	40	20	22	3,884,760	5,523,390	1,638,630	30
	80	20	22	3,338,550	5,523,390	2,184,840	40
	120	20	22	3,338,550	5,523,390	2,184,840	40
Sector8_SMTWTP	10	20	16	1,640,250	6,556,140	4,915,890	75
	40	20	16	1,640,250	6,556,140	4,915,890	75
	80	20	16	1,640,250	6,556,140	4,915,890	75
	120	20	16	1,640,250	6,556,140	4,915,890	75

Tabla 27. Mejores resultados arrojados por el algoritmo *ACO* (variación *Random*) para la fase de "Distribución" respecto a los resultados del OOAPAS.

				Random			
Instance	Cycles	Ants	Nodes	ACO	OOAPAS	Ahorro	% Ahorro
Sector1_SMTWTP	10	20	14	1,326,780	1,326,780	0	0
	40	20	14	1,326,780	1,326,780	0	0
	80	20	14	1,326,780	1,326,780	0	0
	120	20	14	1,326,780	1,326,780	0	0
Sector2_SMTWTP	10	20	16	1,807,650	3,992,490	2,184,840	55
	40	20	16	2,353,860	3,992,490	1,638,630	41
	80	20	16	1,807,650	3,992,490	2,184,840	55
	120	20	16	1,807,650	3,992,490	2,184,840	55
Sector3_SMTWTP	10	20	13	3,343,680	3,343,680	0	0
	40	20	13	3,343,680	3,343,680	0	0
	80	20	13	3,343,680	3,343,680	0	0
	120	20	13	1,705,050	3,343,680	1,638,630	49
Sector4_SMTWTP	10	20	19	5,813,640	21,107,520	15,293,880	72
	40	20	19	6,359,850	21,107,520	14,747,670	70
	80	20	19	5,267,430	21,107,520	15,840,090	75
	120	20	19	4,721,220	21,107,520	16,386,300	78
Sector5_SMTWTP	10	20	15	2,054,970	18,441,270	16,386,300	89
	40	20	15	2,054,970	18,441,270	16,386,300	89
	80	20	15	2,054,970	18,441,270	16,386,300	89
	120	20	15	2,054,970	18,441,270	16,386,300	89
Sector6_SMTWTP	10	20	14	1,872,720	7,334,820	5,462,100	74
	40	20	14	1,872,720	7,334,820	5,462,100	74
	80	20	14	1,872,720	7,334,820	5,462,100	74
	120	20	14	1,872,720	7,334,820	5,462,100	74
Sector7_SMTWTP	10	20	22	3,338,550	5,523,390	2,184,840	40
	40	20	22	3,338,550	5,523,390	2,184,840	40
	80	20	22	3,338,550	5,523,390	2,184,840	40
	120	20	22	3,338,550	5,523,390	2,184,840	40
Sector8_SMTWTP	10	20	16	1,640,250	6,556,140	4,915,890	75
	40	20	16	1,640,250	6,556,140	4,915,890	75
	80	20	16	1,640,250	6,556,140	4,915,890	75
	120	20	16	1,640,250	6,556,140	4,915,890	75

Remarcando la otra etapa importante para el OOAPAS, en las tablas 28 y 29, se muestran los mejores resultados de las ejecuciones del algoritmo *ACO* para la fase de Bacheo, en la que se aplicó la solución del problema *TSP,* considerando nuevamente la distribución en sectores de la ciudad como la maneja el organismo. El valor de la columna “OOAPAS” muestra el recorrido en metros generado a partir de la solución propuesta por el organismo, el valor de la columna “ACO” muestra el recorrido que se hubiera generado en caso de haber aplicado la secuencia de actividades propuesta por el análisis combinatorio, finalmente, las columnas “Ahorro” y “% Ahorro” señalan el ahorro en metros y su porcentaje respectivo empleando la solución propuesta en ésta investigación.

Tabla 28. Mejores resultados arrojados por el algoritmo *ACO* (variación *Best*) para la fase de "Bacheo" respecto a los resultados del OOAPAS.

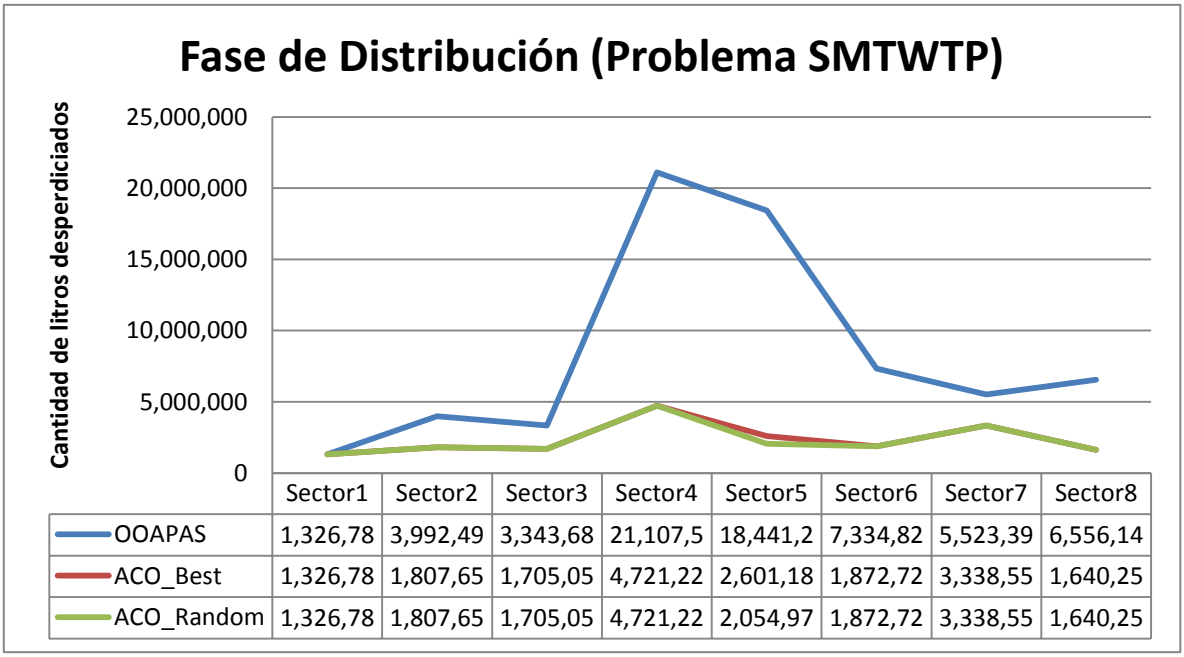
				Best			
Instance	Cycles	Ants	Nodes	ACO	OOAPAS	Ahorro	% Ahorro
Sector1_TSP	10	20	11	11,442	24,187	12,745	53
	40	20	11	11,910	24,187	12,277	51
	80	20	11	11,478	24,187	12,708	53
	120	20	11	11,910	24,187	12,277	51
Sector2_TSP	10	20	19	20,535	44,334	23,798	54
	40	20	19	20,529	44,334	23,805	54
	80	20	19	20,535	44,334	23,798	54
	120	20	19	20,051	44,334	24,283	55
Sector3_TSP	10	20	17	11,872	29,227	17,355	59
	40	20	17	11,966	29,227	17,261	59
	80	20	17	11,872	29,227	17,355	59
	120	20	17	11,872	29,227	17,355	59
Sector4_TSP	10	20	23	19,589	45,188	25,599	57
	40	20	23	19,589	45,188	25,599	57
	80	20	23	19,589	45,188	25,599	57
	120	20	23	19,589	45,188	25,599	57
Sector5_TSP	10	20	16	20,397	33,304	12,907	39
	40	20	16	20,397	33,304	12,907	39
	80	20	16	20,397	33,304	12,907	39
	120	20	16	20,397	33,304	12,907	39
Sector6_TSP	10	20	18	27,869	56,616	28,747	51
	40	20	18	27,869	56,616	28,747	51
	80	20	18	28,092	56,616	28,525	50
	120	20	18	27,869	56,616	28,747	51
Sector7_TSP	10	20	19	25,601	65,617	40,016	61
	40	20	19	25,138	65,617	40,479	62
	80	20	19	25,138	65,617	40,479	62
	120	20	19	25,138	65,617	40,479	62
Sector8_TSP	10	20	25	16,288	50,627	34,339	68
	40	20	25	16,288	50,627	34,339	68
	80	20	25	16,288	50,627	34,339	68
	120	20	25	16,288	50,627	34,339	68

Tabla 29. Mejores resultados arrojados por el algoritmo *ACO* (variación *Random*) para la fase de "Bacheo" respecto a los resultados del OOAPAS.

				Random			
Instance	Cycles	Ants	Nodes	ACO	OOAPAS	Ahorro	% Ahorro
Sector1_TSP	10	20	11	11,442	24,187	12,745	53
	40	20	11	11,387	24,187	12,800	53
	80	20	11	11,442	24,187	12,745	53
	120	20	11	11,387	24,187	12,800	53
Sector2_TSP	10	20	19	22,891	44,334	21,443	48
	40	20	19	22,378	44,334	21,955	50
	80	20	19	22,454	44,334	21,879	49
	120	20	19	21,721	44,334	22,613	51
Sector3_TSP	10	20	17	11,754	29,227	17,473	60
	40	20	17	11,907	29,227	17,320	59
	80	20	17	12,368	29,227	16,859	58
	120	20	17	11,907	29,227	17,320	59
Sector4_TSP	10	20	23	24,924	45,188	20,263	45
	40	20	23	24,316	45,188	20,872	46
	80	20	23	20,749	45,188	24,439	54
	120	20	23	23,237	45,188	21,951	49
Sector5_TSP	10	20	16	21,166	33,304	12,138	36
	40	20	16	20,656	33,304	12,647	38
	80	20	16	20,656	33,304	12,647	38
	120	20	16	20,656	33,304	12,647	38
Sector6_TSP	10	20	18	29,268	56,616	27,348	48
	40	20	18	28,493	56,616	28,123	50
	80	20	18	28,089	56,616	28,528	50
	120	20	18	28,110	56,616	28,506	50
Sector7_TSP	10	20	19	26,681	65,617	38,936	59
	40	20	19	27,303	65,617	38,314	58
	80	20	19	26,518	65,617	39,099	60
	120	20	19	26,533	65,617	39,084	60
Sector8_TSP	10	20	25	21,251	50,627	29,377	58
	40	20	25	19,701	50,627	30,926	61
	80	20	25	19,628	50,627	31,000	61
	120	20	25	19,908	50,627	30,720	61

En las cuatro tablas mostradas con los resultados reales para el problema de Distribución (*SMTWTP*) y el problema de Bacheo (*TSP*) se observa que el orden del ahorro generado para el desperdicio de agua y el ahorro para obtener un recorrido con menor distancia se encuentra arriba del 50 % para ambos problemas, en cuanto a las variaciones del algoritmo *Best* y *Random* se mantuvo el mismo comportamiento en los resultados sin mostrar mayor diferencia, finalmente, en cuanto al número de ciclos utilizados se pudo observar que desde el número de ciclos más bajo se obtuvieron buenos resultados y que a partir de 40 ciclos, fueron semejantes en casi todos los casos.

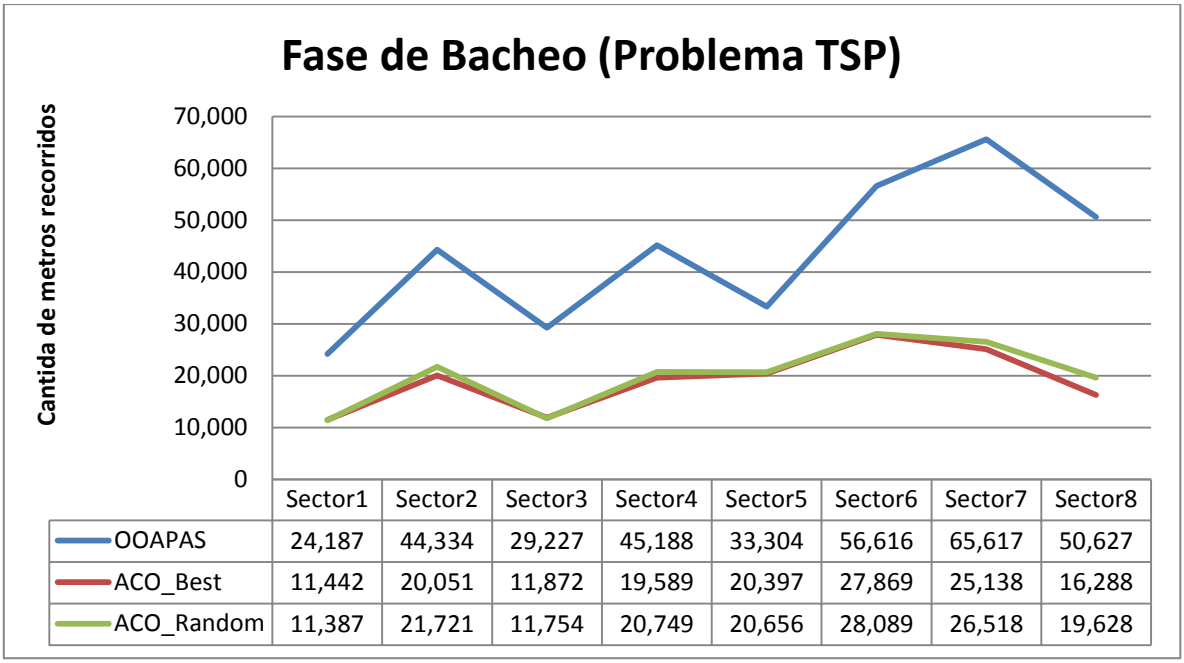
En la Figura 44 se observan los mejores resultados de las ejecuciones del algoritmo *ACO* en sus dos variantes (*Best* y *Random*) frente a los 8 sectores en que se encuentra dividida la ciudad, en las que se aborda la fase de Distribución dentro del Sistema.



	Sector1	Sector2	Sector3	Sector4	Sector5	Sector6	Sector7	Sector8
OOAPAS	1,326,78	3,992,49	3,343,68	21,107,5	18,441,2	7,334,82	5,523,39	6,556,14
ACO_Best	1,326,78	1,807,65	1,705,05	4,721,22	2,601,18	1,872,72	3,338,55	1,640,25
ACO_Random	1,326,78	1,807,65	1,705,05	4,721,22	2,054,97	1,872,72	3,338,55	1,640,25

Figura 44. Comparación de los mejores resultados obtenidos en las ejecuciones de las dos variantes del algoritmo *ACO* para la fase de Distribución (Problema *SMTWTP*).

En la Figura 45 se muestran los mejores resultados de las ejecuciones del algoritmo *ACO* en sus dos variantes (*Best* y *Random*) frente a los 8 sectores en que se encuentra dividida la ciudad, abordando la fase de Bacheo dentro del Sistema.



	Sector1	Sector2	Sector3	Sector4	Sector5	Sector6	Sector7	Sector8
OOAPAS	24,187	44,334	29,227	45,188	33,304	56,616	65,617	50,627
ACO_Best	11,442	20,051	11,872	19,589	20,397	27,869	25,138	16,288
ACO_Random	11,387	21,721	11,754	20,749	20,656	28,089	26,518	19,628

Figura 45. Comparación de los mejores resultados obtenidos en las ejecuciones de las dos variantes del algoritmo *ACO* para la fase de Bacheo (Problema *TSP*).

En ambas gráficas (Figuras 44 y 45) puede verse cómo las dos variantes para la ejecución del algoritmo *ACO* (*Best* y *Random*) tuvieron prácticamente el mismo comportamiento en sus mejores resultados y además, se muestra una ganancia sobresaliente respecto a los procesos actuales del organismo.

5.3 Resumen

Con base en instancias ficticias e instancias reales se realizaron las pruebas al Sistema planteado en esta investigación. Se hizo una evaluación de los resultados que la aplicación entrega frente a problemas ya conocidos y también se realizó la ejecución con los datos de la situación real de la ciudad de Morelia, Michoacán, misma que generó una comparativa entre los resultados reales del OOAPAS y los resultados arrojados por el Sistema propuesto.

6 Conclusiones y Trabajos futuros

6.1 Conclusiones

Los problemas planteados en este trabajo son fundamentales para cualquier empresa involucrada con el trazado de rutas y traslado de elementos o que requiera de una programación de actividades con el objetivo de hacer eficientes sus labores, por lo que la solución propuesta resulta ser una herramienta viable para cualquier organización que se encuentre inmersa en este contexto.

Se logró obtener la información necesaria para atender el mantenimiento correctivo que realiza el OOAPAS (Organismo Operador de Agua Potable Alcantarillado y Saneamiento de Morelia, Michoacán) y así, organizar los datos que se integraron en el sistema.

También, se realizó un estudio comparativo de los algoritmos y técnicas de programación, orientado a problemas de análisis combinatorio, de manera particular, en instancias del Problema del Agente Viajero (*TSP*) y del Problema de la Tardanza Total Ponderada en una Sola Máquina (*SMTWTP*), a partir de los resultados obtenidos, se seleccionó la opción más adecuada que fue: la Optimización basada en Colonias de Hormigas (*ACO*).

ACO se aplica bien en problemas de optimización combinatoria, en donde el espacio de soluciones es exponencial, gracias a su adecuación en este trabajo es que se lograron evitar grandes desperdicios de agua, a través de una programación de actividades proporcionada por esta técnica y ayudar a disminuir los gastos de traslado de los empleados a las reparaciones necesarias mediante una ruta de ejecución, demostrando con esta investigación, que la metaheurística *ACO* es altamente competitiva para este tipo de cuestiones.

Se consiguió aportar un procedimiento alternativo de solución para los problemas de análisis combinatorio, tales como programación de tareas y enrutamiento de vehículos, sustentado en la metaheurística *ACO*, técnica en la que a partir del comportamiento colaborativo de las hormigas, busca entregar soluciones óptimas mediante procesos iterativos, capaz de encajar como motor en los Sistemas de Toma de Decisiones.

En las evaluaciones de la solución propuesta que se realizaron a instancias ficticias y a instancias reales siempre se obtuvieron buenos resultados, variando únicamente la cantidad de hormigas y ciclos a utilizar; en las instancias ficticias, se observó que entre mayor era el número de ciclos los resultados mejoraban bastante, sin embargo, al hacer las pruebas en el entorno real no fue necesario el uso de un gran número de iteraciones debido a que la discrepancia entre los resultados no fue notoria.

Se mostró que para las respuestas dadas por el algoritmo para instancias ficticias cumple acertando a los resultados óptimos o en su defecto con un porcentaje de error mínimo, mientras que con datos reales, el análisis generó ahorros en promedio de un 50 % con su utilización.

6.2 Trabajos futuros

La solución propuesta cumplió satisfactoriamente con las metas planteadas mejorando los procesos de atención a las fugas hidráulicas, minimizando los desperdicios de agua y los tiempos de reparaciones, aun así, algunos de los puntos que podrían abordarse como trabajos futuros son:

- Profundizar en la manera que aplicaciones similares usadas para trazado de rutas o administración de actividades y recursos realizan el análisis combinatorio, con el fin de plantear mejoras en los procesos de atención a fugas hidráulicas.
- Modificar el algoritmo utilizado agregando un tercer parámetro el cual pudiera contemplar un elemento tal como las tuberías de la ciudad, distritos hidrométricos o fuentes de abastecimiento.
- Extender el sistema hacia el desarrollo de una aplicación móvil dirigida a la población para que le sea posible capturar fugas sin la necesidad de contactar al centro de atención a clientes.
- Permitir guardar y descargar en formato .pdf cada uno de los reportes para poder conservar el historial para el archivo dentro del organismo.

6.3 Divulgación de la Investigación

Con el propósito de participar en el fomento de la investigación y de promover el trabajo realizado, se elaboraron los siguientes artículos de divulgación:

- **Optimización en el control de fugas hidráulicas mediante el análisis combinatorio.** Artículo aceptado y presentado en el “VIII Congreso Internacional de Normatividad Legal, Gestión, Calidad y Competitividad Organizacional” celebrado los días 10 y 11 de Octubre del 2013 en la Ciudad de Morelia, Michoacán, México. El mismo artículo fue publicado como capítulo del libro “Herramientas para la mejora de las organizaciones del siglo XXI” con ISBN: 978-607-9096-18-2. (Anexo E)

Referencias

[1] Ian Sommerville, *Software Engineering*, 8th ed. Lancaster, Pennsylvania, United States: Addison Wesley, 2007.

[2] Alejandro Orero, José Ignacio López, and José Luis Arroyo, "Caso Lyma Getafe, S.A.M.: desarrollo e implantación de un plan estratégico de sistemas en una empresa municipal de servicios," *Revista de empresa No.19*, p. 11, 2007.

[3] Carmen De Pablos et al., *Dirección y Gestión de los Sistemas de Información de la Empresa*, 2nd ed. Madrid, España: ESIC, 2006.

[4] Kenneth C. Laudon and Jane P. Laudon, *Management Information Systems: Managing the Digital Firm*, 10th ed. New York, United States: Pearson, 2008.

[5] Chris Kimble. (2013, Septiembre) http://www.chris-kimble.com/. [Online]. http://www.chris-kimble.com/Courses/World_Med_MBA/Types-of-Information-System.html

[6] Michael Kennedy, *Introducing Geographic Information Systems with ArcGIS*, 2nd ed. Kentucky, United States: Wiley, 2009.

[7] Roger Tomlinson, *Thinking About GIS: Geographic Information System Planning for Managers*, 4th ed. Redlands, California, United States: ESRI, 2003.

[8] Ganesh Datt Bhatt and Jigish Zaveri, "The enabling role of decision support systems in organizational learning," *Elsevier No. 32 297-309*, p. 13, 2001.

[9] Jim Q. Chen and Sang M. Lee, "An exploratory cognitive DSS for strategic decision making," *Elsevier No.36 146-170*, p. 14, 2002.

[10] J.P. Shim et al., "Past, present, and future of decision support technology," *Elsevier No. 33 111-126*, p. 16, 2002.

[11] Marco Dorigo and Thomas Stützle, *Ant Colony Optimization*. Massachusetts: Bradford, 2004.

[12] Emad Elbeltagi, Tarek Hegazy, and Donald Grierson, "Comparison among five evolutionary-based optimization algorithms," *Elsevier No.19 43-53*, p. 11, 2005.

[13] Chandrasekharan Rajendran and Hans Ziegler, "Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs," *Elsevier No. 155 426-438*, p. 13, 2002.

[14] Wang Hui, "Comparison of several intelligent algorithms for solving TSP problem in industrial engineering," *Elsevier No. 4 226-235*, p. 10, 2012.

[15] Rubén Ruiz and Concepción Maroto, "A comprehensive review and evaluation of permutation flowshop heuristics," *Elsevier No. 165 479-494*, p. 16, 2005.

[16] James Kennedy and Russell Eberhart, *Swarm Intelligence*. USA: Morgan Kaufmann, 2001.

[17] V Selvi and R Umarani, "Comparative Analysis of Ant Colony and Particle Swarm Optimization Techniques," *International Journal of Computer Applications Vol. 5 No.4* , p. 6, 2010.

[18] Viggo Kann and Pierluigi Crescenzi. (2005, Julio) A compendium of NP optimization problems. [Online]. http://www.nada.kth.se/~viggo/wwwcompendium/

[19] G Ausiello et al., *Complexity and Approximation. Combinatorial Optimization Problems and Their Approximability Properties*. USA: Springer, 2003.

[20] Jon Kleinberg and Éva Tardos, *Algorithm Design*, 1st ed. Ithaca, New York, United States: Addison Wesley, 2006.

[21] Sanjoy Dasgupta, Christos Papadimitriou, and Umesh Vazirani, *Algorithms*, 1st ed. New York, United States: McGraw-Hill, 2008.

[22] César Rego, Dorabela Gamboa, Fred Glover, and Colin Osterman, "Traveling salesman problem heuristics: Leading methods, implementations and latest advances," *Elsevier No. 211 427-441*, p. 15, 2011.

[23] Paolo Toth and Daniele Vigo, *The Vehicle Routing Problem*. USA: Siam, 2002.

[24] J K Lenstra, Rinnooy Kan, and P Brucker, "Complexity of machine scheduling problems," *Annals of Discrete Mathematics*, p. 20, 1977.

[25] Ali Allahverdi, C Ng, T Cheng, and Mikhail Kovalyoc, "A surver of scheduling problems

with setup times or costs," *Elsevier*, p. 48, 2008.

[26] Ricardo Hincapié, Carlos Ríos, and Ramón Gallego, "Técnicas heurísticas aplicadas al problema del cartero viajante (TSP)," *Scientia et Technica*, p. 6, 2004.

[27] Nohaidda Binti Sariff and Norlida Buniyamin, "Comparative Study of Genetic Algorithm and Ant Colony Optimization Algorithm Performances for Robot Path Planning in Global Static Environments of Different Complexities," *IEEE*, p. 6, 2009.

[28] Russell Eberhart and Yuhui Shi, "Particle Swarm Optimzation. Developments, Aplications and Resources," *IEEE*, p. 6, 2001.

[29] Ding-Zhu Du and Panos Pardalos, *Handbook of Combinatorial Optimization*. USA: Springer, 2013.

[30] Marco Dorigo and Christian Blum, "Ant colony optimization theory: A survey," *Elsevier*, p. 36, 2005.

[31] Christian Blum, "Ant colony optimization: Introduction and recent trends," *Elsevier No. 2 353-373*, p. 21, 2005.

[32] Ümit Bilge, Müjde Kurtulan, and Furkan Kirac, "A tabu search algorithm for the single machine total weighted tardniess problem," *Elsevier No. 176 1423-1435*, p. 13, 2006.

[33] Ning Liu, M Abdelrahman, and Srini Ramaswamy, "A Genetic Algorithm for Single Machine Total Weighted Tardiness Scheduling Problem," *International journal of intelligent control and systems*, p. 8, 2005.

[34] M Fatih, Yun-Chia Liang, Mehmet Sevkli, and Gunes Gencyilmaz, "Particle Swarm Optimization Algorithm For Single Machine Total Weighted Tardiness Problem," *IEEE*, p. 8, 2004.

[35] Óscar Fuentes, Adriana Palma, and Katya Rodríguez, "Estimación y localización de fugas en una red de tuberías de agua potable usando algoritmos genéticos," *Ingeniería, Investigación y Tecnología*, p. 8, 2011.

[36] Leonel Ochoa Alejo and Victor Bourguett Ortiz, "Reducción Integral de Pérdidas de Agua Potable," Instituto Mexicano de Tecnología del Agua, México, 2001.

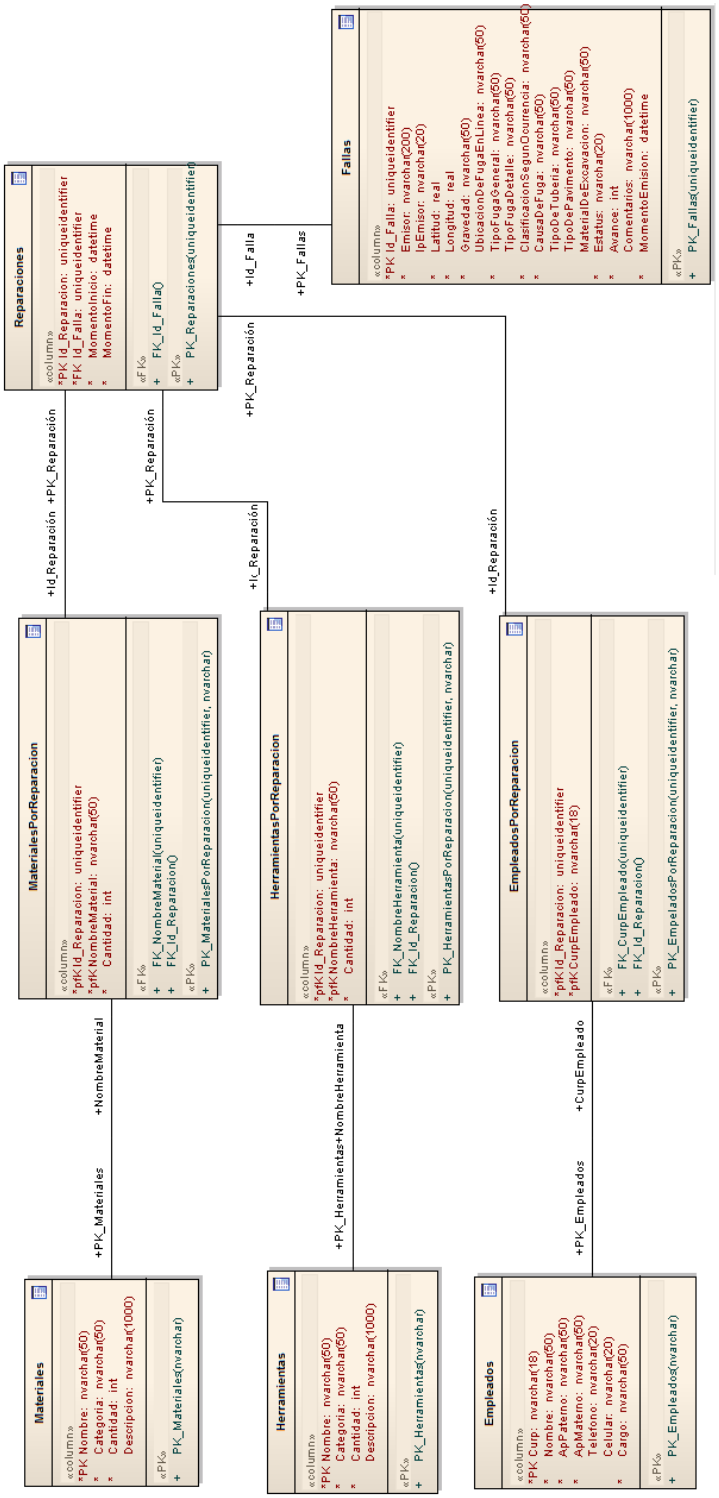
[37] Joel Shwimer, "On the n-job, One-machine, Sequence-independent Scheduling Problem with Tardiness Penalties: A Branch-Bound Solution," *Management Science*, p. 54, 1972.

[38] Federico González-Santoyo, *Probabilidad y Estadística*. Morelia, Michoacán, México: FeGoSa, 2006.

[39] Willliam Cook. (2013, Septiembre) The Traveling Salesman Problem. [Online]. http://www.math.uwaterloo.ca/tsp/index.html

[40] Gabriel Svennerberg, *Beginning Google Maps API 3*, Primera ed. New York, USA: Apress, 2010.

Anexo A. Modelo Relacional



Anexo B. Manual de Usuario

Anexo C. Manual de Instalación

Anexo D. Script para restaurar la Base de Datos

Anexo E - Congreso IAIDRES