DESARROLLO DE UN PROTOTIPO INFORMÁTICO DE CONSULTA Y ASIGNACIÓN DE PLAZAS LIBRES EN PARQUEADEROS



FACULTAD DE INGENIERÍA

ESCUELA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS Y COMPUTACIÓN

TEMA:

DESARROLLO DE UN PROTOTIPO INFORMÁTICO DE CONSULTA Y ASIGNACIÓN DE PLAZAS LIBRES EN PARQUEADEROS

DIRECTOR:

ING. ALFREDO CALDERÓN

AUTOR:

JUAN CARLOS ANDRADE

QUITO – 2012

DESARROLLO DE UN PROTOTIPO INFORMÁTICO DE CONSULTA Y ASIGNACIÓN DE PLAZAS LIBRES EN PARQUEADEROS

Declaratoria de responsabilidad

Los conceptos desarrollados, análisis realizado y las conclusiones del presente trabajo, son de exclusiva responsabilidad del autor

Juan Carlos Andrade Chávez

Dedicatoria y Agradecimiento

A Dios.

Por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos y metas, además de su infinita bondad y amor.

A mi madre Teresa.

Por haberme apoyado en todo momento, por sus consejos, sus valores, por la motivación constante que me ha permitido ser una persona de bien, pero más que nada, por su amor.

A mi padre Jorge.

Por los ejemplos de perseverancia y constancia que lo caracterizan y que me ha infundado siempre, por el valor mostrado para salir adelante y por su amor.

A mis familiares.

A mis hermanos Ana María y Jorge Luis, por ser ambos el ejemplo de hermanos mayores y de los cuales aprendí valores importantes y cómo actuar en momentos difíciles; y a todos aquellos que participaron directa o indirectamente en la elaboración de esta mi tesis.

A mis maestros.

A todos aquellos ingenieros que me apoyaron en mi formación académica y culminación de mis estudios profesionales; al Ing. Alfredo Calderón por dirigir y apoyar a la culminación de este trabajo: al Ing. Oswaldo Espinosa por su tiempo compartido y por impulsar mi motivación de buscar un posgrado; al Ing. Xavier Castellanos que con su ejemplo me inspiro a una superación intelectual y académica; a la Ing. Anita Urgiles que gracias a su experiencia en proyectos me supo guiar a soluciones prácticas.

A mis amigos.

Que nos apoyamos mutuamente en nuestra formación profesional y que hasta ahora, seguimos siendo amigos: Andrés Argoti, Jorge Recalde, Kathy Narváez, Ronald López, Alex Nieto y Carlos Guamán, por haberme ayudado a realizar este trabajo.

Finalmente a mi querida Claudia, que a pesar de la distancia, me apoyo brindándome ánimos y siendo una de las principales razones de la finalización pronta de mi tesis.

¡Gracias a ustedes!

Contenido

Declaratoria de responsabilidad 1
Dedicatoria y Agradecimiento 2
Índice de Gráficos 5
Índice de Diagramas 6
1. CAPITULO 1: Introducción 7
1.2 Metodologías de Desarrollo 7
1.1.1 Modelo en Cascada o Clásico 8
1.1.2 Modelo en espiral 10
1.1.3 RAD (Rapid Application Development) 12
1.1.4 PROCESO UNIFICADO 14
1.2 Selección de Herramientas de Desarrollo 16
1.2.1 Lenguaje 16
1.2.2 Base de Datos 19
1.2.3 IDE 23
1.2.3 Tecnologías a ser Utilizadas 25
1.3 Procesos 28
1.3.1 Clasificación 29
1.3.2 Cadena de Valor 30
1.3.3 Mapa de Procesos 31
1.1.1 Diagrama SIPOC 32
2. CAPITULO 2: Análisis de Requerimientos 34
2.2 Levantamiento de Requerimientos 34
2.1.1 Funcionales 37
2.2.1 No Funcionales 38
2.2 Diagramación de los procesos 40
2.3 Priorización de los Procesos 45
3. CAPITULO 3: Diseño-Modelamiento 46
3.2 Diseño Entidad/Relación (E/R) 46

3.2 Diagramas UML 47
3.2.1 Diagrama de Clases 47
3.2.2 Diagrama de Despliegue 51
3.2.3 Diagramas de Secuencia 52
4. CAPITULO 4: Desarrollo y Pruebas 59
4.2 Diseño del Prototipo 59
4.3 Pruebas de Ingresos 72
4.4 Pruebas de Validaciones de Usuario 77
4.4 Pruebas de Funcionamiento 81
5. CAPITULO 5: Conclusiones y Recomendaciones 84
BIBLIOGRAFÍA 85
TRABAJOS CITADOS 86
ANEXOS 89
ANEXO 1.1 90
ANEXO 1.2 92
MANUAL TÉCNICO 94
MANUAL DE USUARIO 145

Índice de Gráficos

Gráfico 1.1.1 Modelo de Cascada 7
Gráfico 1.1.2 Modelo en Espiral 10
Gráfico 1.1.3 Modelo RAD 12
Gráfico 1.1.4 Modelo del Proceso Unificado 15
Gráfico 1.2.3 Diagramación de los códigos de Barras 24
Gráfico 1.3.4 Ejemplo de Diagrama Sipoc 31
Gráfico Proceso Asignar Espacios 42
Gráfico Proceso Registrar Utilización 43

Índice de Diagramas

Diagrama 2.1 SIPOC 34
Diagrama 2.2 Modelo SIPOC 35
Cadena de Valor 39
Mapa de Procesos 41
Diagrama Entidad/Relación 45
UML 46
1. Diagrama Clases(Clase DP) 46
2. Diagrama Clases(Clase GUI) 47
3. Diagrama Clases(Clase MD) 48
4. Diagrama de Paquetes 49
5. Diagrama de Despliegue 50
Diagramas de Secuencia 51
1. Administración Vehículos(Ingresar) 52
2. Administración Vehículos (Modificar) 53
3. Administración Vehículos (Eliminar) 54
4. Consultas Vehículos (Consulta General) 55
5. Consultas Vehículos (Consulta Parámetro) 56
6. Asignar (Asignar Espacio) 57
7. Asignar (Liberar Espacio) 58

1. CAPITULO 1: Introducción

1.2 Metodologías de Desarrollo

Conceptos Generales

- **Metodología:** Conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software.
- **Tarea:** Actividades elementales en que se dividen los procesos.
- **Procedimiento:** Definición de la forma de ejecutar la tarea.
- **Técnica:** Herramienta utilizada para aplicar un procedimiento. Se pueden utilizar una o varias
- **Herramienta:** Para realizar una técnica, podemos apoyarnos en las herramientas software que automatizan su aplicación.
- **Producto:** Resultado de cada etapa.

Metodología VS Ciclo de Vida

Una metodología puede seguir uno o varios modelos de ciclo de vida, es decir, el ciclo de vida indica qué es lo que hay que obtener a lo largo del desarrollo del proyecto pero no de cómo hacerlo.

La metodología indica cómo hay que obtener los distintos productos parciales y finales.

Qué es una Metodología

La metodología es un Ciclo de Vida. Un modelo de ciclo de vida de software es una vista de las actividades que ocurren durante el desarrollo de software, intenta determinar el orden de las etapas involucradas y los criterios de transición asociadas entre estas etapas.

- Un modelo de ciclo de vida del software:
- Describe las fases principales de desarrollo de software.
- Define las fases primarias esperadas de ser ejecutadas durante esas fases.
- Ayuda a administrar el progreso del desarrollo

- Provee un espacio de trabajo para la definición de un detallado proceso de desarrollo de software.

1.1.1 Modelo en Cascada o Clásico

En los años 70 se impuso un nuevo enfoque de desarrollo del software, introducido por Royce en 1970, a través de un ciclo de vida en "cascada" (así denominado por la disposición de las distintas fases de desarrollo, en las que los resultados de una fase parecen caer en cascada hacia la siguiente fase, tal como se muestra en el Gráfico 1.1.1). [1] El método ideado por Royce constituye uno de los primeros modelos de ciclo de vida publicados, por lo que también recibe el nombre de modelo de ciclo de vida clásico. Este método modela el ciclo convencional de la Ingeniería del Software, aplicando un enfoque sistemático y secuencial de desarrollo que comienza con la ingeniería del sistema y progresa a través del análisis, diseño, codificación, pruebas y mantenimiento.

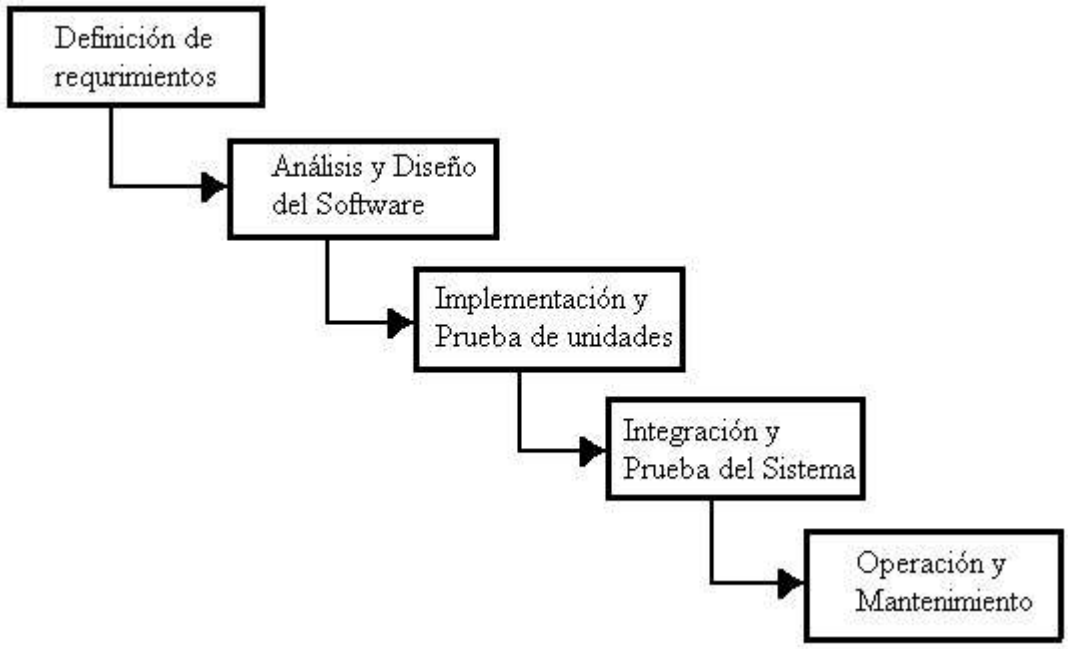


Gráfico 1.1.1 Modelo de Cascada, Fuente: Wikipedia/Software, Elaborado: wikipedia.org

Como sugiere el esquema del modelo en cascada, antes de poder avanzar a la siguiente etapa, es necesario haber finalizado completamente la etapa anterior. Asociada con cada etapa del proceso existen hitos y documentos, de tal forma que se puede utilizar el modelo para comprobar los avances del proyecto y para estimar cuánto falta para su finalización.

Este modelo es muy útil pues ayuda a los desarrolladores a comprender qué es lo que tienen que hacer en cada momento. Su simplicidad hace que resulte sencillo explicárselo a los clientes que no están familiarizados el proceso software. Además, se muestran de forma explícita qué productos intermedios se tienen que obtener antes de abordar las siguientes tareas.

[2] Una modificación sobre este modelo consiste en la introducción de una revisión y vuelta atrás, con el fin de corregir las deficiencias detectadas durante las distintas etapas, o para completar o aumentar las funcionalidades del sistema en desarrollo, resultando un diagrama de fases y etapas. De esta manera, durante cualquiera de las fases se puede retroceder momentáneamente a una fase previa para solucionar los problemas que se pudieran haber encontrado.

Ventajas:

- Es un modelo sencillo y disciplinado
- Es fácil aprender a utilizarlo y comprender su funcionamiento
- Está dirigido por los tipos de documentos y resultados que deben obtenerse al final de cada etapa
- Ha sido muy usado y, por tanto, está ampliamente contrastado
- Ayuda a detectar errores en las primeras etapas a bajo costo
- Ayuda a minimizar los gastos de planificación, pues se realiza sin problemas

Desventajas:

- Los proyectos raramente siguen el proceso lineal tal como se definía originalmente el ciclo de vida
- El cliente debe tener paciencia pues obtendrá el producto al final del ciclo de vida
- No refleja exactamente cómo se programa realmente el sistema, en el que suele haber un gran componente iterativo
- Puede resultar complicado regresar a etapas anteriores (ya acabadas) para realizar correcciones
- El producto final obtenido puede que no refleje todos los requisitos del usuario

1.1.2 Modelo en espiral

[3] El modelo en espiral, propuesto por Bohem en 1988 [BOE88], es un modelo de proceso de software evolutivo que acompaña la naturaleza evolutiva de con los aspectos controlados y sistemáticos del ciclo de vida tradicional. Proporciona el potencial para el desarrollo rápido de versiones incrementales del software. En este modelo, el sistema se desarrolla en una serie de versiones incrementales. Durante las primeras iteraciones, la versión incremental podría ser un modelo en papel o un prototipo. Durante las últimas iteraciones se producen versiones cada vez más completas de ingeniería del sistema. .

El Modelo en Espiral se divide en un número de actividades estructurales, también llamadas "regiones de tareas". (Ver Gráfico 1.1.2)

Generalmente existen entre tres y seis regiones de tareas:

1. Comunicación con el cliente.- Las tareas requeridas para establecer comunicación entre el desarrollador y el cliente, sea revisar especificaciones, plantear necesidades, etc.
2. Planificación.- Las tareas requeridas para definir recursos, tiempos e información relacionada con el proyecto.
3. Análisis de riesgos.- Las tareas requeridas para evaluar riesgos técnicos y de gestión.
4. Ingeniería.- Las tareas requeridas para construir una o más representaciones de la aplicación
5. Construcción y adaptación.- Las tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario.
6. Evaluación del cliente.- Las tareas requeridas para obtener la reacción del cliente, según la evaluación de las representaciones del software creadas durante la etapa de ingeniería e implementada durante la etapa de instalación

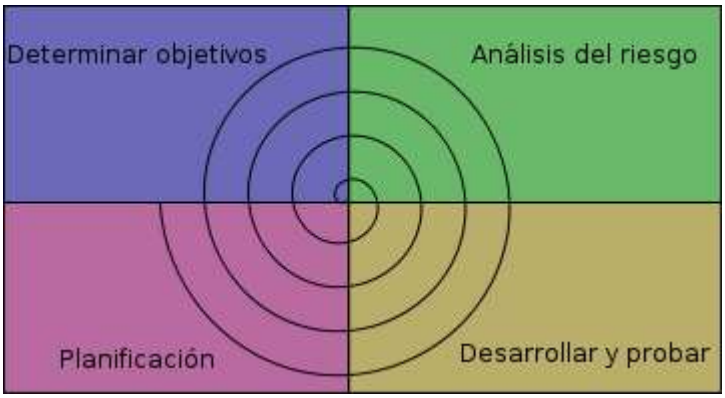


Gráfico 1.1.2 Modelo en Espiral, Fuente: scruz334.blogspot.es, Elaborado: Santiago Cruz

[4] El modelo de desarrollo en espiral es un generador de modelo de proceso guiado por el riesgo que se emplea para conducir sistemas intensivos de ingeniería de software concurrente y a la vez con muchos usuarios.

Se caracteriza principalmente por:

- Un enfoque cíclico para el crecimiento incremental del grado de definición e implementación de un sistema, mientras que disminuye su grado de riesgo.
- Un conjunto de puntos de fijación para asegurar el compromiso del usuario con soluciones de sistema que sean factibles y mutuamente satisfactorias.

El modelo espiral captura algunos principios básicos:

- Decidir qué problema se quiere resolver antes de viajar a resolverlo.
- Examinar tus múltiples alternativas de acción y elegir una de las más convenientes.
- Evaluar qué tienes hecho y qué tienes que haber aprendido después de hacer algo.
- No ser tan ingenuo para pensar que el sistema que estás construyendo será "EL" sistema que el cliente necesita
- Conocer (comprender) los niveles de riesgo, que tendrás que tolerar.

El modelo espiral no es una alternativa del modelo cascada, ellos son completamente compatibles.

1.1.3 RAD (Rapid Application Development)

El desarrollo rápido de aplicaciones o RAD (Rapid Application Development) es un proceso de desarrollo de software, desarrollado inicialmente por James Martin en 1980. El método comprende el desarrollo iterativo, la construcción de prototipos y el uso de utilidades CASE.

[5] Tradicionalmente, el desarrollo rápido de aplicaciones tiende a englobar también la usabilidad, utilidad y la rapidez de ejecución. El Desarrollo Rápido de Aplicaciones (DRA) (Rapid Application Development RAD) es un modelo de proceso del desarrollo del software lineal secuencial que enfatiza un ciclo de desarrollo extremadamente corto (ver Gráfico 1.1.3). DRA es una adaptación a "Alta velocidad" en el que se logra el desarrollo rápido utilizando un enfoque de construcción basado en componentes. Si se comprenden bien los requisitos y se limita el ámbito del proyecto, el proceso DRA permite al equipo de desarrollo crear un "sistema completamente funcional" dentro de periodos cortos de tiempo. Cuando se utiliza principalmente para aplicaciones de sistemas de información, el enfoque DRA comprende las siguientes fases:

- Modelado de gestión: el flujo de información entre las funciones de gestión se modela de forma que responda a las siguientes preguntas: ¿Qué información conduce el proceso de gestión? ¿Qué información se genera? ¿Quién la genera? ¿A dónde va la información? ¿Quién la proceso?
- Modelado de datos: el flujo de información definido como parte de la fase de modelado de gestión se refina como un conjunto de objetos de datos necesarios para apoyar la empresa. Se definen las características (llamadas atributos) de cada uno de los objetos y las relaciones entre estos objetos.
- Modelado de proceso: los objetos de datos definidos en la fase de modelado de datos quedan transformados para lograr el flujo de información necesario para implementar una función de gestión. Las descripciones del proceso se crean para añadir, modificar, suprimir, o recuperar un objeto de datos. Es la comunicación entre los objetos.
- Generación de aplicaciones: El DRA asume la utilización de técnicas de cuarta generación. En lugar de crear software con lenguajes de programación de tercera generación, el proceso DRA trabaja para volver a utilizar componentes de programas

ya existentes (cuando es posible) o a crear componentes reutilizables (cuando sea necesario). En todos los casos se utilizan herramientas automáticas para facilitar la construcción del software.

- Pruebas de entrega: Como el proceso DRA enfatiza la reutilización, ya se han comprobado muchos de los componentes de los programas. Esto reduce tiempo de pruebas. Sin embargo, se deben probar todos los componentes nuevos y se deben ejercitar todas las interfaces a fondo.

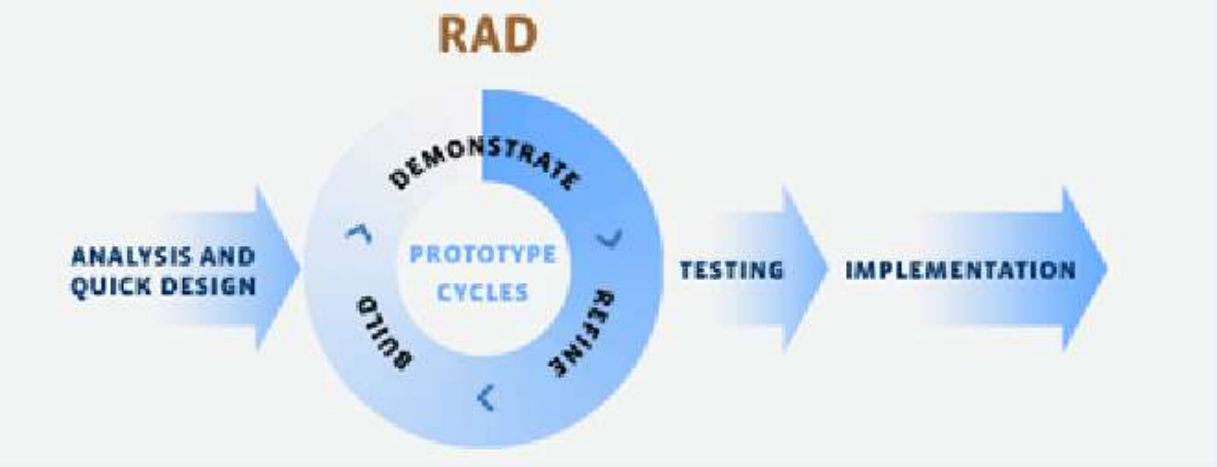


Gráfico 1.1.3 Modelo RAD, Fuente: software-document.blogspot.com, Elaborado: Software Engineering Blog

Otra versión de fases RAD

- Modelado de Negocios: El flujo de información entre las funciones de la empresa se define por responder a preguntas como qué tipo de información conduce el proceso de negocio, la información que se genera, que lo genera, dónde se va la información, que la procesan y así sucesivamente.
- Modelado de datos: La información obtenida de modelado de negocio se refina en un conjunto de objetos de datos (entidades) que se necesitan para apoyar el negocio. Los atributos (carácter de cada entidad) se identifican y la relación entre estos objetos de datos (entidades) es definido.
- Modelado de Procesos: Los objetos de datos definidos en la fase de modelado de datos son transformados para lograr el flujo de información necesario para implementar una función de negocios. Descripciones de proceso se crean para añadir, modificar, borrar o recuperar un objeto de datos.
- Generación de aplicación: Las herramientas automatizadas se utilizan para facilitar la construcción del software, e incluso utilizan las técnicas GL cuarto.

- Pruebas y Voltee: Muchos de los componentes de programación ya han sido probados desde hace hincapié en la reutilización RAD. Esto reduce el tiempo de prueba en general. Pero los nuevos componentes debe ser probado y todas las interfaces deben ejercerse plenamente.

1.1.4 PROCESO UNIFICADO

[6] El Proceso Unificado es un proceso de desarrollo de software: "conjunto de actividades necesarias para transformar los requisitos del usuario en un sistema software".

RUP es un marco genérico que puede especializarse para una variedad de tipos de sistemas, diferentes áreas de aplicación, tipos de organizaciones, niveles de aptitud y diferentes tamaños de proyectos.

RUP está basado en componentes. El software esta formado por componentes software interconectados a través de interfaces.

RUP está dirigido por casos de uso, centrado en la arquitectura, y es iterativo e incremental.

Dirigido por Casos de Uso

- Un caso de uso es un fragmento de funcionalidad del sistema que proporciona un resultado de valor a un usuario. Los casos de uso modelan los requerimientos funcionales del sistema.
- Todos los casos de uso juntos constituyen el modelo de casos de uso.
- Los casos de uso también guían el proceso de desarrollo (diseño, implementación, y prueba). Basándose en los casos de uso los desarrolladores crean una serie de modelos de diseño e implementación que llevan a cabo los casos de uso. De este modo los casos de uso no solo inician el proceso de desarrollo sino que le proporcionan un hilo conductor, avanza a través de una serie de flujos de trabajo que parten de los casos de uso.

Centrado en la Arquitectura

La arquitectura de un sistema software se describe mediante diferentes vistas del sistema en construcción.

El concepto de arquitectura software incluye los aspectos estáticos y dinámicos más significativos del sistema.

La arquitectura es una vista del diseño completo con las características más importantes resaltadas, dejando los detalles de lado.

> Arquitectura: Conjunto de decisiones significativas acerca de la organización de un sistema software, la selección de los elementos estructurales a partir de los cuales se compone el sistema, las interfaces entre ellos, su comportamiento, sus colaboraciones, y su composición.

Los casos de uso y la arquitectura están profundamente relacionados. Los casos de uso deben encajar en la arquitectura, y a su vez la arquitectura debe permitir el desarrollo de todos los casos de uso requeridos, actualmente y a futuro.

El arquitecto desarrolla la forma o arquitectura a partir de la comprensión de un conjunto reducido de casos de uso fundamentales o críticos, podemos decir que el arquitecto:

- Crea un esquema en borrador de la arquitectura comenzando por la parte no específica de los casos de uso (por ejemplo la plataforma) pero con una comprensión general de los casos de uso fundamentales.
- A continuación, trabaja con un conjunto de casos de usos claves o fundamentales. Cada caso de uso es especificado en detalle y realizado en términos de subsistemas, clases, y componentes.
- A medida que los casos de uso se especifican y maduran, se descubre más de la arquitectura, y esto a su vez lleva a la maduración de más casos de uso.

Este proceso continúa hasta que se considere que la arquitectura es estable.

El Proceso Unificado se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo constituye una versión del sistema. (Ver Gráfico 1.1.4)

Fases: Cada ciclo constas de cuatro fases: inicio, elaboración, construcción, y transición.



Gráfico 1.1.4 Modelo del Proceso Unificado, Fuente: chaco.gov.ar/ApuntesRUP, Elaborado: Gustavo Torossi

1.2 Selección de Herramientas de Desarrollo

1.2.1 Lenguaje

[7] Java es un lenguaje de programación y la primera plataforma informática creada por Sun Microsystems en 1995. Es la tecnología subyacente que permite el uso de programas punteros, como herramientas, juegos y aplicaciones de negocios. Java se ejecuta en más de 850 millones de ordenadores personales de todo el mundo y en miles de millones de dispositivos, como dispositivos móviles y aparatos de televisión.

El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. La memoria es gestionada mediante un recolector de basura.

Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible.

Historia

[8] Java se creó como una herramienta de programación para ser usada en un proyecto de set-top-box en una pequeña operación denominada the Green Project en Sun Microsystems en el año 1991. El equipo (Green Team), compuesto por trece personas y dirigido por James Gosling, trabajó durante 18 meses en Sand Hill Road en Menlo Park en su desarrollo.

El lenguaje se denominó inicialmente Oak (por un roble que había fuera de la oficina de Gosling), luego pasó a denominarse Green tras descubrir que Oak era ya una marca comercial registrada para adaptadores de tarjetas gráficas y finalmente se renombró a Java.

Filosofía

- El lenguaje Java se creó con cinco objetivos principales:
- Debería usar el paradigma de la programación orientada a objetos.
- Debería permitir la ejecución de un mismo programa en múltiples sistemas operativos.
- Debería incluir por defecto soporte para trabajo en red.
- Debería diseñarse para ejecutar código en sistemas remotos de forma segura.
- Debería ser fácil de usar y tomar lo mejor de otros lenguajes orientados a objetos, como C++.

Para conseguir la ejecución de código remoto y el soporte de red, los programadores de Java a veces recurren a extensiones como CORBA (Common Object Request Broker Architecture), Internet Communications Engine o OSGi respectivamente.

Orientado a objetos

La primera característica, orientado a objetos (“OO”), se refiere a un método de programación y al diseño del lenguaje. Aunque hay muchas interpretaciones para OO, una primera idea es diseñar el software de forma que los distintos tipos de datos que usen estén

unidos a sus operaciones. Así, los datos y el código (funciones o métodos) se combinan en entidades llamadas objetos. Un objeto puede verse como un paquete que contiene el "comportamiento" (el código) y el "estado" (datos). El principio es separar aquello que cambia de las cosas que permanecen inalterables. Frecuentemente, cambiar una estructura de datos implica un cambio en el código que opera sobre los mismos, o viceversa. Esta separación en objetos coherentes e independientes ofrece una base más estable para el diseño de un sistema software. El objetivo es hacer que grandes proyectos sean fáciles de gestionar y manejar, mejorando como consecuencia su calidad y reduciendo el número de proyectos fallidos.

Independencia de la plataforma

La segunda característica, la independencia de la plataforma, significa que programas escritos en el lenguaje Java pueden ejecutarse igualmente en cualquier tipo de hardware. Este es el significado de ser capaz de escribir un programa una vez y que pueda ejecutarse en cualquier dispositivo, tal como reza el axioma de Java, ‘’’write once, run anywhere’’’.

Para ello, se compila el código fuente escrito en lenguaje Java, para generar un código conocido como "bytecode" (específicamente Java bytecode)—instrucciones máquina simplificadas específicas de la plataforma Java. Esta pieza está "a medio camino" entre el código fuente y el código máquina que entiende el dispositivo destino. El bytecode es ejecutado entonces en la máquina virtual (JVM), un programa escrito en código nativo de la plataforma destino (que es el que entiende su hardware), que interpreta y ejecuta el código. Además, se suministran bibliotecas adicionales para acceder a las características de cada dispositivo (como los gráficos, ejecución mediante hebras o threads, la interfaz de red) de forma unificada. Se debe tener presente que, aunque hay una etapa explícita de compilación, el bytecode generado es interpretado o convertido a instrucciones máquina del código nativo por el compilador JIT (Just In Time).

1.2.2 Base de Datos

Oracle es un sistema de gestión de base de datos objeto-relacional (o ORDBMS por el acrónimo en inglés de Object-Relational Data Base Management System), desarrollado por Oracle Corporation.



Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacando:

- Soporte de transacciones,
- Estabilidad,
- Escalabilidad y
- Soporte multiplataforma.

Su dominio en el mercado de servidores empresariales ha sido casi total hasta hace poco, recientemente sufre la competencia del Microsoft SQL Server de Microsoft y de la oferta de otros RDBMS con licencia libre como PostgreSQL, MySQL o Firebird. Las últimas versiones de Oracle han sido certificadas para poder trabajar bajo GNU/Linux.

Historia

[9] Oracle surge en 1977 bajo el nombre de SDL (Software Development Laboratories), luego en 1979 SDL cambia su nombre porRelational Software, Inc. (RSI). La fundación de Software Development Laboratories (SDL) fue motivada principalmente a partir de un estudio sobre los SGBD (Sistemas Gestores de Base de Datos) de George Koch. Computer World definió este estudio como uno de los más completos jamás escritos sobre bases de datos. Este artículo incluía una comparativa de productos que erigía a Relational Software como el más completo desde el punto de vista técnico. Esto se debía a que usaba la filosofía de las bases de datos relacionales, algo que por aquella época era todavía desconocido.

En la actualidad, Oracle (Nasdaq: ORCL) todavía encabeza la lista. La tecnología Oracle se encuentra prácticamente en todas las industrias alrededor del mundo y en las oficinas de 98 de las 100 empresas Fortune 100. Oracle es la primera compañía de software que desarrolla e implementa software para empresas 100 por ciento activado por Internet a través de toda su línea de productos: base de datos, aplicaciones comerciales y herramientas de desarrollo de aplicaciones y soporte de decisiones. Oracle es el proveedor mundial líder de software para administración de información, y la segunda empresa de software.

Oracle a partir de la versión 10g Release 2, cuenta con 6 ediciones:

- Oracle Database Enterprise Edition (EE).
- Oracle Database Standard Edition (SE).
- Oracle Database Standard Edition One (SE1).
- Oracle Database Express Edition (XE).
- Oracle Database Personal Edition (PE).
- Oracle Database Lite Edition (LE).

La única edición gratuita es la Express Edition, que es compatible con las demás ediciones de Oracle Database 10gR2 y Oracle Database 11g.

Recientemente, Oracle adquirió a Sun Microsystems y con ella la empresa encargada comercial de MySQL.

Estructuras de Oracle

Una BD Oracle tiene una estructura física y una estructura lógica:

- La estructura física se corresponde a los ficheros del sistema operativo.
- La estructura lógica está formada por los tablespace y los objetos de un esquema de BD

Estructura lógica.

Se divide en unidades de almacenamiento lógicas: Tablespaces. Cada BD estará formada por uno o mas tablespaces (al menos existe el tablespace SYSTEM catálogo del sistema)

Cada tablespace se corresponde con uno o más ficheros de datos.

Objetos: tablas, vistas, índices asociados a una tabla, clusters.

Estructura física:

Una B.D. tiene uno o más ficheros de datos. Estos ficheros son de tamaño fijo y se establecen en el momento en que se crea la base de datos o en el momento en el que se crean tablespaces.

Los datos del fichero de datos son leídos cuando se necesitan y situados en una caché de memoria compartida para que el próximo acceso a los mismos sea más rápido.

Ventajas

- Oracle es el motor de base de datos relacional más usado a nivel mundial [10].
- Puede ejecutarse en todas las plataformas, desde una Pc hasta un supercomputador.
- Oracle soporta todas las funciones que se esperan de un servidor "serio": un lenguaje de diseño de bases de datos muy completo (PL/SQL) que permite implementar diseños "activos", con triggers y procedimientos almacenados, con una integridad referencial declarativa bastante potente.
- Permite el uso de particiones para la mejora de la eficiencia, de replicación e incluso ciertas versiones admiten la administración de bases de datos distribuidas.
- El software del servidor puede ejecutarse en multitud de sistemas operativos.
- Existe incluso una versión personal para Windows 9x, lo cual es un punto a favor para los desarrolladores que se llevan trabajo a casa.
- ¿Qué hay de los objetos de Oracle? Este sistema ha comenzado a evolucionar en esta dirección, añadiendo tipos de clases, referencias, tablas anidadas, matrices y otras estructuras de datos complejas. Desafortunadamente, la implementación actual de las mismas no ofrece una ventaja clara en eficiencial, como sería de esperar, y sí provocan la incompatibilidad de los diseños que aprovechan las nuevas características con otras bases de datos.
- Oracle es la base de datos con más orientación hacía INTERNET
- Un aceptable soporte

Desventajas

- Una de las versiones más recientes de Oracle es la 8 punto algo (Aunque ya está la 9i). Y es que desde el lanzamiento original de la 8 se sucedieron varias versiones con correcciones, hasta alcanzar la estabilidad en la 8.0.3. El motivo de tantos fallos fue, al parecer, la remodelación del sistema de almacenamiento por causa de la introducción de extensiones orientadas a objetos.
- El mayor inconveniente de Oracle es quizás su precio. Incluso las licencias de Personal Oracle son excesivamente caras, en mi opinión. Otro problema es la necesidad de ajustes. Un error frecuente consiste en pensar que basta instalar el Oracle en un servidor y enchufar directamente las aplicaciones clientes. Un Oracle mal configurado puede ser desesperantemente lento.
- También es elevado el coste de la formación, y sólo últimamente han comenzado a aparecer buenos libros sobre asuntos técnicos distintos de la simple instalación y administración.

1.2.3 IDE

[11] NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos.



[12] La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software.

NetBeans IDE permite rápida y fácilmente desarrollar Java de escritorio, móviles y aplicaciones web, mientras que también proporciona una gran herramienta para PHP y C / C + + desarrolladores.

Mejor soporte para las últimas tecnologías Java

NetBeans IDE proporciona soporte de primera clase completo para las últimas tecnologías Java y las mejoras más recientes de Java antes de otros IDE. Es el primer IDE proporciona soporte para JDK 7, Java EE 6 y 2,0 JavaFX.

Con su constante mejora Editor Java, muchas características avanzadas y una amplia gama de herramientas, plantillas y ejemplos, NetBeans IDE establece el estándar para el desarrollo de tecnologías de vanguardia fuera de la caja.

La Plataforma NetBeans

La Plataforma NetBeans es una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones.

La plataforma ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma están:

- Administración de las interfaces de usuario (ej. menús y barras de herramientas)
- Administración de las configuraciones del usuario
- Administración del almacenamiento (guardando y cargando cualquier tipo de dato)
- Administración de ventanas
- Framework basado en asistentes (diálogos paso a paso)

NetBeans IDE

El IDE NetBeans es un entorno de desarrollo integrado - una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación. Existe además un

número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso.

El NetBeans IDE es un IDE de código abierto escrito completamente en Java usando la plataforma NetBeans. El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Entre sus características se encuentra un sistema de proyectos basado en Ant, control de versiones y refactoring.

1.2.3 Tecnologías a ser Utilizadas

Códigos de Barras

[13] El código de barras es un sistema de codificación que consta de una serie de líneas y espacios paralelos de diferente grosor que contienen información; los datos almacenados pueden ser captados de una manera rápida y con una gran precisión.

Este sistema proporciona un método simple y fácil para codificar la información tanto numérica como de texto, que puede ser leída por lectores electrónicos. (Ver Gráfico 1.2.3)

¿Cómo funcionan?

El lector de código de barras decodifica la información a través de la digitalización proveniente de una fuente de luz reflejada en el código y luego se envía la información a una computadora como si la información hubiese sido ingresada por teclado.

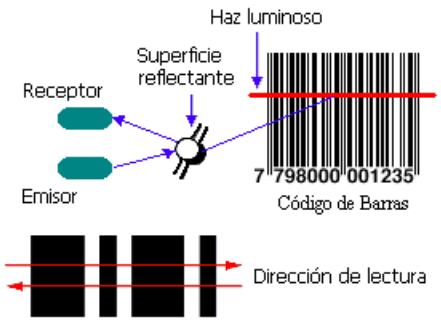


Gráfico 1.2.3 Diagramación de los códigos de Barras, Fuente: sznet.com.ar, Elaborado: SZnet

El procedimiento: el símbolo de código de barras es iluminado por una fuente de luz visible o infrarrojo, las barras oscuras absorben la luz y los espacios las reflejan nuevamente hacia un escáner.

El escáner transforma las fluctuaciones de luz en impulsos eléctricos los cuales copian las barras y el modelo de espacio en el código de barras. Un decodificador usa algoritmos matemáticos para traducir los impulsos eléctricos en un código binario y transmite el mensaje decodificado a un terminal manual, PC, o sistema centralizado de computación.

El decodificador puede estar integrado al escáner o ser externo al mismo. Los escáners usan diodos emisores de luz visible o infrarroja (LED), láser de Helio-Neón o diodes láser de estado sólido (visibles o inflarrojos) con el fin de leer el símbolo.

Algunos de ellos necesitan estar en contacto con el símbolo, otros leen desde distancias de hasta varios pies. Algunos son estacionarios, otros portátiles como los escáners manuales.

Digitalización: Los procesos de digitalización, consistentes en la transformación de documentos físicos a imágenes digitales para su visualización instantánea desde cualquier equipo de cómputo

Fluctuación: Diferencia entre el valor instantáneo de una cantidad y su valor normal.

Decodificador: Un decodificador o descodificador es un circuito combinacional, cuya función es inversa a la del codificador, esto es, convierte un código binario de entrada (natural, BCD, etc.) de N bits de entrada y M líneas de salida, tales que cada línea de salida será activada para una sola de las combinaciones posibles de entrada.

Diodo: Un diodo es un componente electrónico de dos terminales que permite la circulación de la corriente eléctrica a través de él en un solo sentido.

Utilidad

El código de barras ha sido creado para identificar objetos y facilitar el ingreso de información, eliminando la posibilidad de error en la captura.

En la actualidad la Tecnología de Código de Barras es utilizada en muchas áreas ya que ha probado ser adaptable y exitosa para los propósitos de una identificación automática de

productos. El campo de acción que abarca va desde la recepción de los materiales, su procesamiento, hasta su despacho final.

1.3 Procesos

Definiciones

[14] Un proceso es un conjunto de actividades o eventos (coordinados u organizados) que se realizan o suceden (alternativa o simultáneamente) bajo ciertas circunstancias con un fin determinado.

[15] Un proceso en informática es la ejecución de un conjunto de instrucciones entregadas a la CPU, para el cumplimiento de una etapa específica señalada por los comandos de algún programa.

Un proceso (del latín processuss) es un conjunto de actividades o eventos que se realizan o suceden con un determinado fin.

Conjunto de actividades que convierten insumos (inputs) en productos (outputs) de mayor valor para el cliente.

Conjunto de actividades secuenciales que realizan una transformación de una serie de inputs (material, mano de obra, capital, información,etc.) en los outputs deseados (bienes y/o servicios) añadiendo valor.

PROCEDIMIENTO: Sucesión. Serie de cosas que siguen cada una a otra

PROCESO:

- Marcha hacia delante (progreso)
- Desarrollo o marcha de una cosa

Qué es un proceso

- Componente de un sistema que ejecuta una acción
- Transformación de uno o más inputs en uno o más outputs usando mecanismos y bajo ciertos controles
- Cada proceso es representado por una caja la cual tiene un nombre y un número de nodo

- El nombre de un proceso debe ser una “frase verbal activa” que describa las actividades que representa.
- El número de nodo es un Id que refleja su posición en la estructura del proceso.

Tipos de procesos

Procesos Centrales: Son aquellos claves para satisfacer los requerimientos y necesidades del cliente

Procesos de Soporte: Son aquellos necesarios para que los procesos centrales operen eficaz y eficientemente

Son de apoyo y generalmente tienen poco contacto con el cliente externo.

1.3.1 Clasificación

Procesos Creadores de Valor: Son aquellos que permiten intercambiar valor entre el cliente y la institución

Procesos Habilitantes: Son aquellos que facilitan que los creadores de valor se realicen eficazmente

Procesos Creadores de Ventajas Competitivas: Desarrollan y consolidan las ventajas competitivas que la organización dispone.

Procesos Gobernantes: Direccionan el desempeño de todos los procesos que se desarrollan en la organización.

1.3.2 Cadena de Valor

[16] La cadena de valor empresarial, o cadena de valor, es un modelo teórico que permite describir el desarrollo de las actividades de una organización empresarial generando valor al cliente final.



Se conoce como cadena de valor a un concepto teórico que describe el modo en que se desarrollan las acciones y actividades de una empresa. En base a la definición de cadena, es posible hallar en ella diferentes eslabones que intervienen en un proceso económico: se inicia con la materia prima y llega hasta la distribución del producto terminado. En cada eslabón, se añade valor, que, en términos competitivos, está entendido como la cantidad que los consumidores están dispuestos a abonar por un determinado producto o servicio.

[17] El análisis de la cadena de valor permite optimizar el proceso productivo, ya que puede apreciarse, al detalle y en cada paso, el funcionamiento de la compañía. La reducción de costos y la búsqueda de eficiencia en el aprovechamiento de los recursos suelen ser los principales objetivos del empresario a la hora de revisar la cadena de valor. De esta manera, toda firma consigue ampliar su margen (resultado de la diferencia que se obtiene al comparar el valor total con el costo de las actividades).

Por otra parte, el estudio de la cadena de valor posibilita lograr una ventaja estratégica, ya que existe la chance de generar una propuesta de valor que resulte única en el mercado.

Existen dos subsistemas en la conformación de la cadena de valor. Una cadena de demanda, que involucra a los procesos vinculados con la creación de la demanda, y una cadena de suministros, dedicada a la satisfacción de la demanda en tiempo y forma.

También es posible diferenciar entre dos tipos de actividades de valor. Uno de ellos es la relacionada a las actividades primarias, un grupo de acciones enfocadas en la elaboración física de cada producto y su transferencia al comprador. Las actividades de apoyo, en cambio, sustentan a las primarias y suponen la participación de los recursos humanos, los insumos y la tecnología.

1.3.3 Mapa de Procesos

[18] El mapa de procesos presenta una visión general del sistema organizacional de su empresa, en donde además se presentan los procesos que lo componen así como sus relaciones principales. Dentro de los procesos cabe destacar gestión de la organización como planificación estratégica, establecimiento de políticas, procesos de medición, análisis y mejora. Estos últimos incluyen procesos para medir y obtener datos sobre el análisis del desempeño y mejora de la efectividad y eficiencia, pueden incluir la medición, seguimiento y procesos de auditoría, acciones correctivas y preventivas y ser aplicados a todos los procesos de la organización siendo una parte integral en la gestión.

El mapa de proceso contribuye a hacer visible el trabajo que se lleva a cabo en una unidad de una forma distinta a la que ordinariamente lo conocemos, A través de este tipo de gráfica podemos percatarnos de tareas o pasos que a menudo pasan desapercibidos en el día a día, y que sin embargo, afectan positiva o negativamente el resultado final del trabajo. Los mapas de proceso nos permite identificar claramente los individuos que intervienen en el proceso, la tarea que realizan, a quién afectan cuando su trabajo no se realiza correctamente y el valor de cada tarea o su contribución al proceso. [19] También nos permite evaluar cómo se entrelazan las distintas tareas que se requieren para completar el trabajo, si son paralelas o secuénciale. En los mapas de procesos se representa uno y cada uno de los procesos que componen un sistema así como sus relaciones principales. Dichas relaciones se indican mediante gráficos en forma de mapas conceptuales los cuales representan los flujos de información.

Los mapas de procesos son útiles para:

- Conocer cómo se llevan a cabo los trabajos actualmente- analizar los pasos del proceso para reducir el ciclo de tiempo o aumentar la calidad.

- Utilizar el proceso actual como punto de partida para llevar a cabo proyectos de mejoramiento del proceso.
- Orientar a nuevos empleados.
- Desarrollar formas alternas de realizar el trabajo en momentos críticos.
- Evaluar, establecer o fortalecer los indicadores o medidas de resultados.

1.1.1 Diagrama SIPOC

[20] La definición de un proceso en la estructura básica que debe de existir para identificar la interrelación y delimitación de los procesos y como una entrada para la optimización de los mismos.

SIPOC es una herramienta que consiste en un diagrama, que permite visualizar al proceso de manera sencilla y general. Este esquema puede ser aplicado a procesos de todos los tamaños y a todos los niveles, incluso a una organización completa. (Ver Gráfico 1.3.4)

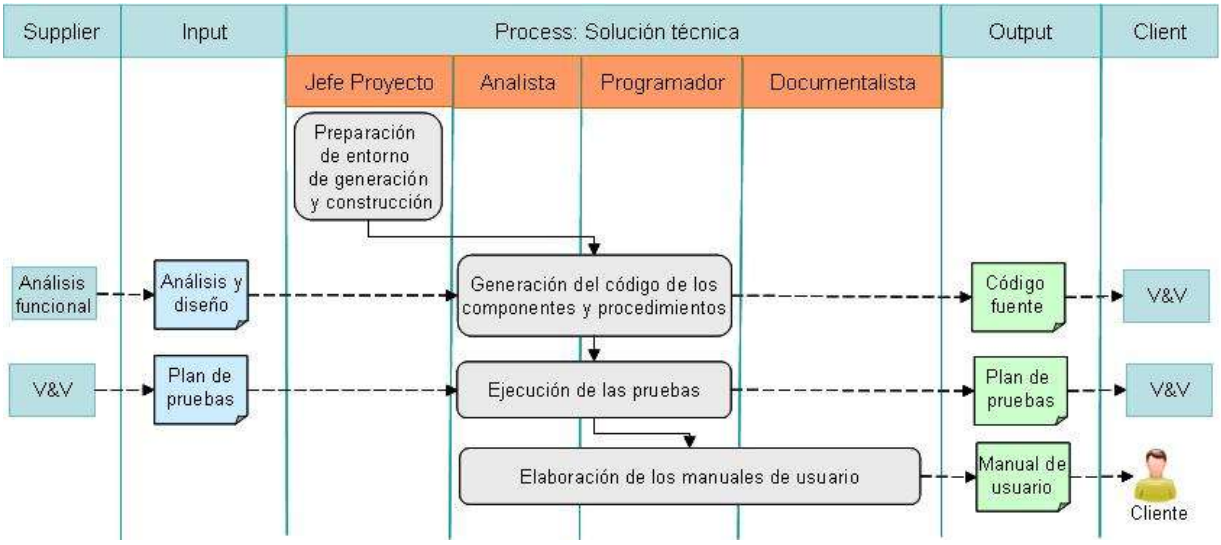


Gráfico 1.3.4 Ejemplo de Diagrama Sipoc Fuente: http://teodorabozheva.blogspot.com/2011/03/sipoc-la-definicion-de-tu-proceso-en.html Elaborado: Teodora Bozheva

SIPOC

- Suppliers = Proveedores
- Inputs = Insumos
- Process = Proceso
- Outputs =Salidas
- Customers = Clientes

[21] El diagrama SIPOC es particularmente útil a la hora de identificar:

- ✓ Qué es necesario como entradas para que se ejecute el proceso
- ✓ Quién proporciona las entradas para el proceso
- ✓ Quién es el verdadero cliente del proceso
- ✓ Qué necesita el cliente del proceso
- ✓ Cuál es el propósito y el alcance del proceso
- ✓ Cómo medir el rendimiento del proceso
- ✓ Cuáles de las actividades del proceso aportan valor para el cliente o el negocio (y por lo tanto hay que mantenerlas) y cuáles son inútiles (y por lo tanto hay que tratar de eliminarlas)

El diagrama SIPOC te ayuda en las siguientes actividades:

- Entender bien el proceso:
  - Su propósito y alcance
  - Pasos que se tienen que dar para realizarlo
  - Qué recursos se necesitan para ejecutar el proceso
  - Qué roles están involucrados en el proceso
  - Cuáles son las actividades que aportan valor
  - Definir métricas de los procesos
  - Identificar puntos de mejora en el proceso
  - Entender los puentes entre el proceso y otros procesos

2. CAPITULO 2: Análisis de Requerimientos

2.2 Levantamiento de Requerimientos

Para el levantamiento vamos a utilizar tres conceptos:

- Identificación de Actores: Son los usuarios involucrados directa o indirectamente en el uso del sistema.
- Descripción de funcionalidades: El cual es una abstracción que describe una clase de escenarios y procesos que cumplirá el sistema.
- Diagrama SIPOC: Herramienta de representación gráfica del proceso (Ver Diagrama 2.1)

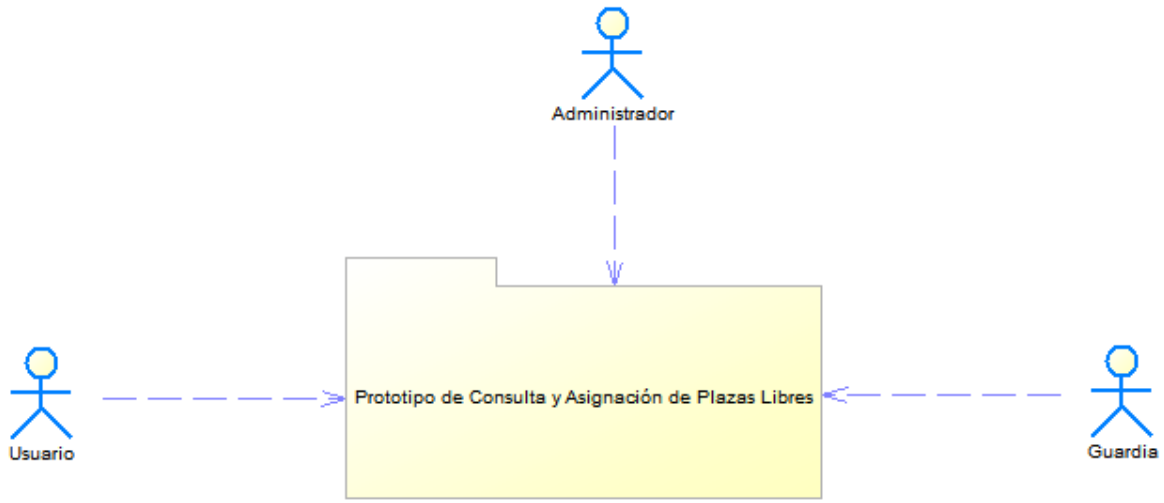
Identificación de Actores

Dentro del sistema hemos identificado a tres principales actores que intervienen dentro de nuestro prototipo.

Usuarios: Las personas involucradas dentro del proceso de registro de los vehículos en el sistema.

Guardias: Las personas involucradas con la seguridad del parqueadero.

Administrador: La persona encargada de registros y modificaciones en el sistema.



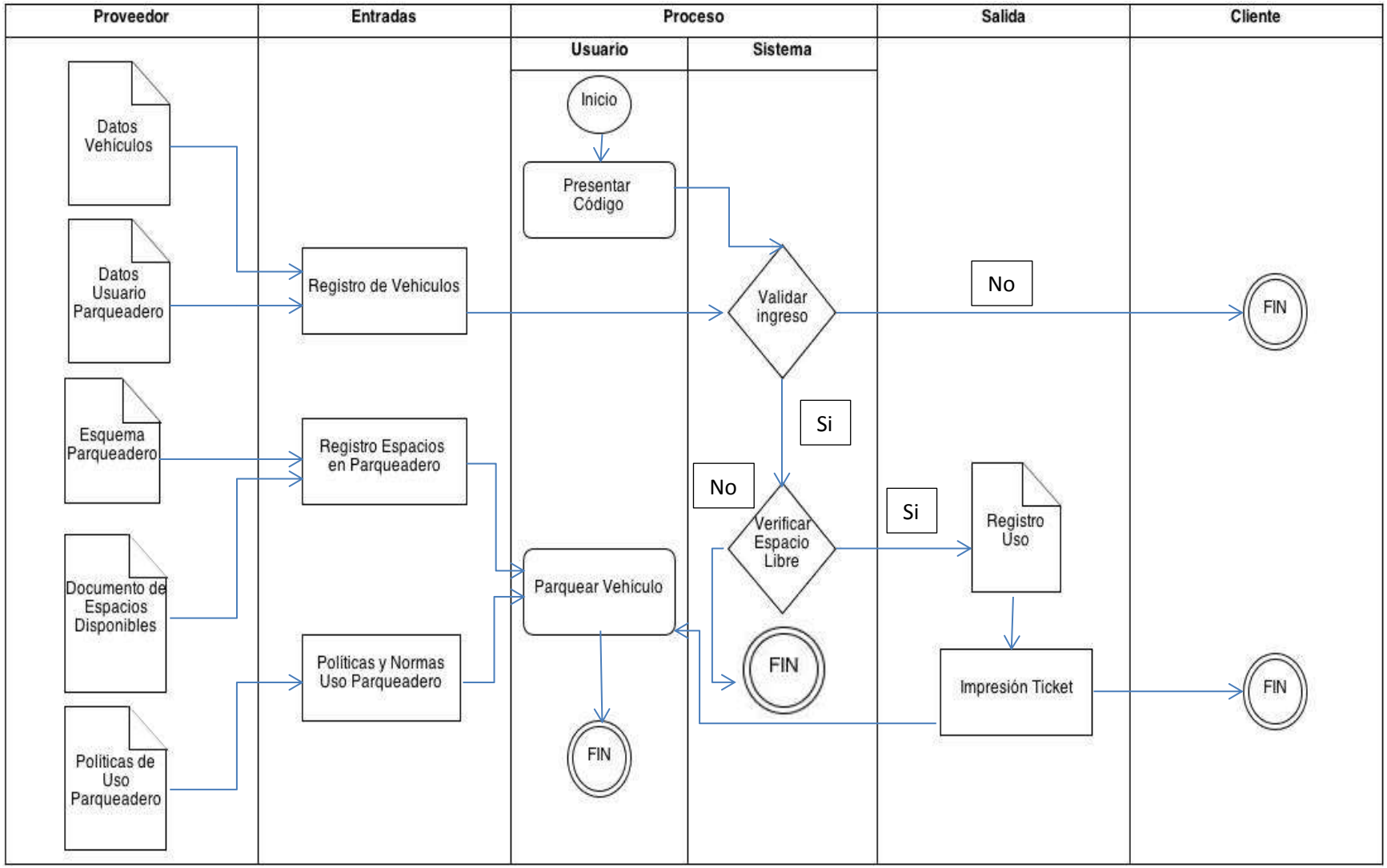


Diagrama 2.1 SIPOC Fuente: http://teodorabozheva.blogspot.com/2011/03/sipoc-la-definicion-de-tu-proceso-en.html, Elaborado: Juan Carlos Andrade

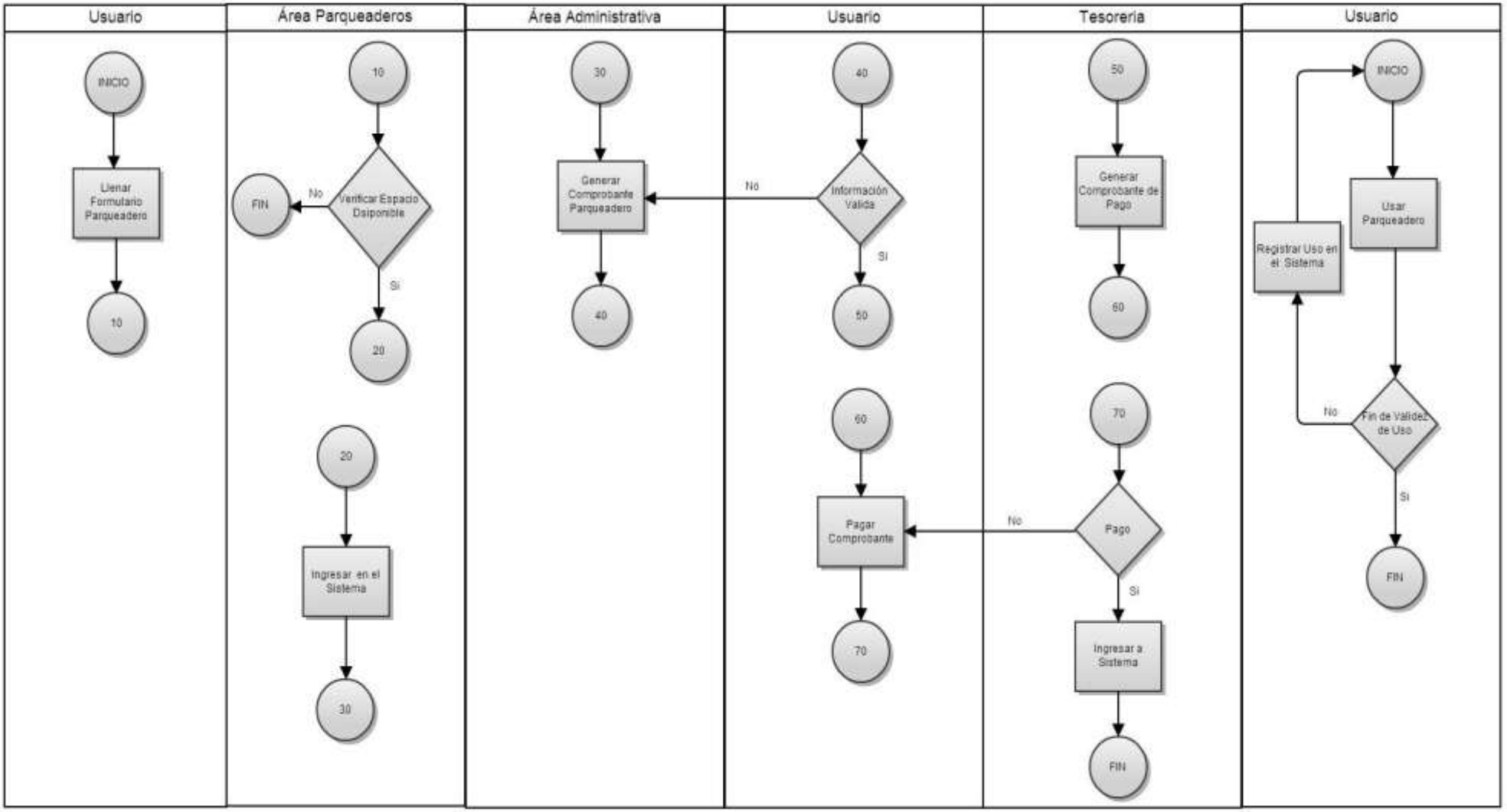


Diagrama 2.2 Modelo SIPOC Procesos Elaborado: Juan Carlos Andrade

2.1.1 Funcionales

- El Sistema administrará un registro de espacios disponibles dentro del parqueadero en base a esquemas del mismo.
- El Sistema permitirá registrar a todos los vehículos y asignarles un responsable para un fiel control del parqueadero.
- El Sistema permitirá llevar un completo registro de utilización en el que constarán datos como el vehículo, responsable, plaza de asignación, hora de entrada y hora de salida.
- El Sistema registrará el código de validación de ingreso por medio de un lector de código de barras.
- El Sistema permitirá un proceso de validación de acceso al parqueadero tomándose en cuenta el código de validación de ingreso, tipo de acceso del vehículo y disponibilidad de espacio.
- El Sistema imprimirá un ticket con la información del espacio asignado, la hora de entrada, el código del vehículo y un pequeño croquis del lugar a parquear, que servirá de respaldo al usuario en el caso que exista alguna confusión de otro usuario que haya parqueado donde no se le asigno.
- El Sistema brindará un mayor nivel de seguridad por medio un control minucioso de ingresos al parqueadero lo que permite a los guardias vigilar de mejor manera a los vehículos sin tener que preocuparse por la entrada de los mismos.

2.2.1 No Funcionales

Tecnología

Modelo	Magellan 800i	Magellan 1100i
	Lector ominidireccional de presentación con veloz e intuitiva lectura de códigos 1D y 2D, captura de imágenes y desactivación de tags EAS.	Un lector semifijo tipo imager rápido, agresivo, pequeño, ergonómico y con excelente relación costo/beneficio
Distancia máxima de lectura	15.2 cms. (UPC de 13 mils)	17.15 cms. (20 mils)
Codigos decodificables	Lineales, apilados y códigos 2D	Lineales. Lectura de códigos 2D de manera opcional
Otras características	Tecnología Illumix™ para lectura de códigos en pantallas de celulares y captura de imágenes. Desactivación de tags EAS Checkpoint e Interlock. Grado de protección industrial IP52.	Dimensiones de sólo 8.4 x 7.1 x 9.4 cms. Grado de protección industrial IP52.
Especificaciones	Ver Anexo 1.2	Ver Anexo 1.1

Licenciamiento

El sistema está elaborado bajo las siguientes especificaciones

Lenguaje: Java BDD: Oracle

Lo que implica que implicaremos un licenciamiento libre del tipo software de dominio público sometido a uso, copia, modificación o redistribución con o sin fines de lucro.

Requerimientos:

Hardware	Sistemas Operativos	Software
Computador • **Memoria RAM:** 256 MB • **Espacio en Disco:** 1 GB • **Procesador:** Núcleo Simple **Código lector de Barras** • Magellan™ 1100i	Windows XP Professional Windows Vista Professional Windows 7 Ultimate o Inferior Windows 8 Home Edition Windows Server 2000/2008	• Oracle 9 o Superior • Java JRE 5 o Superior

2.2 Diagramación de los procesos

Descripción:

La administración de parqueaderos involucra a varios actores, procesos y escenarios, los mismos que vendrán descritos dentro de las diferentes diagramaciones y explicaciones de actividades en este capítulo.

Actores:

- Usuarios
- Guardias
- Administrador

Procesos Involucrados:

Dentro de todo el manejo en el parqueadero se ha identificado los siguientes macro procesos y subprocesos

- Administrar Vehículos
  - Registrar Vehículo
  - Asociar Responsable
  - Asignar Código al Vehículo
  - Registrar Tipo de Accesos y Periodos
- Administrar Espacios
  - Esquematizar Parqueadero
  - Identificar Disponibilidad
- Registrar Utilización
  - Validar Ingreso
  - Asignar Plaza Libre
  - Registrar Ingreso
  - Registrar Salida

Cadena de Valor



Administrar Vehículo	Asignar Espacios	Registrar Utilización
1. Registrar Vehículo	1. Esquematizar Parqueadero	1. Validar Ingreso
2. Asociar Responsable	2. Identificar Disponibilidad	2. Asignar Plaza Libre
3. Asignar Código al Vehículo		3. Registrar Ingreso
4. Registrar Tipo de Acceso y Periodos		4. Registrar Salida

Procesos Centrales

- Administrar Vehículos
- Administrar Espacios
- Registrar Utilización

Procesos Soporte

- Registrar Vehículo
- Asociar Responsable
- Asignar Código al Vehículo
- Registrar Tipo de Accesos y Periodos
- Esquematizar Parqueadero
- Identificar Disponibilidad
- Validar Ingreso
- Asignar Plaza Libre
- Registrar Ingreso
- Registrar Salida

Mapa de Procesos

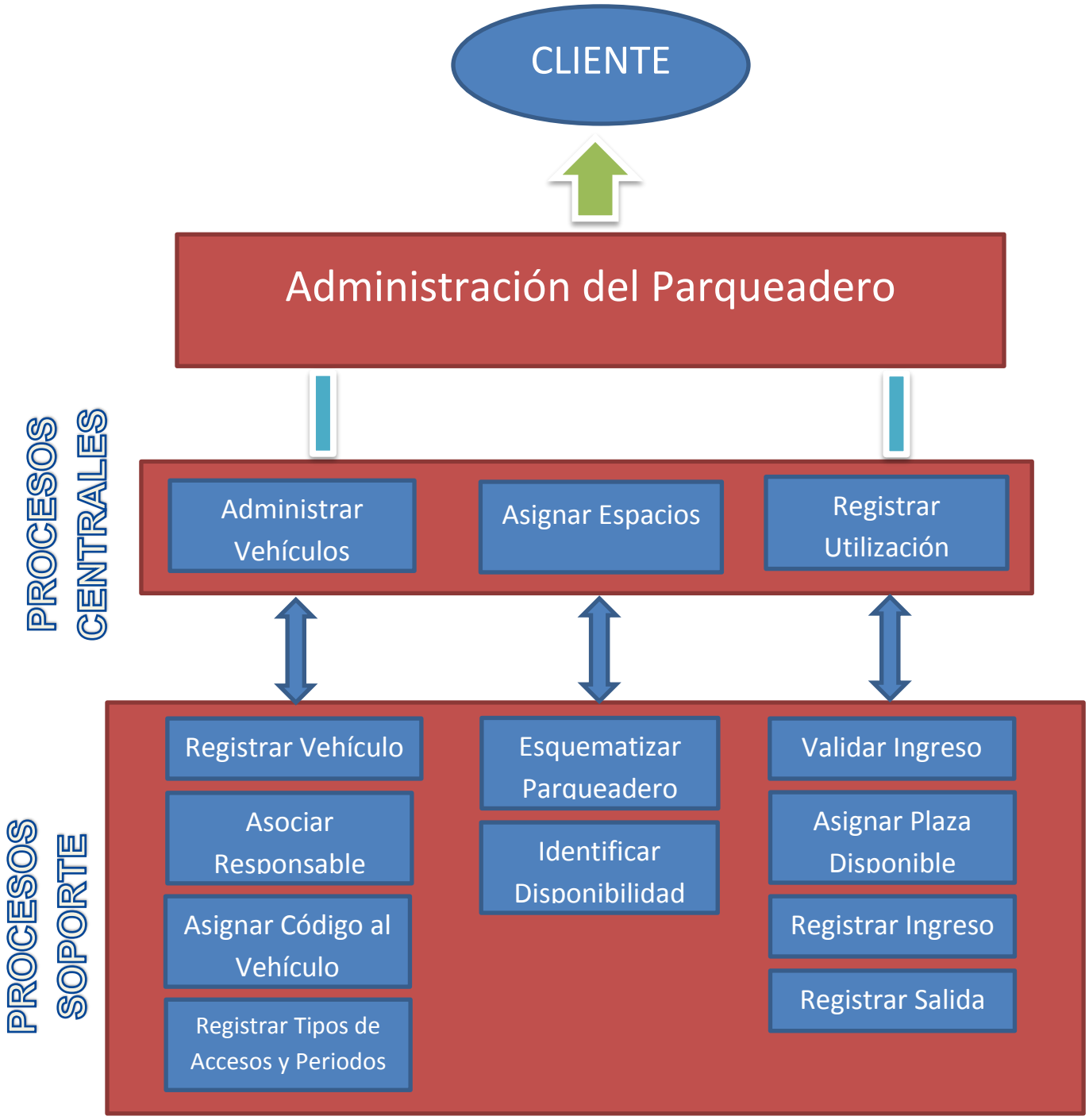
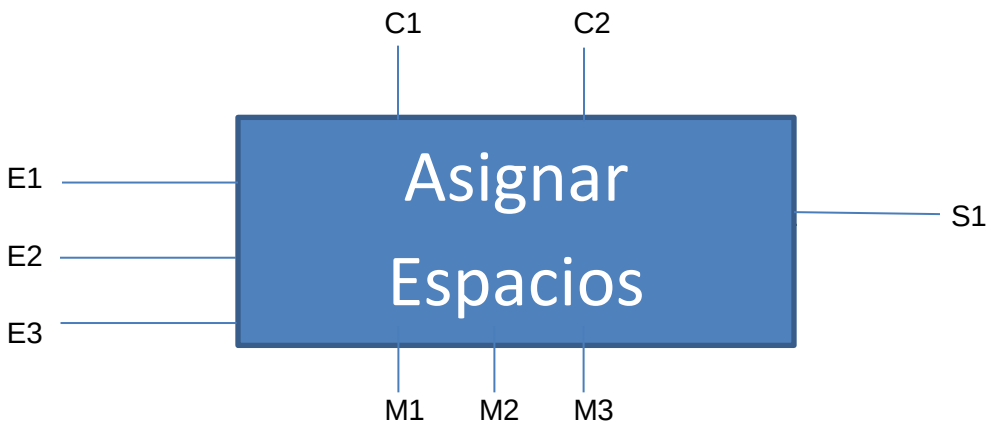


GRÁFICO PROCESO ASIGNAR ESPACIOS



Nombre Proceso: Asignación de Espacio

Propósito: Automatizar el proceso de asignación de plazas libres para los vehículos.

Actividades: Apartar un espacio libre

Unidad Organizacional: Parqueadero

E1: Código Vehículo (Código de Barras)

E2: Datos Usuario

E3: Datos Vehículo

C1: Políticas uso parqueaderos

C2: Normas para uso parqueaderos

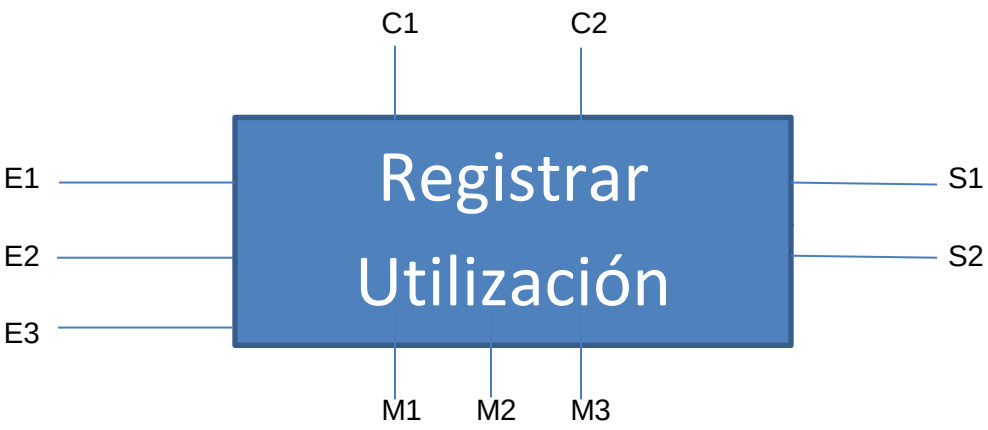
M1: Parqueaderos

M2: Tecnología

M3: Recursos Humanos

S1: Espacio Asignado

GRÁFICO PROCESO REGISTRAR UTILIZACIÓN



Nombre Proceso: Asignación de Espacio

Propósito: Automatizar el proceso de registro de uso del parqueadero.

Actividades: Registrar todos los datos pertinentes al uso del parqueadero.

Unidad Organizacional: Finanzas

E1: Datos Vehículo

E2: Datos Usuario

E3: Hora de Entrada

C1: Políticas uso parqueaderos

C2: Normas para uso parqueaderos

M1: Parqueaderos

M2: Tecnología

M3: Recursos Humanos

S1: Espacio Libre

S2: Hora de Salida

2.3 Priorización de los Procesos

En base a análisis de todos los procesos involucrados dentro de este proyecto se ha priorizado como más importantes a los siguientes procesos y subprocesos:

- **Administrar Vehículos**

Con los siguientes procesos asociados y de menor prioridad:

- Registrar Vehículo
- Asociar Responsable
- Asignar Código al Vehículo
- Registrar Tipo de Accesos y Periodos

- **Asignar Espacios**

Con los siguientes procesos asociados y de menor prioridad:

- Esquematizar Parqueadero
- Identificar Disponibilidad

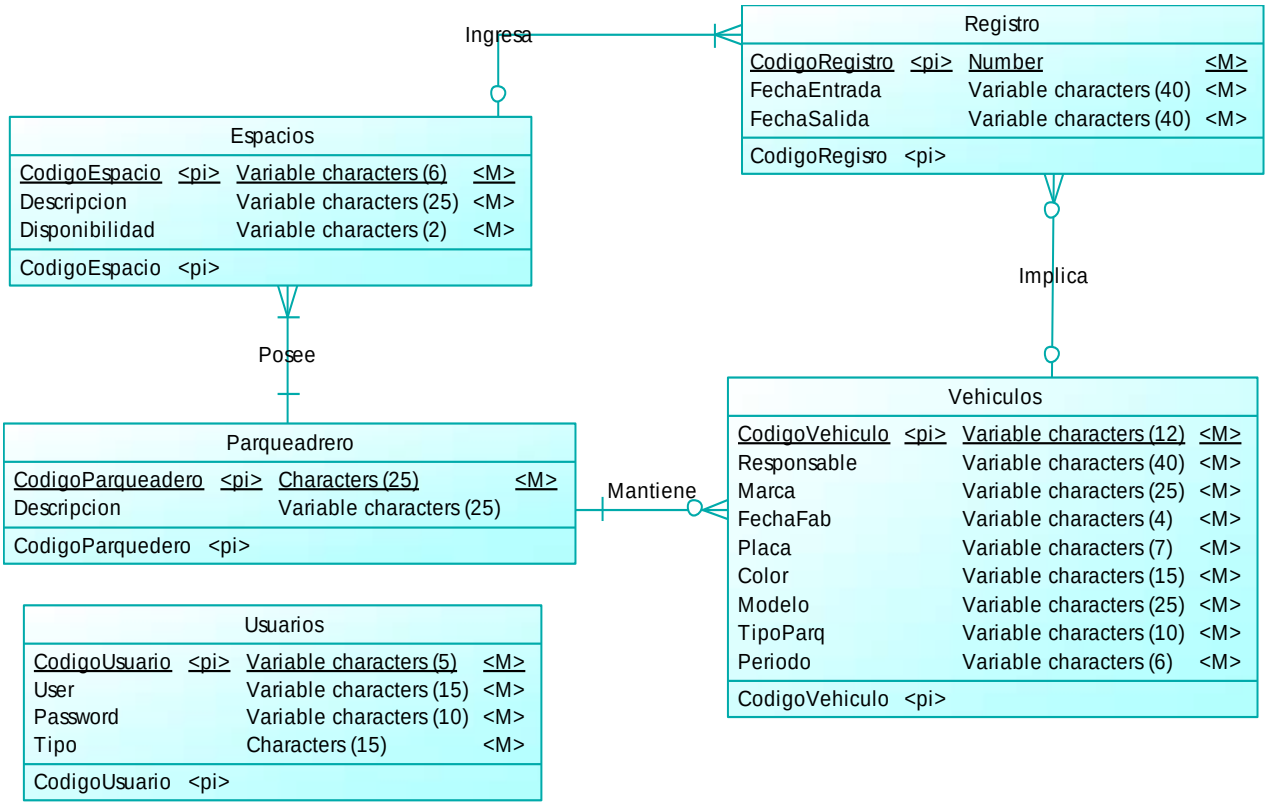
- **Registrar Utilización**

Con los siguientes procesos asociados y de menor prioridad:

- Validar Ingreso
- Asignar Plaza Libre
- Registrar Ingreso
- Registrar Salida

3. CAPITULO 3: Diseño-Modelamiento

3.2 Diseño Entidad/Relación (E/R)



Este diagrama nos permite visualizar como se diseñara la base de datos y todas sus entidades y relaciones involucradas dentro del sistema.

Casi todas las entidades tienen relación entre sí excepto la entidad usuarios la que se encargará solo de verificar las autentificaciones al sistema.

La tabla de usuarios podía desaparecer ya que con Oracle tenemos un módulo propio de administrador de usuarios en la cual se puede configurar roles y permisos, sin necesidad de una tabla.

3.2 Diagramas UML

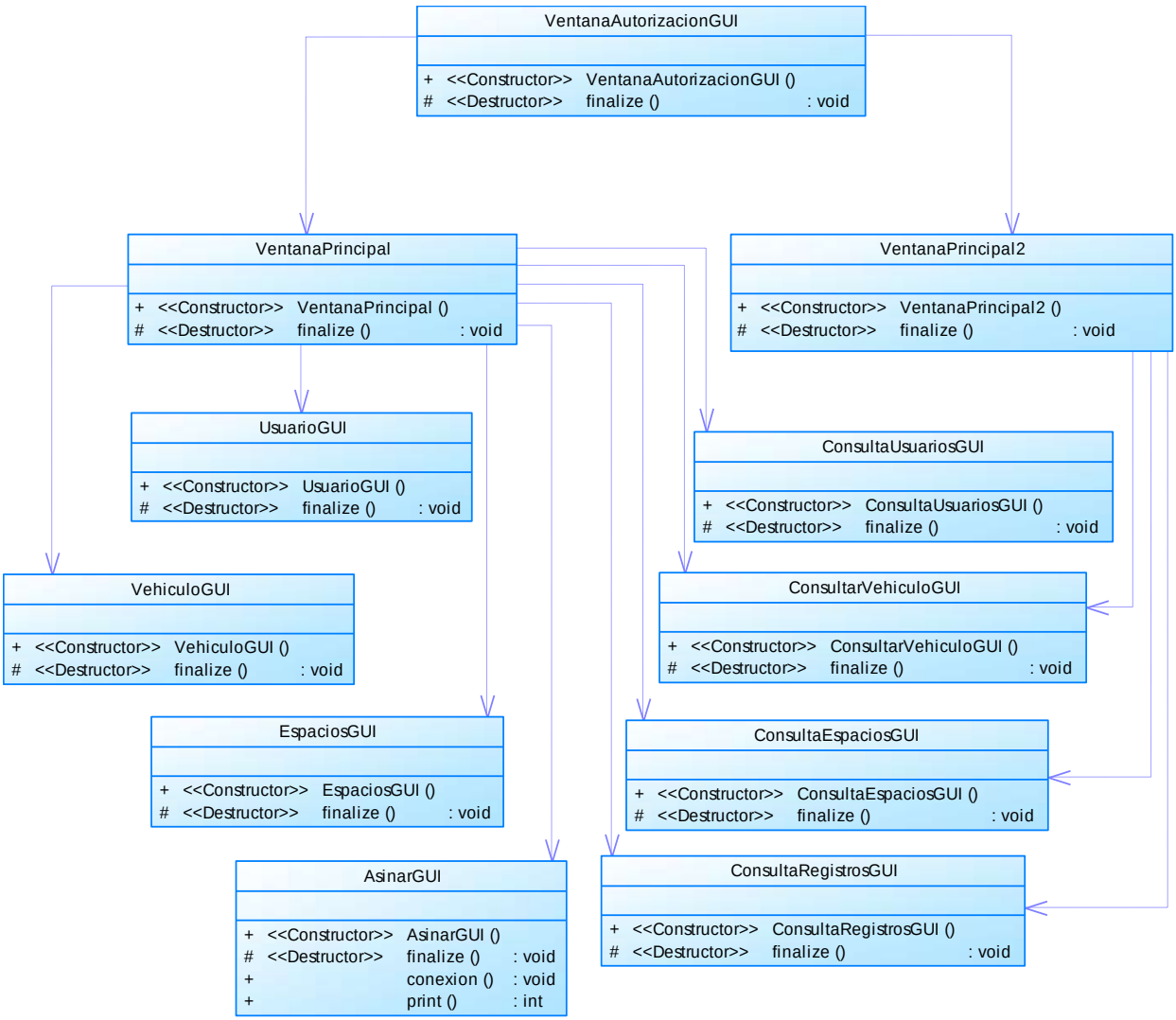
3.2.1 Diagrama de Clases

Diagrama de Clases DP



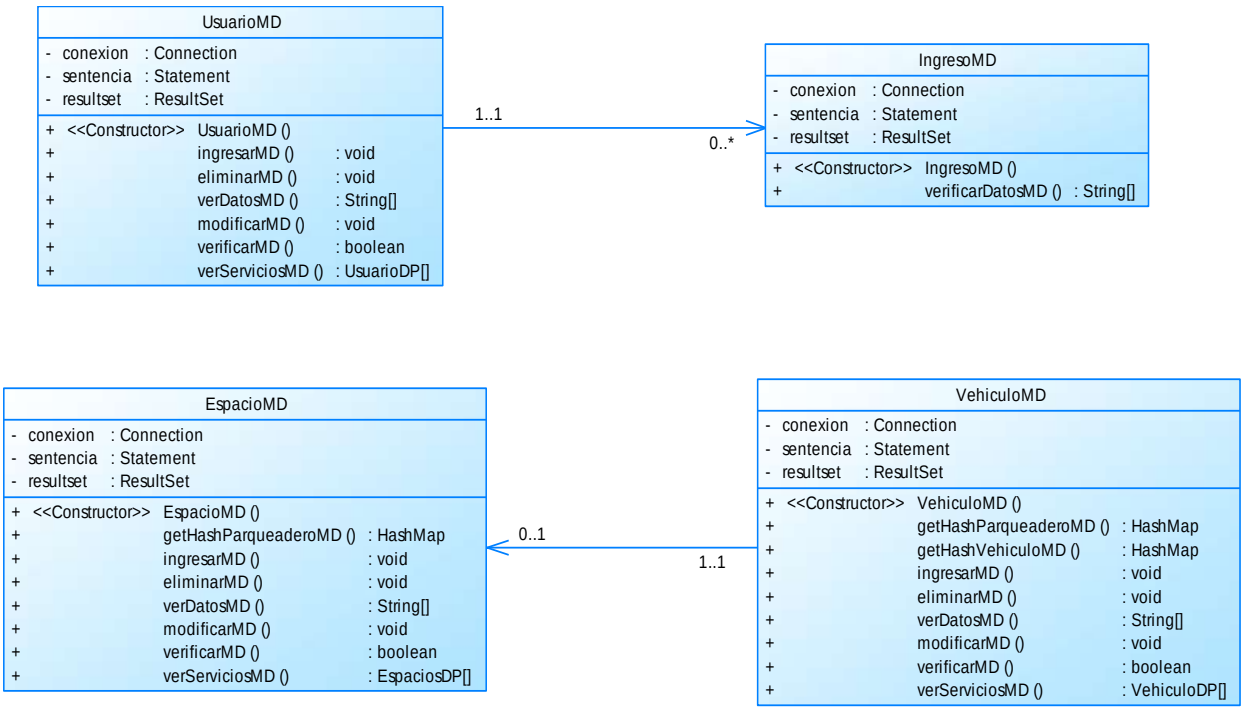
Este diagrama muestra todas las clases involucradas en el empaquetado DP (Data Process) así como sus relaciones, variables y métodos involucrados dentro de las mismas.

Diagrama de Clases GUI



Este diagrama muestra todas las clases involucradas en el empaquetado GUI (Graphical User Interface) así como el despliegue de estas interfaces dependientes de una interfaz superior.

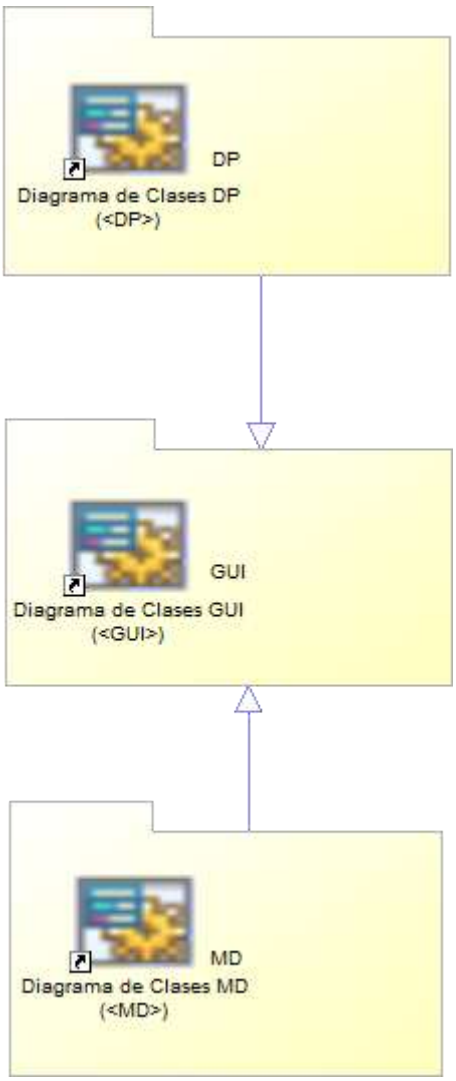
Diagrama de Clases MD



Este diagrama muestra todas las clases involucradas en el empaquetado MD (Management Data) así como sus relaciones, variables y métodos involucrados dentro de las mismas.

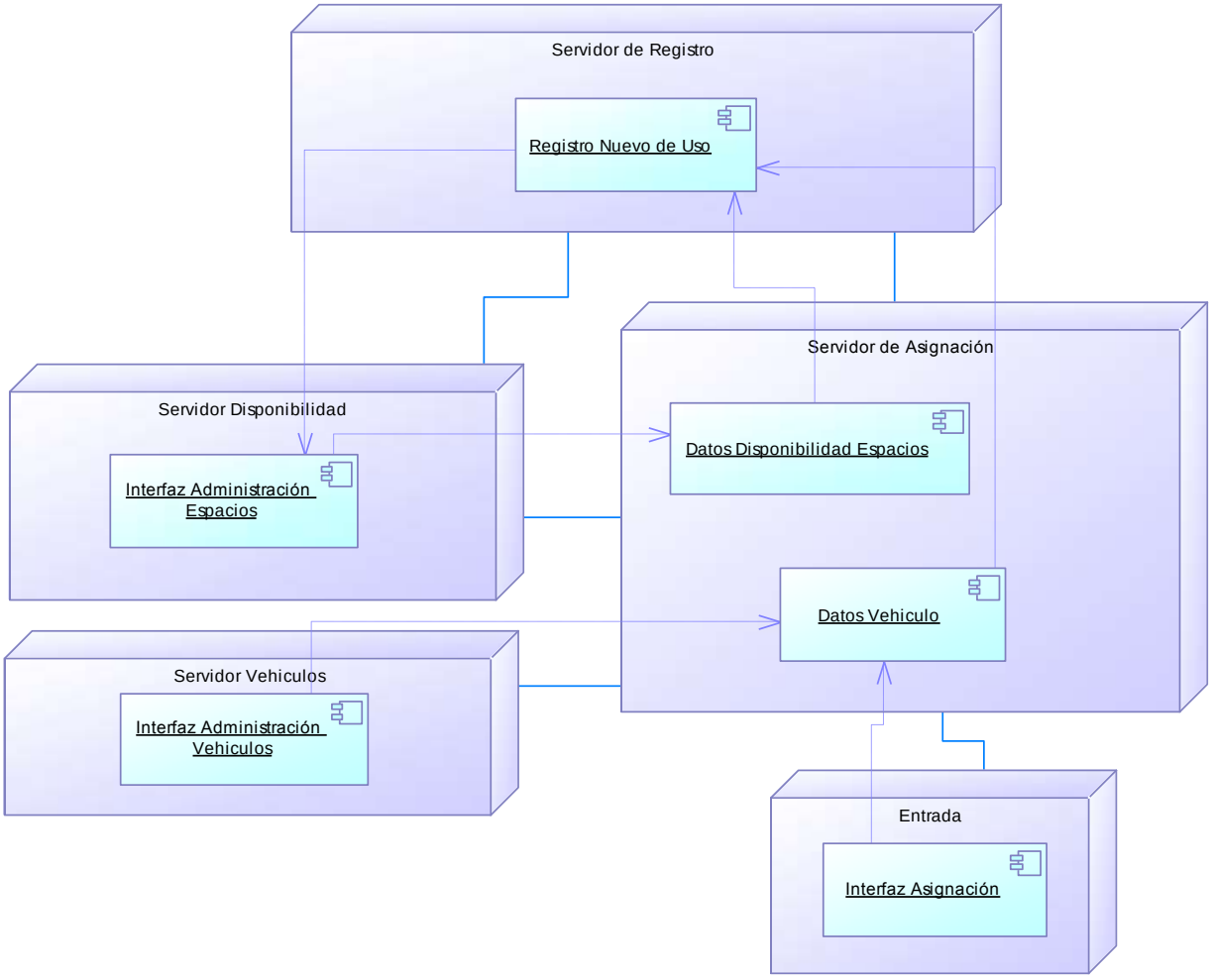
Esta clase es netamente dedicada a todo el manejo de la base de datos como las inserciones, modificaciones y eliminaciones.

Diagrama de Paquetes



Este diagrama muestra la relación entre clases y las posibles visualizaciones entre las mismas así como la interactividad de sus funciones dependientes del lazo de asociación.

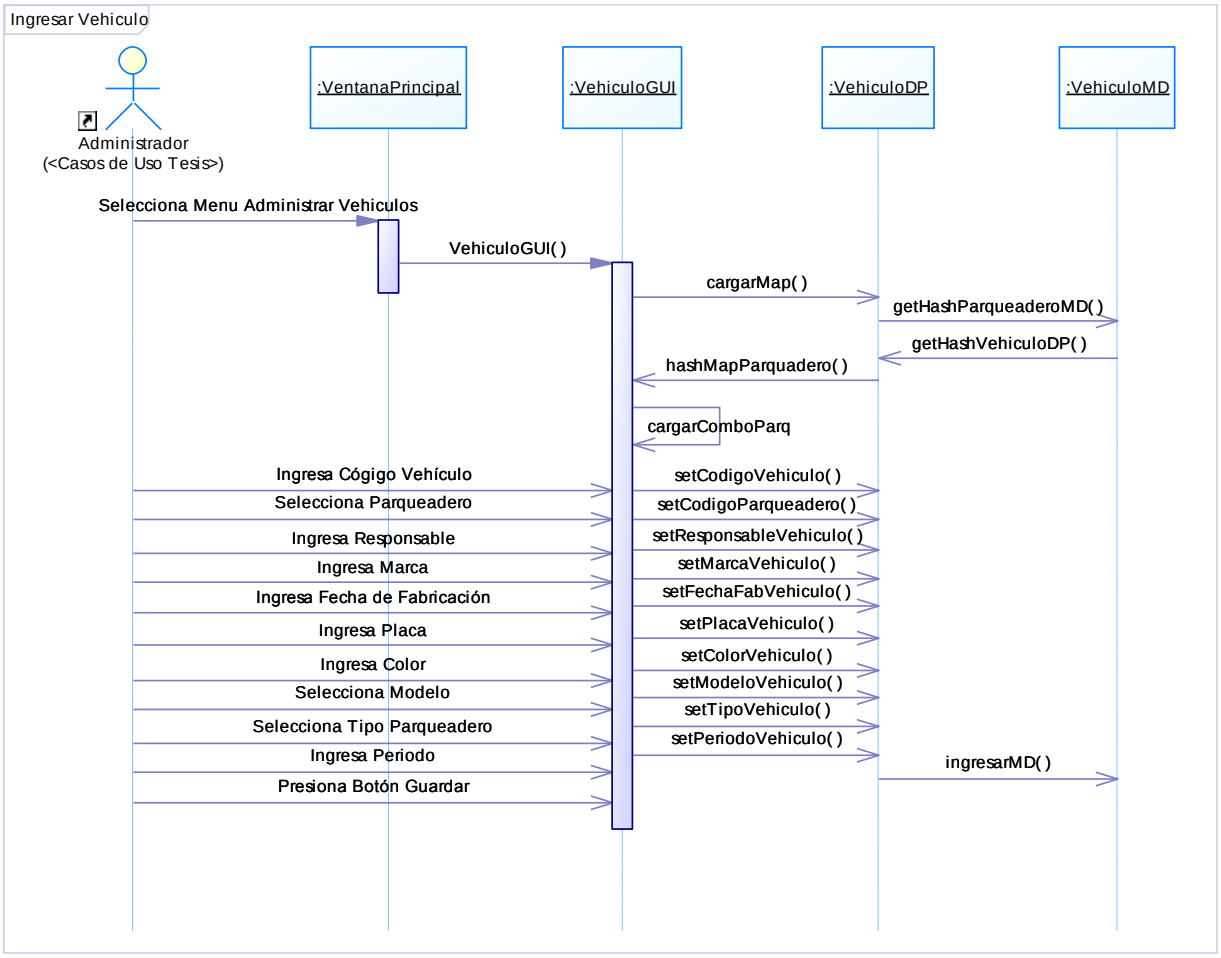
3.2.2 Diagrama de Despliegue



Este diagrama muestra todos los servidores involucrados en el proceso de asignación de espacios así como todos los datos involucrados dentro del mismo.

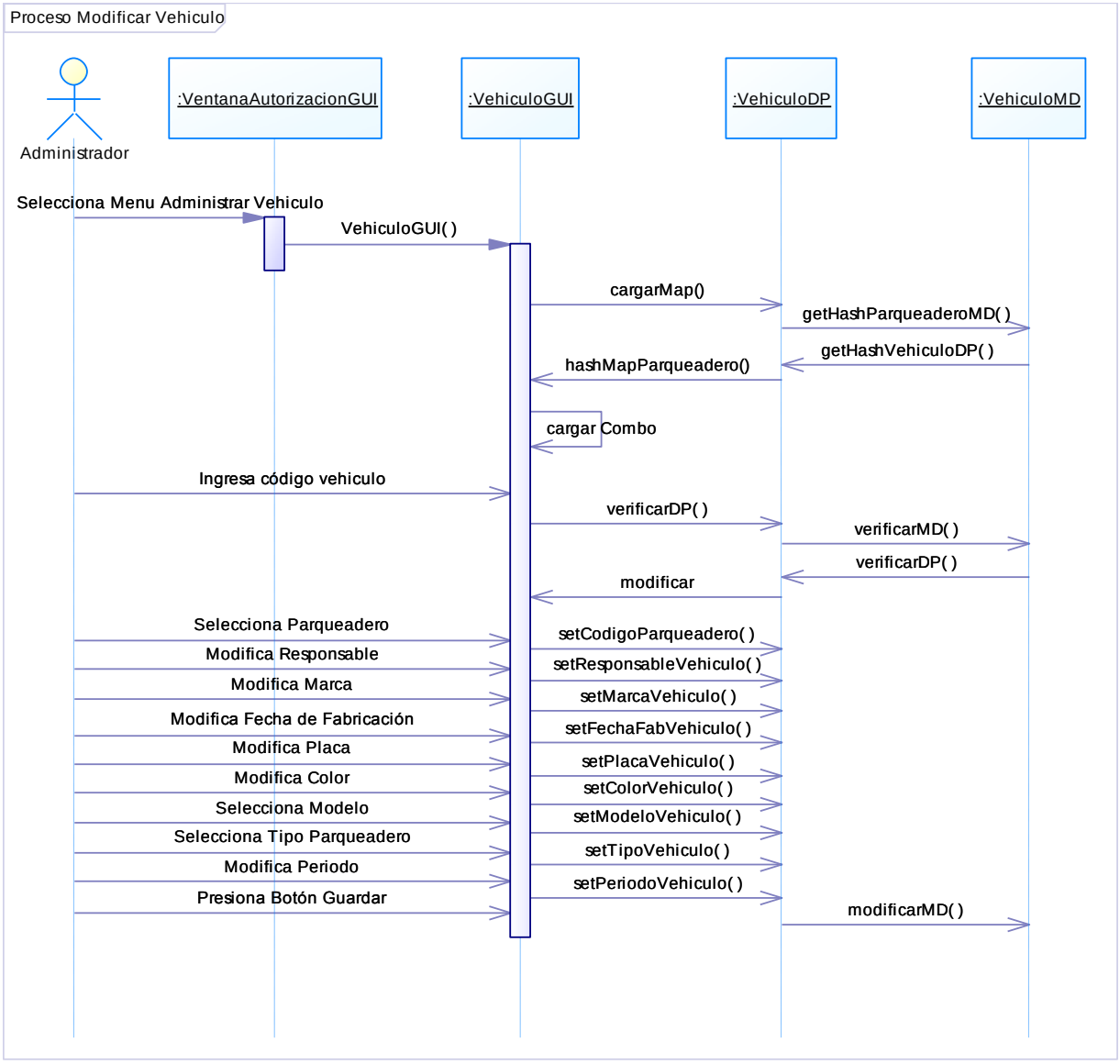
3.2.3 Diagramas de Secuencia

Proceso: Administración Vehículos (Ingresar Vehículo)



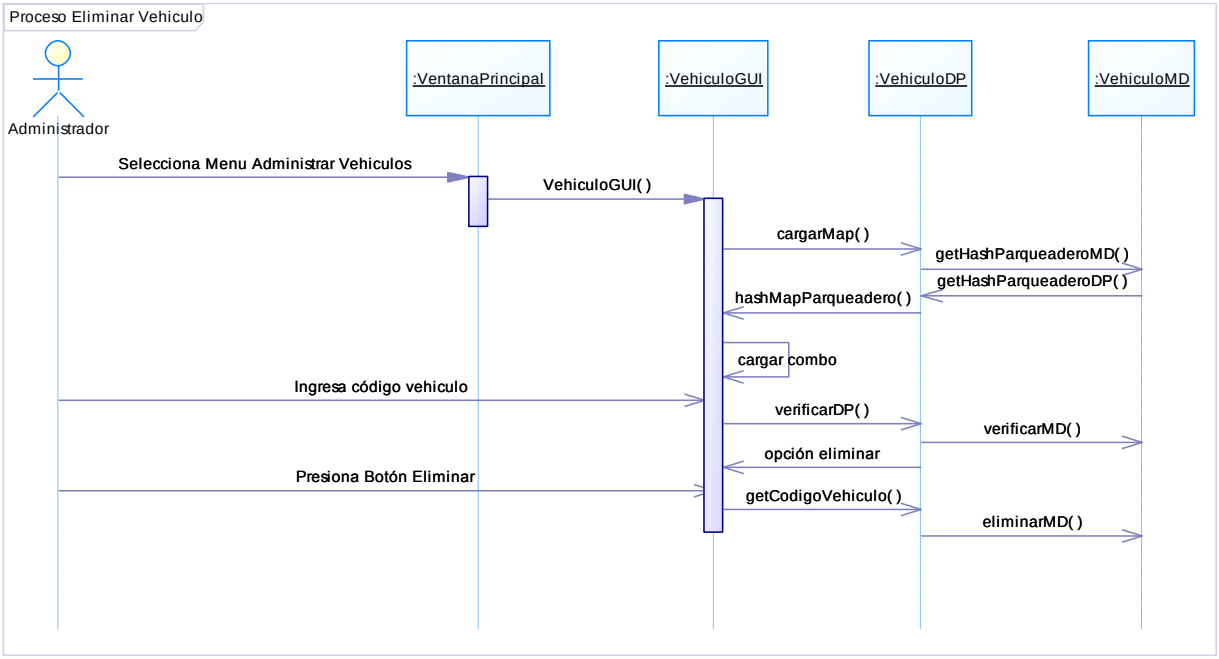
Este diagrama indica la secuencia iterativa entre el sistema y el administrador en el proceso de administración de vehículos específicamente el ingreso de un nuevo vehículo.

Proceso: Administración Vehículos (Modificar)



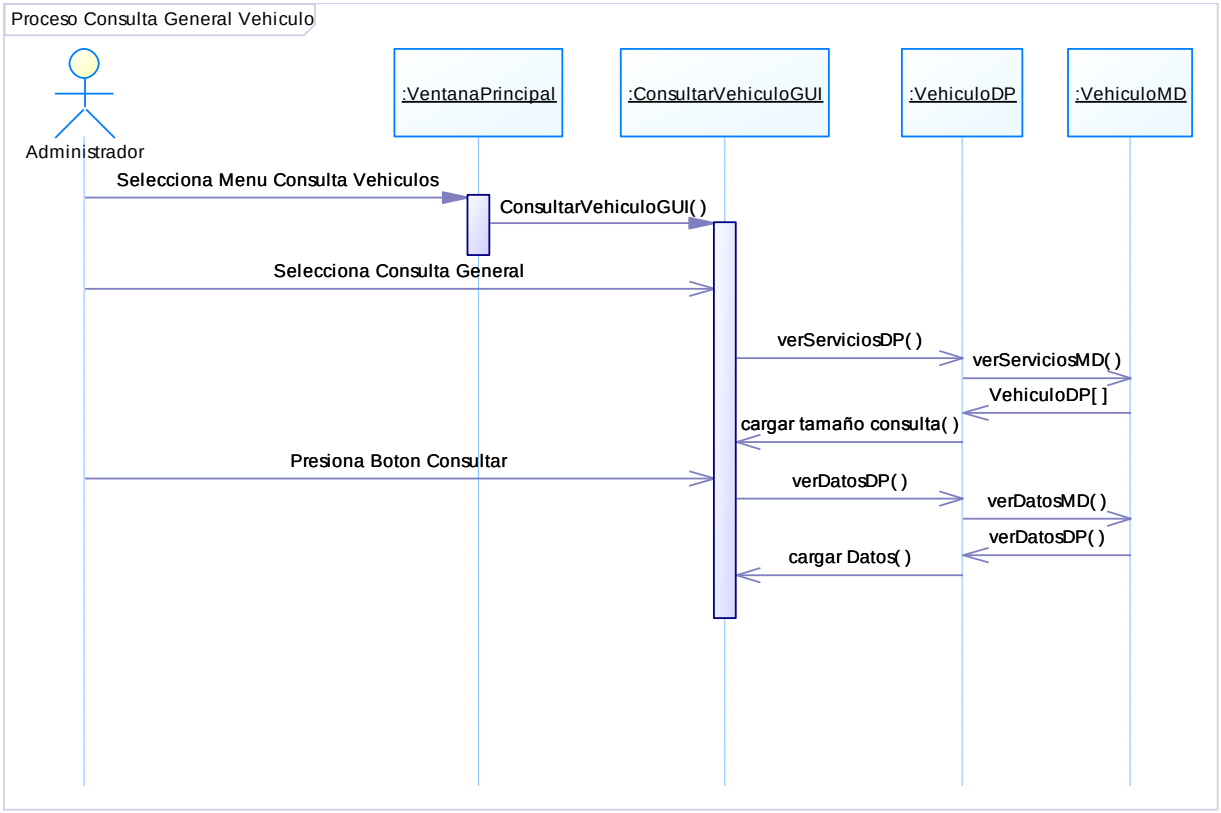
Este diagrama indica la secuencia iterativa entre el sistema y el administrador en el proceso de administración de vehículos específicamente la modificación de un vehículo.

Proceso: Administración Vehículos (Eliminar Vehículo)



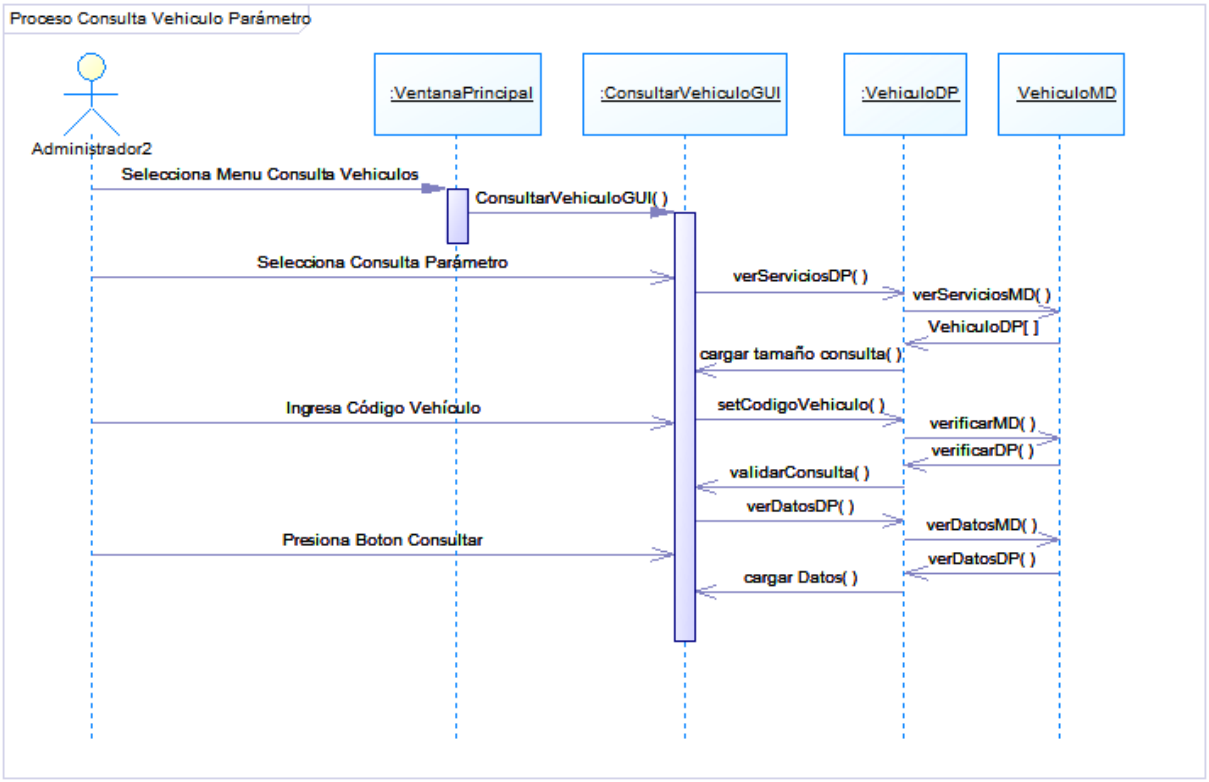
Este diagrama indica la secuencia iterativa entre el sistema y el administrador en el proceso de administración de vehículos específicamente la eliminación de un vehículo.

Proceso: Consultas Vehículos (Consulta General)



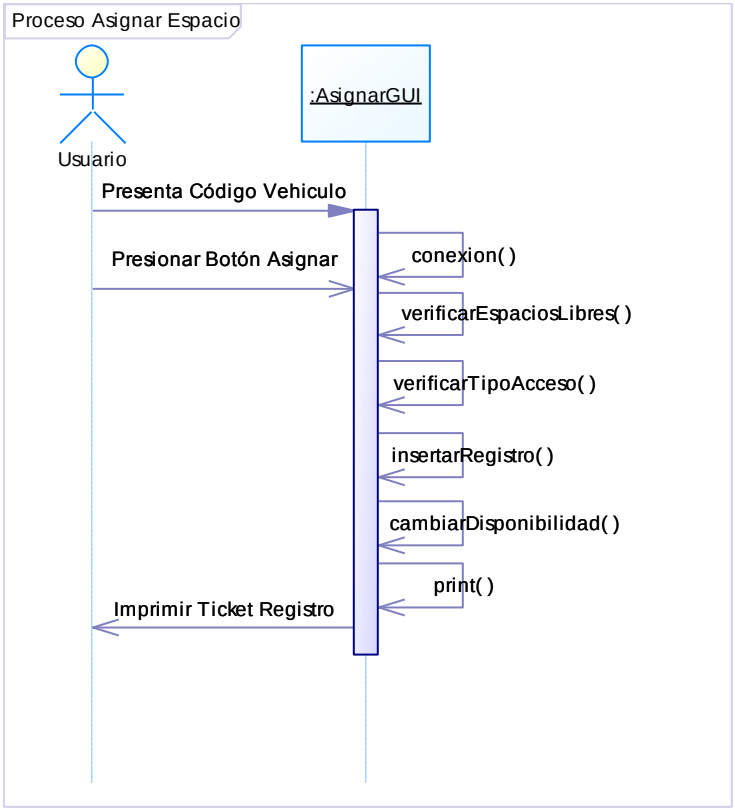
Este diagrama indica la secuencia iterativa entre el sistema y el administrador en el proceso de consulta de vehículos específicamente la consulta general de todos los datos involucrados en la clase vehículos.

Proceso: Consultas Vehículos (Consulta Parámetro)



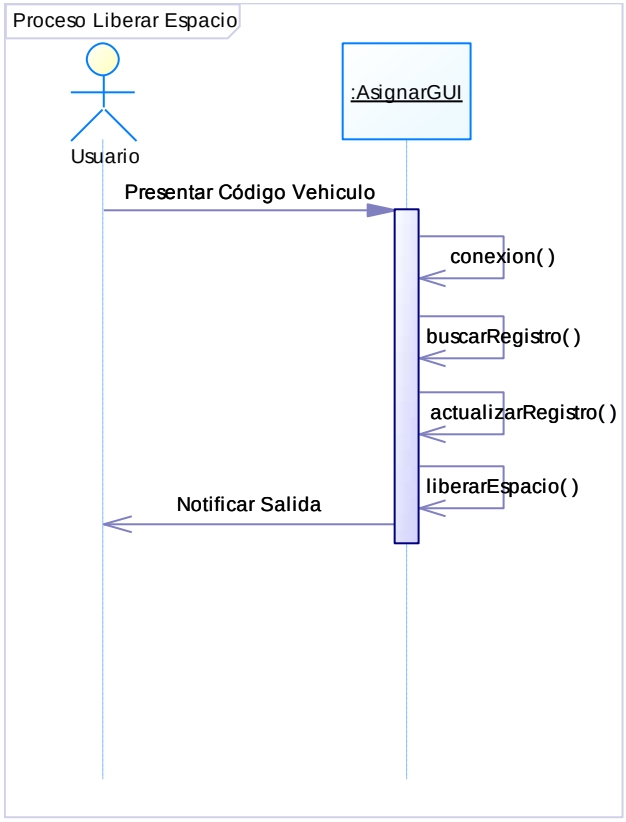
Este diagrama indica la secuencia iterativa entre el sistema y el administrador en el proceso de consulta de vehículos específicamente la consulta por parámetro que será establecida por el dato: código vehículo, el cual ayudará a verificar todos los datos involucrados con dicho código.

Proceso: Asignar (Asignar Espacio)



Este diagrama indica la secuencia iterativa entre el sistema y el administrador en el proceso de asignación de espacios el que involucra verificación de plazas disponibles y el registro de uso del parqueadero.

Proceso: Asignar (Liberar Espacio)



Este diagrama indica la secuencia iterativa entre el sistema y el administrador en el proceso de liberación de espacio el mismo que involucra procesos de actualización del registro de utilización y la liberación del espacio previamente asignado.

4. CAPITULO 4: Desarrollo y Pruebas

4.2 Diseño del Prototipo

En este capítulo visualizaremos el desarrollo total del sistema a través de imágenes actuales del prototipo.

Ventana de Autorización



Esta ventana se encarga del control de usuarios verificando sus identidades y privilegios ya pueden ser estos Administradores o Guardias.

Ventana Principal (Administrador)



Esta es la ventana principal de Administración la que permite el control total del sistema con opciones de:

- Administración:
  - Usuarios, Parqueaderos, Vehículos y Espacios
- Procesos:
  - Asignación y Reporte
- Consultas:
  - Usuario, Vehículo, Espacios y Registros

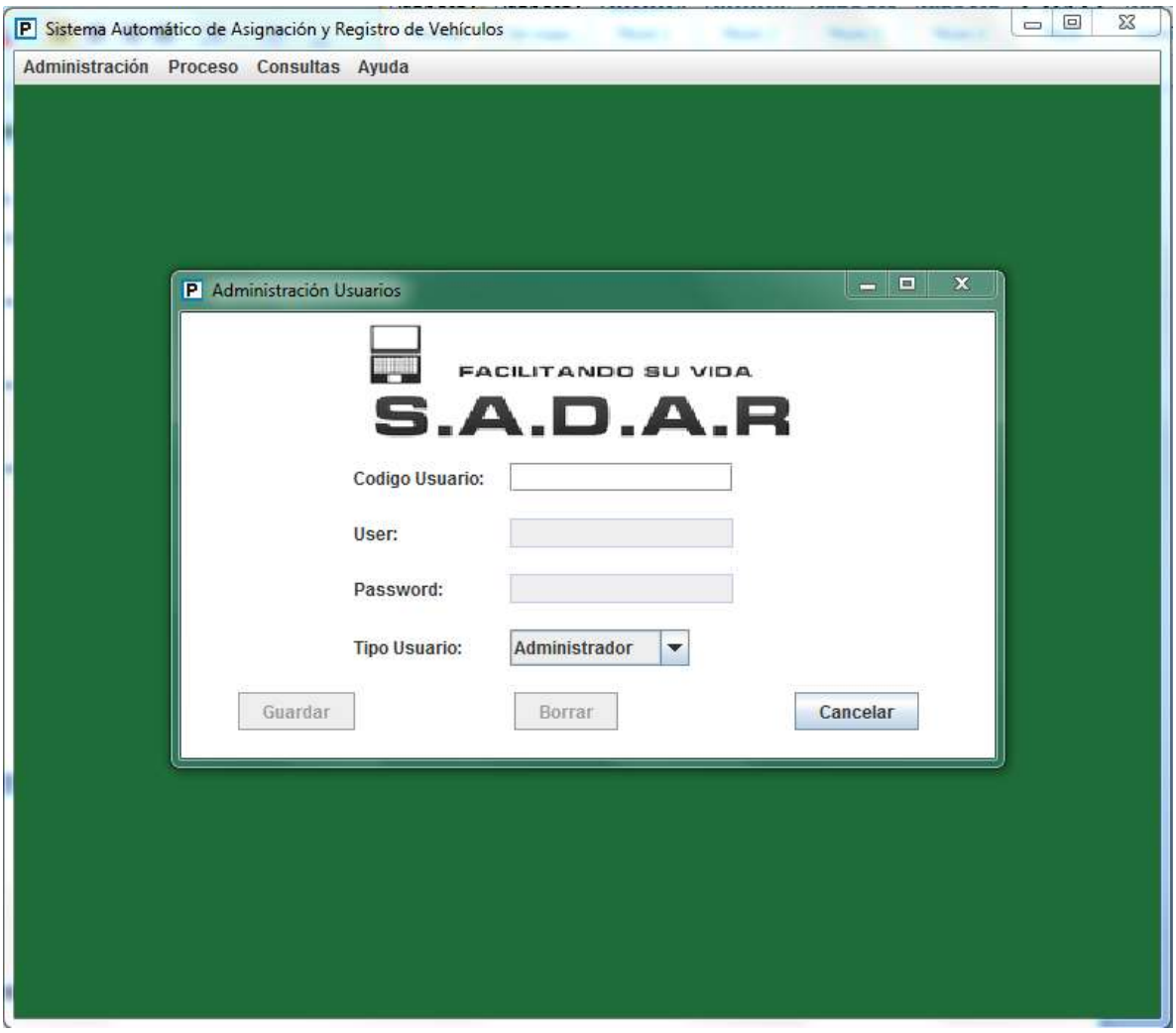
Ventana Principal (Guardia)



Esta es la ventana principal de Administración la que permite el control total del sistema con opciones de:

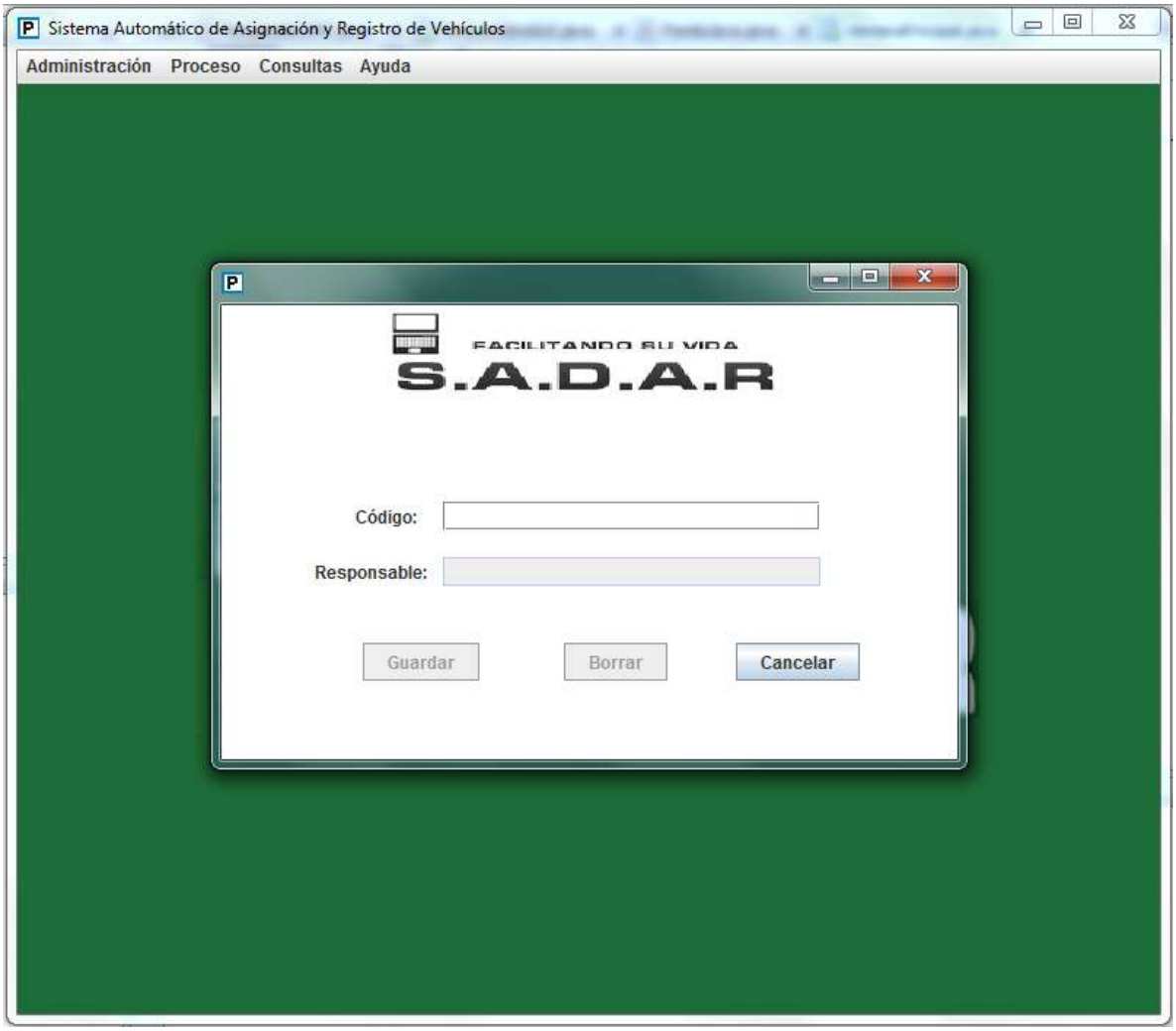
- Consultas:
    - Vehículo, Espacios y Registros

Ventana Administración Usuarios



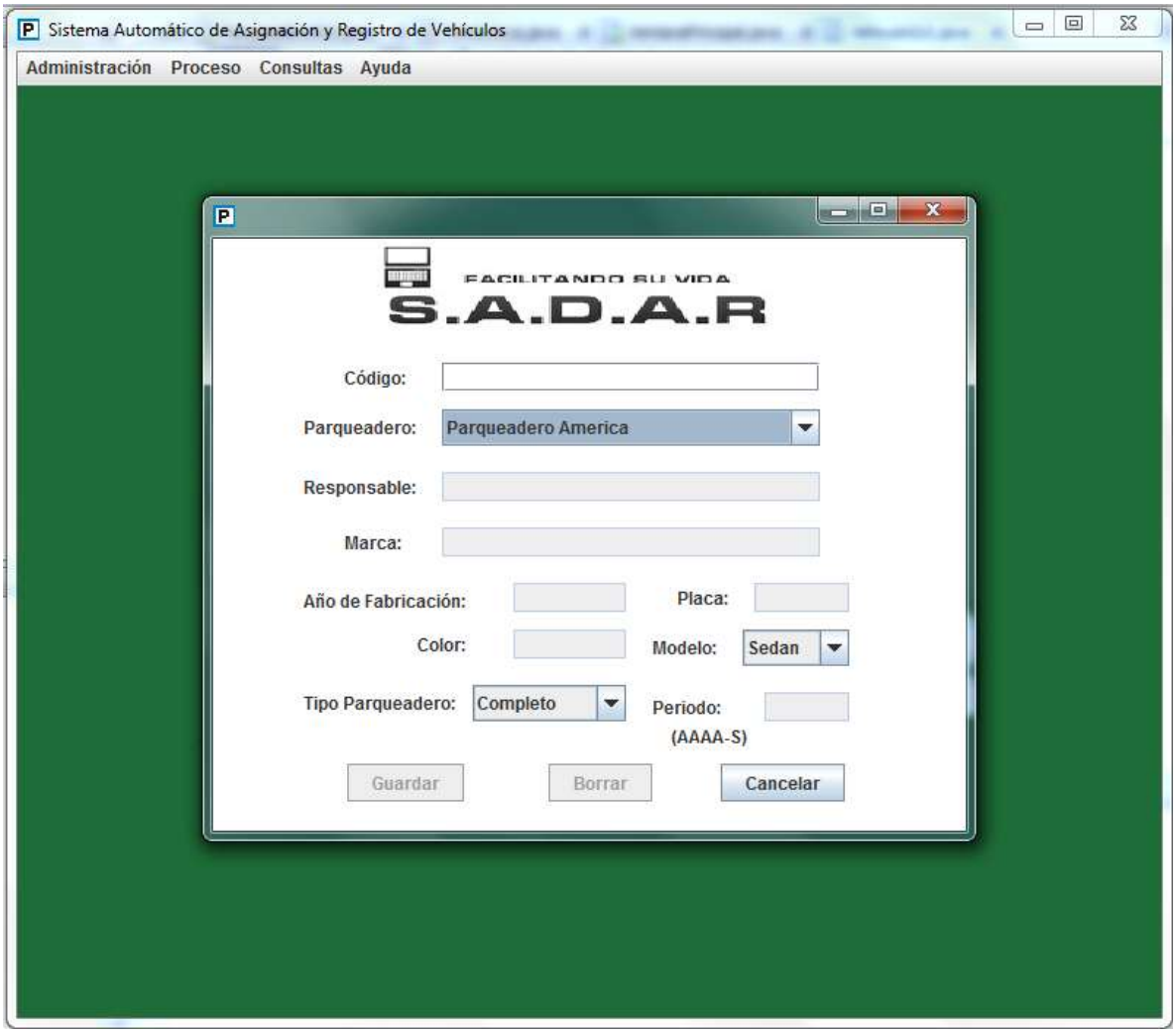
Esta ventana permite el ingreso, modificación y eliminación de usuarios que harán uso del sistema, este es el único método de agregación de usuarios para el uso del sistema por lo tanto el único en privilegios de creación de usuarios es el Administrador del sistema.

Ventana de Administración



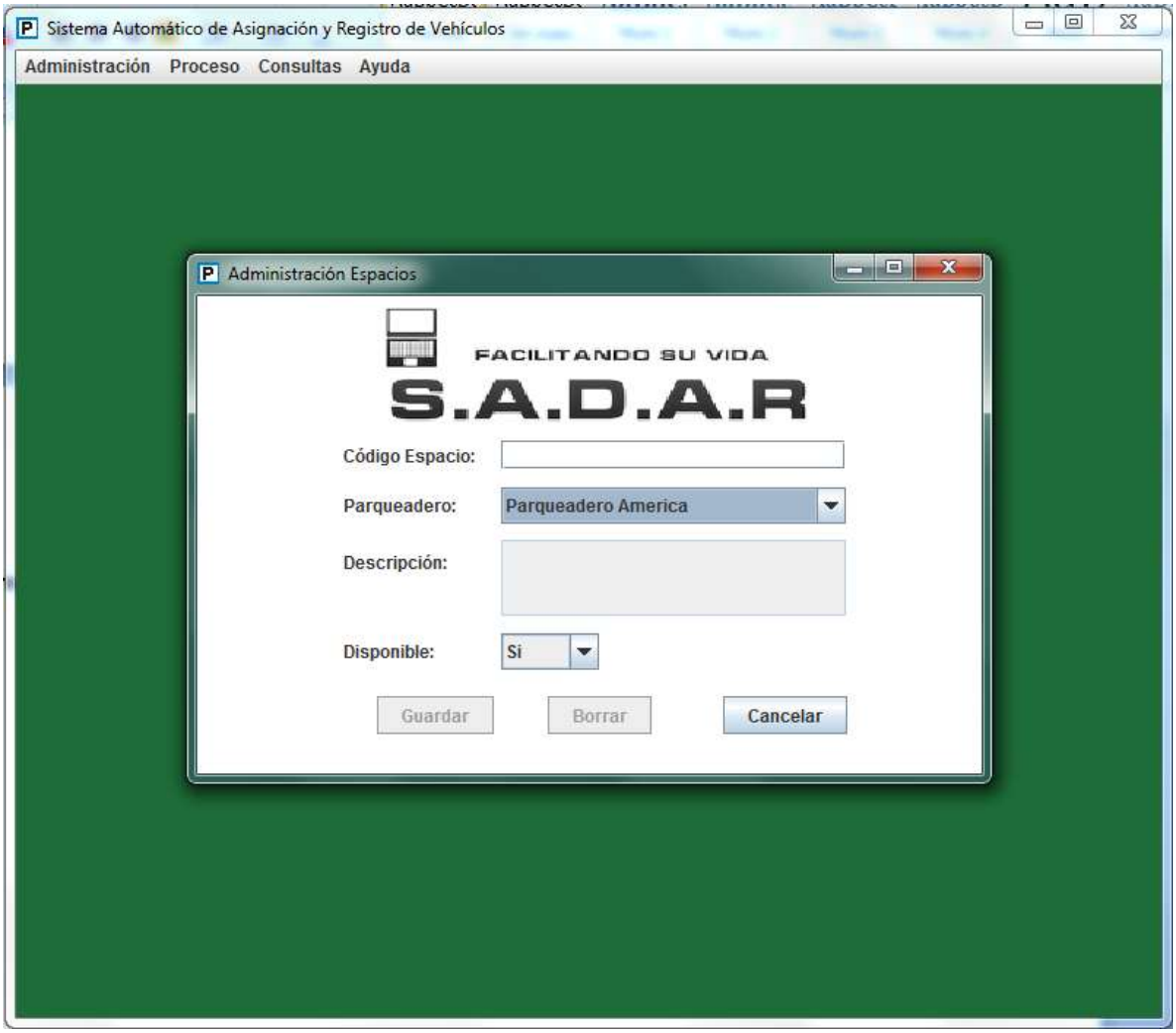
Esta ventana permite el ingreso, modificación y eliminación de parqueaderos ya que el sistema permite el manejo múltiple de varias sucursales de parqueaderos asociados con sus espacios respectivamente.

Administración de Vehículos



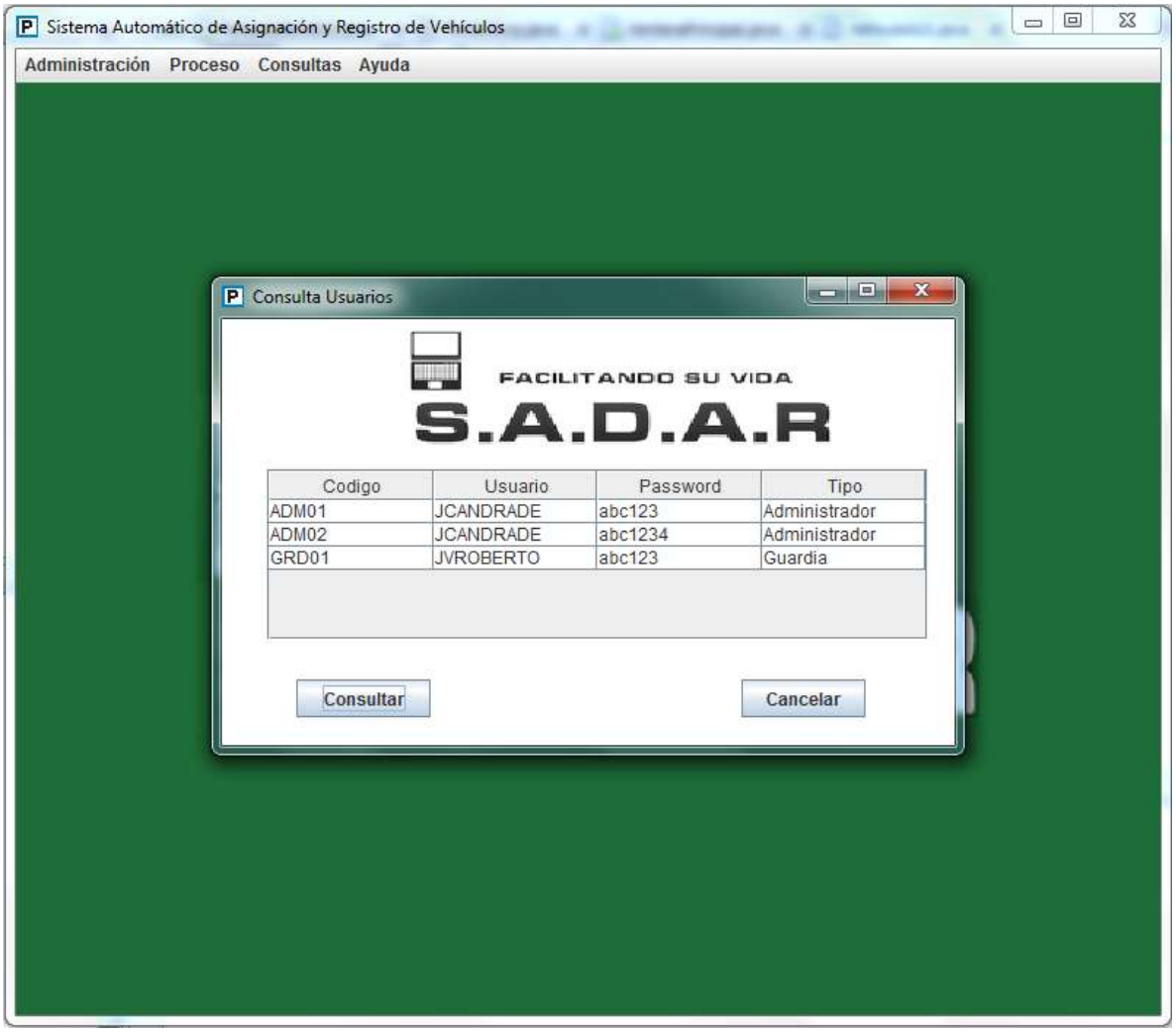
Esta ventana permite el ingreso, modificación y eliminación de vehículos en el sistema los cuales estarán sujetos a reglas y normas del parqueadero como tipo de parqueadero y periodo de uso.

Administración de Espacios



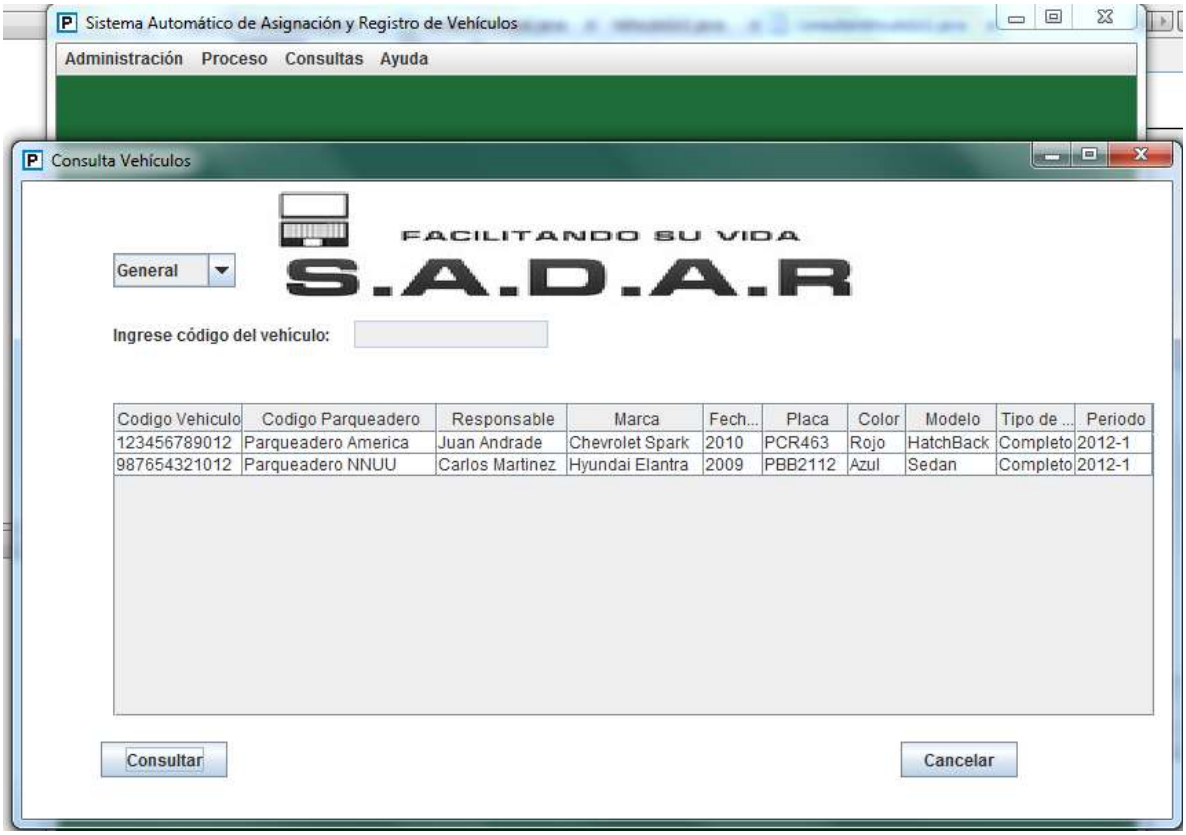
Esta ventana permite el ingreso, modificación y eliminación de espacios dentro de los diferentes parqueaderos, se puede manejar también la disponibilidad de los espacios dentro de este módulo en el caso de agregar nuevos lugares.

Consulta de Usuarios



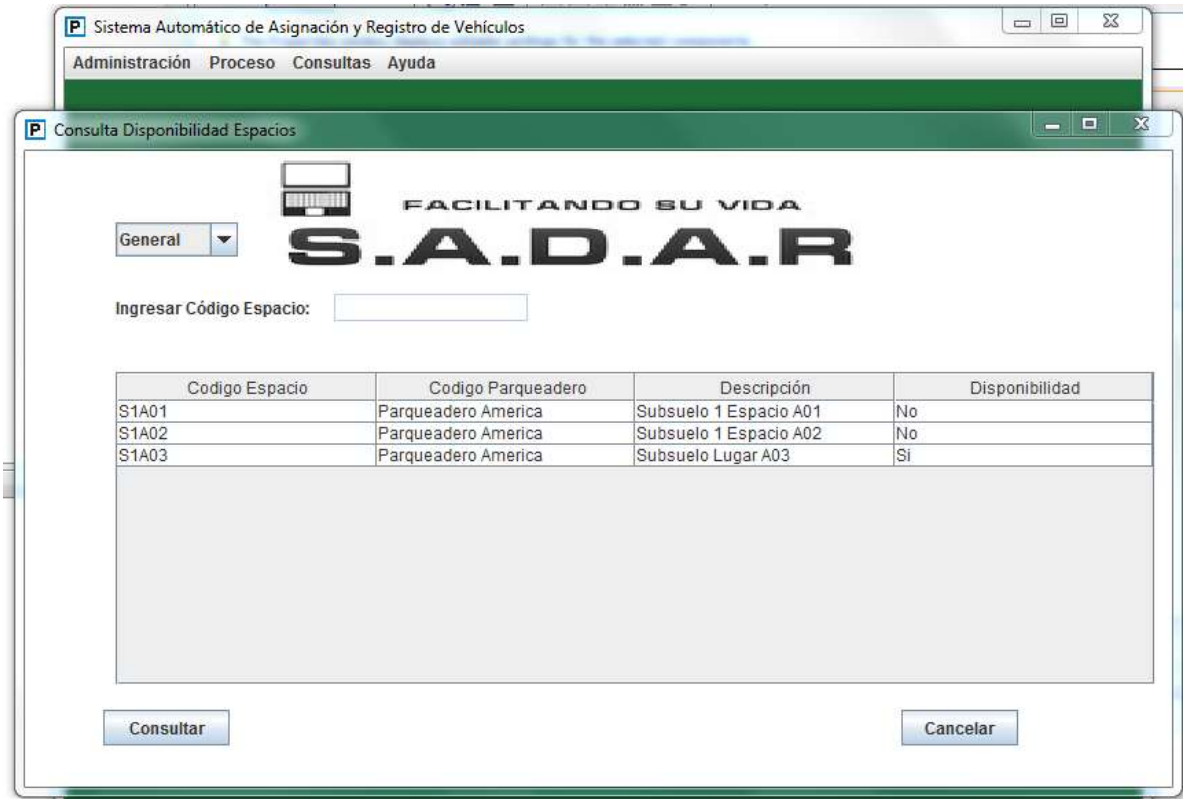
Esta ventana permite hacer consultas generales de los usuarios dentro de la base así como la identificación de sus claves y tipos de accesos.

Consultas de Vehículos



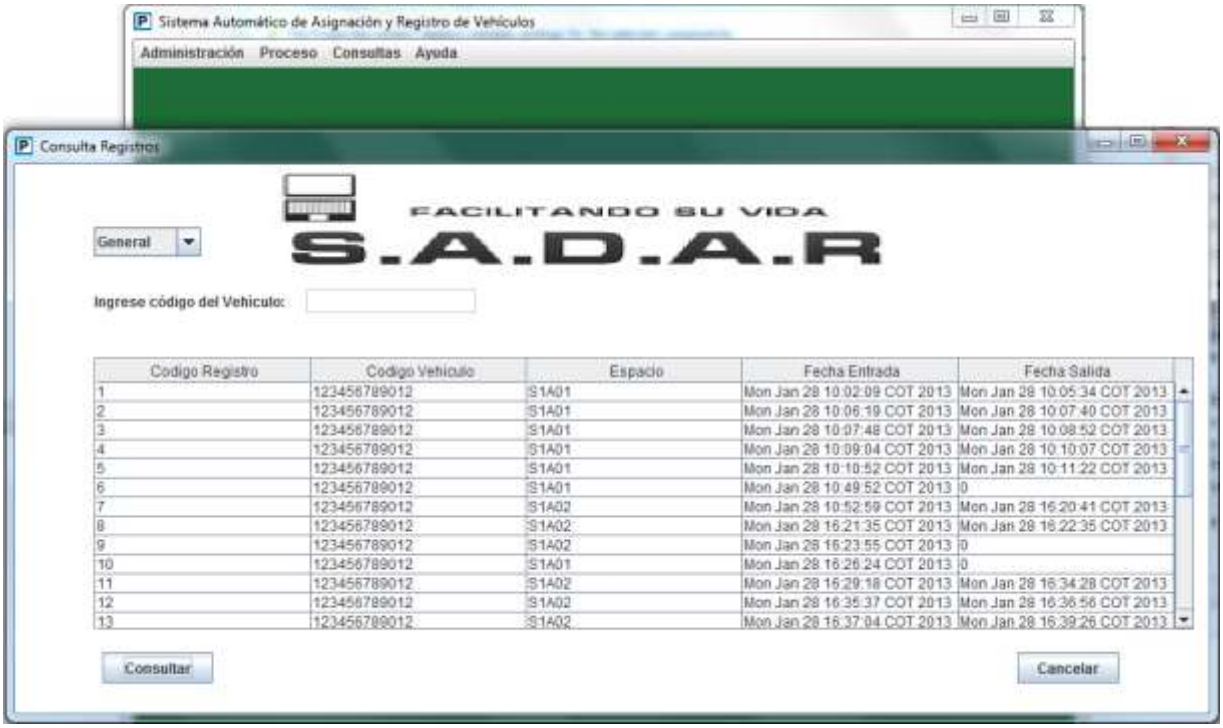
Esta ventana permite hacer consultas generales y por parámetro la cual hace referencia al código identificativo del vehículo y donde se desplegará todos los datos referentes a dicho código.

Consulta de Espacios



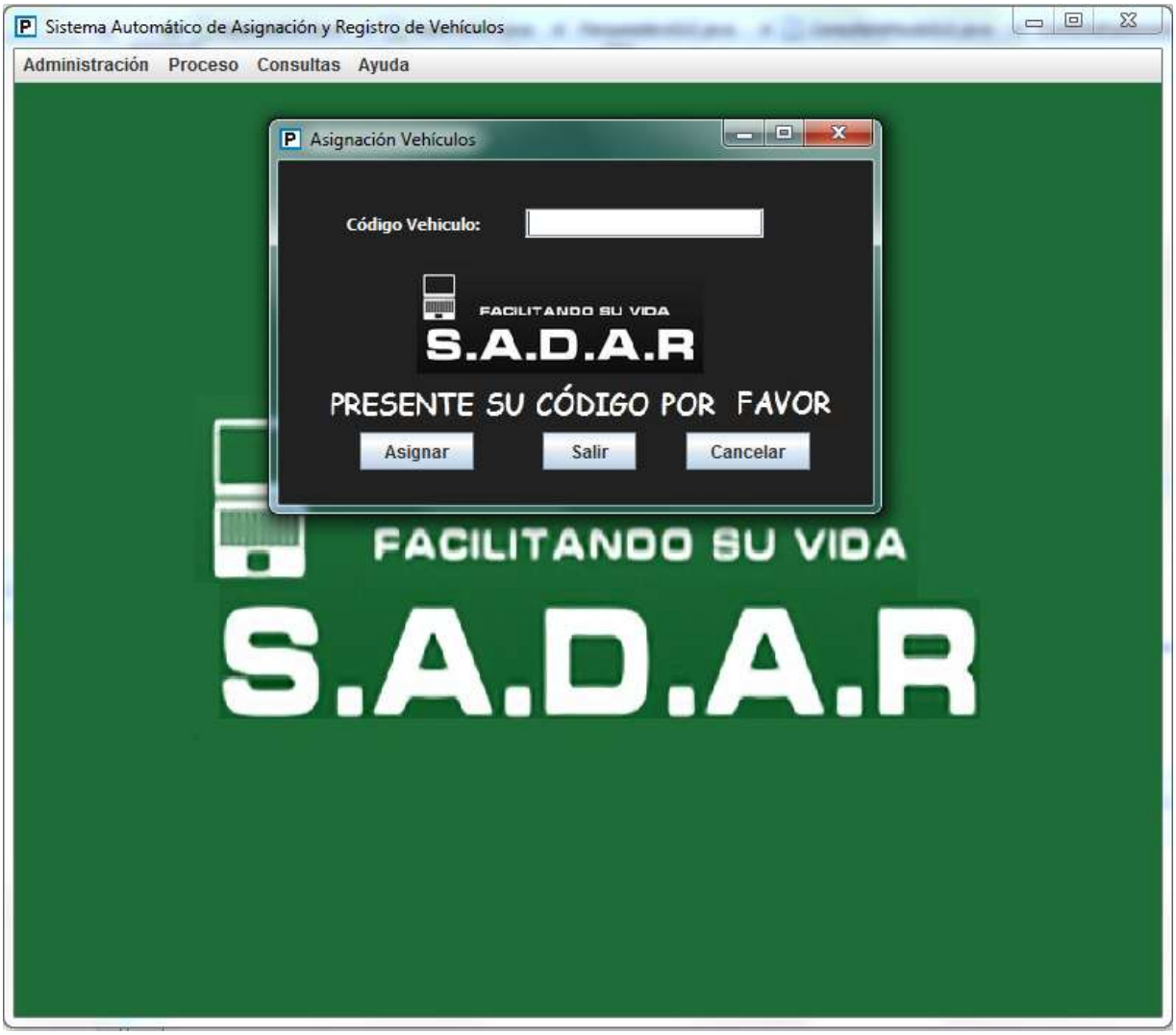
Esta ventana permite hacer consultas generales y por parámetro la cual hace referencia al código identificativo del espacio y donde se desplegará todos los datos referentes a dicho código.

Consulta de Registros



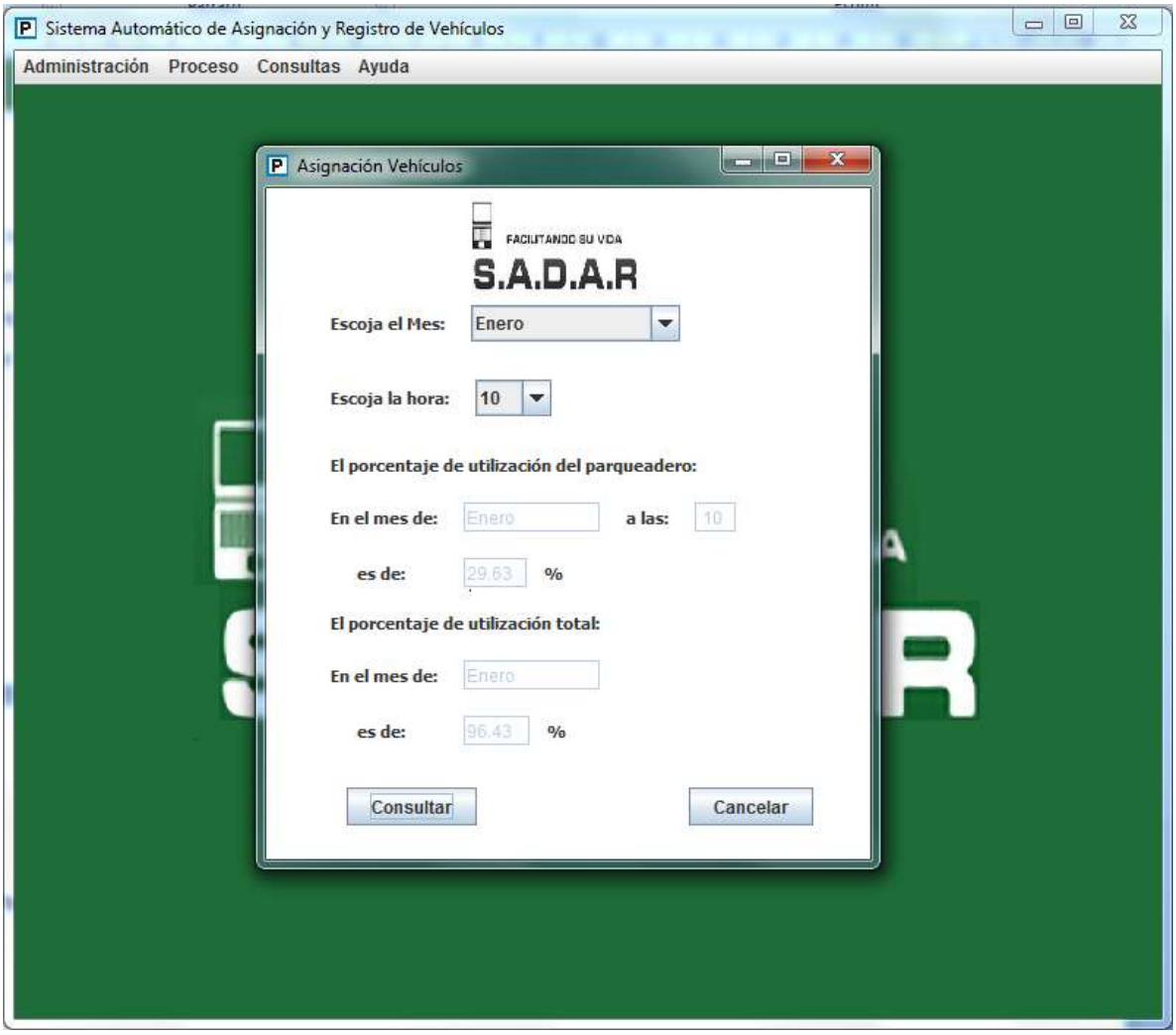
Esta ventana permite hacer consultas generales y por parámetro la cual hace referencia al código identificativo del vehículo y donde se desplegará todos los datos de registros referentes a dicho código.

Proceso de Asignación de Plazas Libres



Esta ventana involucra al proceso de asignación de plazas libres en el cual por medio de la presentación del código del vehículo el usuario formará parte del proceso de asignación de espacios libres dentro del estacionamiento, una vez realizado este proceso simultáneamente se registrará el uso del espacio conjunto a otros datos de igual relevancia.

Proceso de Elaboración de Reportes



Esta ventana permitirá la elaboración de consulta en base a reportes de mes y hora elegidos por el usuario, calculará el porcentaje de utilización del parqueadero a dicho mes y hora así como el porcentaje de utilización total para dicho mes.

4.3 Pruebas de Ingresos

Para las pruebas se ha optado por el uso de un cuadro que describe entradas esperadas, rangos de valores no válidos y salidas esperadas. Así como pantallas que verifican el total funcionamiento del sistema.

Ingreso Usuarios



EDITAR	CODIGOUSUARIO	USER	PASSWORD	TIPO
	ADM01	JCANDRADE	abc123	Administrador
	ADM02	JCANDRADE	abc1234	Administrador
	GRD02	Jose Morales	jmorales	Guardia
	GRD01	JVROBERTO	abc123	Guardia

fila(s) 1 - 4 de 4

Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Código Usuario	Identificador del usuario: 3 Letras; 2 Números Identificador Tipo (3 Letras) Numerador (2 Números)	Identificador del usuario: + 3 Letras; + 2 Números	Registro almacenado en base de datos Campo VARCHAR2(5)
User	Nombre del Usuario: 15 Letras o Números	Nombre del Usuario: + 15 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(15)
Password	Clave de Ingreso: 10 Letras o Números	Clave Ingreso: + 10 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(10)
Tipo	Tipo Acceso: 15 Letras	Tipo Acceso: Sin errores truncado con Combo Box	Registro almacenado en base de datos Campo CHAR(15)

Ingreso Parqueadero

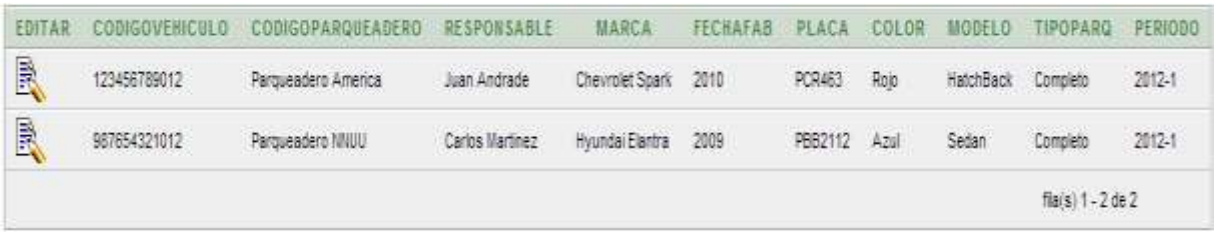


EDITAR	CODIGOPARQUEADERO	DESCRIPCION
	Parqueadero NNUU	Parq NNUU y Amazonas
	Parqueadero Colon	Parq Colon y 6 Dic
	Parqueadero America	Parq America y NNUU
		fila(s) 1 - 3 de 3

Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Código Parqueadero	Identificador parqueadero: 25 Letras	Identificador parqueadero: + 25 Letras	Registro almacenado en base de datos Campo CHAR(25)
Descripción	Ubicación Parqueadero: 25 Letras o Números	Nombre del Usuario: + 25 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(25)

Ingreso Vehículo





EDITAR	CODIGOVEHICULO	CODIGOPARQUEADERO	RESPONSABLE	MARCA	FECHAFAB	PLACA	COLOR	MODELO	TIPOPARQ	PERIODO
	123456789012	Parqueadero America	Juan Andrade	Chevrolet Spark	2010	PCR463	Rojo	HatchBack	Completo	2012-1
	987654321012	Parqueadero NNUU	Carlos Martinez	Hyundai Elantra	2009	PBB2112	Azul	Sedan	Completo	2012-1

fila(s) 1 - 2 de 2

Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Código Vehículo	Identificador del vehículo: 12 Números Identificador Numerador (12 Números)	Identificador del vehículo: +12 Números	Registro almacenado en base de datos Campo VARCHAR2(12)
Código Parqueadero	Identificador parqueadero: 25 Letras	Identificador parqueadero: Sin errores truncado con Combo Box	Registro almacenado en base de datos Campo CHAR(25)
Responsable	Responsable Vehículo: 40 Letras o Números	Responsable Vehículo: + 40 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(40)
Marca	Marca Vehículo: 25 Letras o Números	Marca Vehículo: + 25 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(25)
Año Fabricación	Año de Fabricación Vehículo: 4 Letras o Números	Año de Fabricación Vehículo: + 4 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(4)
Placa	Placa Vehículo: 7 Letras o Números	Placa Vehículo: + 7 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(7)
Color	Color Vehículo: 15 Letras o Números	Color Vehículo: + 15 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(15)
Modelo	Modelo Vehículo: 25 Letras o Números	Modelo Vehículo: Sin errores truncado con Combo Box	Registro almacenado en base de datos Campo VARCHAR2(25)
Tipo	Tipo Autorización: 10 Letras o Números	Tipo Autorización: Sin errores truncado con Combo Box	Registro almacenado en base de datos Campo VARCHAR2(10)
Periodo	Periodo Parqueo: 6 Letras o Números	Periodo Parqueo: + 6 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(6)

Ingreso Espacio



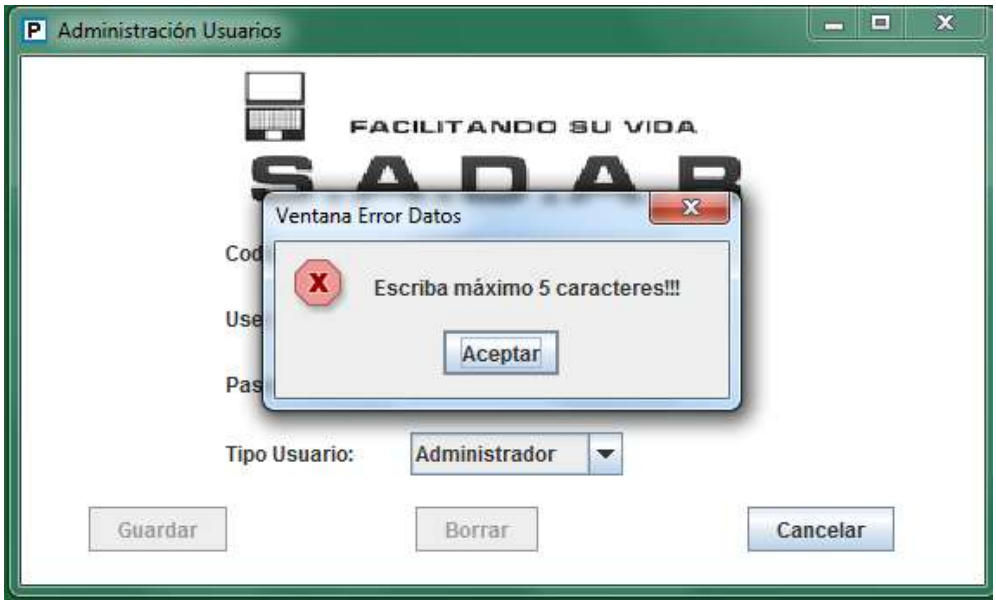
EDITAR	CODIGOESPACIO	CODIGOPARQUEADERO	DESCRIPCION	DISPONIBILIDAD
	S1A01	Parqueadero America	Subsuelo 1 Espacio A01	No
	S1A02	Parqueadero America	Subsuelo 1 Espacio A02	No
	S1A03	Parqueadero America	Subsuelo Lugar A03	Si

fila(s) 1 - 3 de 3

Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Código Espacio	Identificador del Espacio: 2 Letras 4 Números Identificador Piso y Letra (2 Letras) Numerador (4 Números)	Identificador del Espacio: + 2 Letras + 4 Números	Registro almacenado en base de datos Campo VARCHAR2(6)
Código Parqueadero	Identificador parqueadero: 25 Letras	Identificador parqueadero: Sin errores truncado con Combo Box	Registro almacenado en base de datos Campo CHAR(25)
Descripción	Descripción Ubicación: 25 Letras o Números	Descripción Ubicación: + 25 Letras o Números	Registro almacenado en base de datos Campo VARCHAR2(25)
Disponibilidad	Disponibilidad Espacio: 2 Letras	Disponibilidad Espacio: Sin errores truncado con Combo Box	Registro almacenado en base de datos Campo VARCHAR2(2)

4.4 Pruebas de Validaciones de Usuario

Ingreso Usuarios



Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Código Usuario	Identificador del usuario: 5 Caracteres	Identificador del usuario: + 5 Caracteres	Ventana de Mensaje de error supero caracteres
User	Nombre del Usuario: 15 Caracteres	Nombre del Usuario: + 15 Letras o Números	Ventana de Mensaje de error supero caracteres
Password	Clave de Ingreso: 10 Caracteres	Clave Ingreso: + 10 Caracteres	Ventana de Mensaje de error supero caracteres
Tipo	Tipo Acceso: 15 Caracteres	Tipo Acceso: Sin errores truncado con Combo Box	

Ingreso Parqueadero



Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Código Parqueadero	Identificador parqueadero: 25 Caracteres	Identificador parqueadero: + 25 Caracteres	Ventana de Mensaje de error supero caracteres
Descripción	Ubicación Parqueadero: 25 Caracteres	Nombre del Usuario: + 25 Caracteres	Ventana de Mensaje de error supero caracteres

Ingreso Vehículo



Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Código Vehículo	Identificador del vehículo: 12 Números	Identificador del vehículo: +12 Números	Ventana de Mensaje de error supero caracteres no puede ingresar letras
Código Parqueadero	Identificador parqueadero: 25 Caracteres	Identificador parqueadero: Sin errores truncado con Combo Box	
Responsable	Responsable Vehículo: 40 Caracteres	Responsable Vehículo: + 40 Caracteres	Ventana de Mensaje de error supero caracteres
Marca	Marca Vehículo: 25 Caracteres	Marca Vehículo: + 25 Caracteres	Ventana de Mensaje de error supero caracteres
Año Fabricación	Año de Fabricación Vehículo: 4 Caracteres	Año de Fabricación Vehículo: + 4 Caracteres	Ventana de Mensaje de error supero caracteres
Placa	Placa Vehículo: 7 Caracteres	Placa Vehículo: + 7 Caracteres	Ventana de Mensaje de error supero caracteres
Color	Color Vehículo: 15 Caracteres	Color Vehículo: + 15 Caracteres	Ventana de Mensaje de error supero caracteres
Modelo	Modelo Vehículo: 25 Caracteres	Modelo Vehículo: Sin errores truncado con Combo Box	
Tipo	Tipo Autorización: 10 Caracteres	Tipo Autorización: Sin errores truncado con Combo Box	
Periodo	Periodo Parqueo: 6 Caracteres	Periodo Parqueo: + 6 Caracteres	Ventana de Mensaje de error supero caracteres

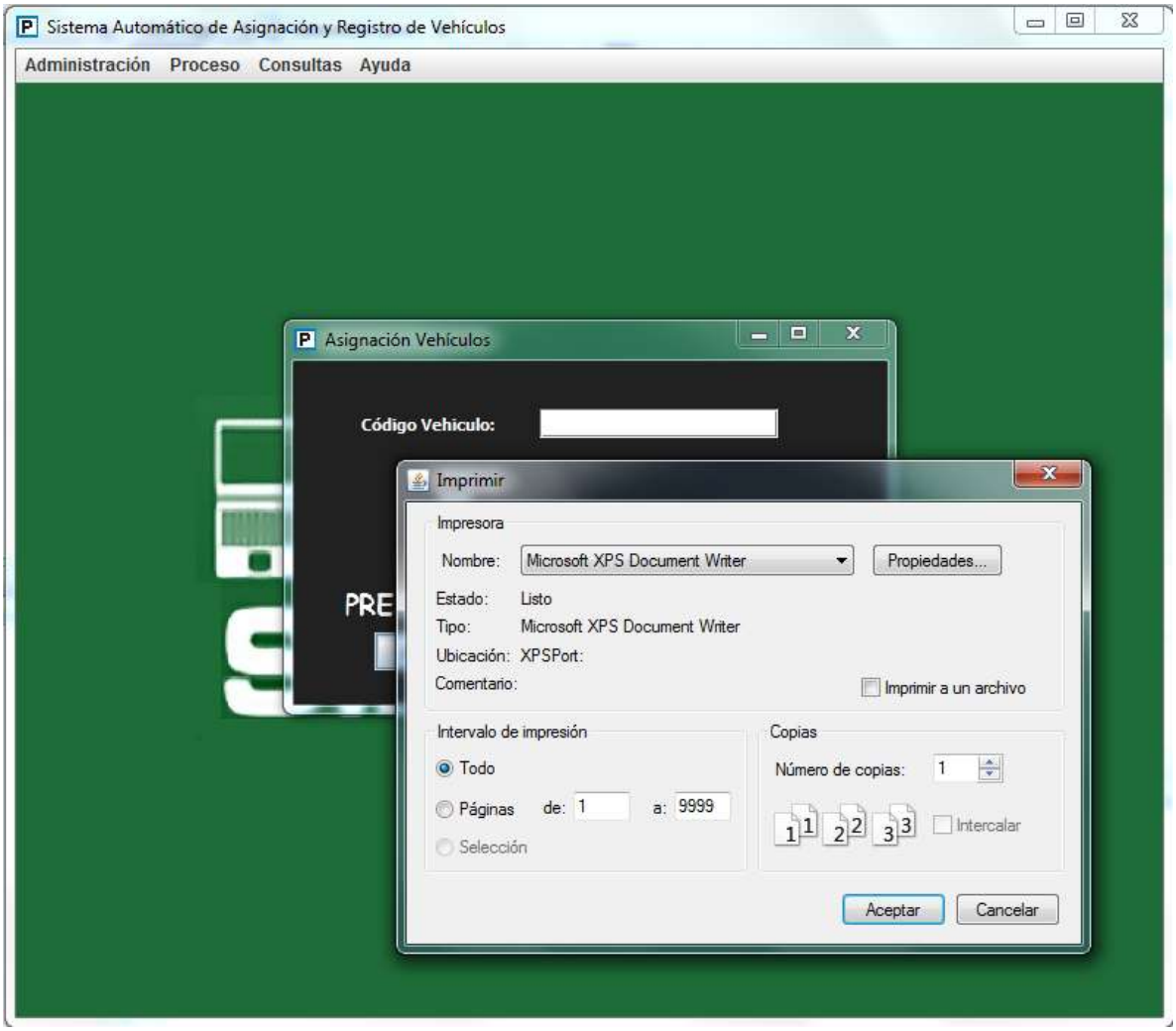
Ingreso Espacio



Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Código Espacio	Identificador del Espacio: 6 Caracteres	Identificador del Espacio: + 6 Caracteres	Ventana de Mensaje de error supero caracteres
Código Parqueadero	Identificador parqueadero: 25 Caracteres	Identificador parqueadero: Sin errores truncado con Combo Box	
Descripción	Descripción Ubicación: 25 Caracteres	Descripción Ubicación: + 25 Letras o Números	Ventana de Mensaje de error supero caracteres
Disponibilidad	Disponibilidad Espacio: 2 Caracteres	Disponibilidad Espacio: Sin errores truncado con Combo Box	

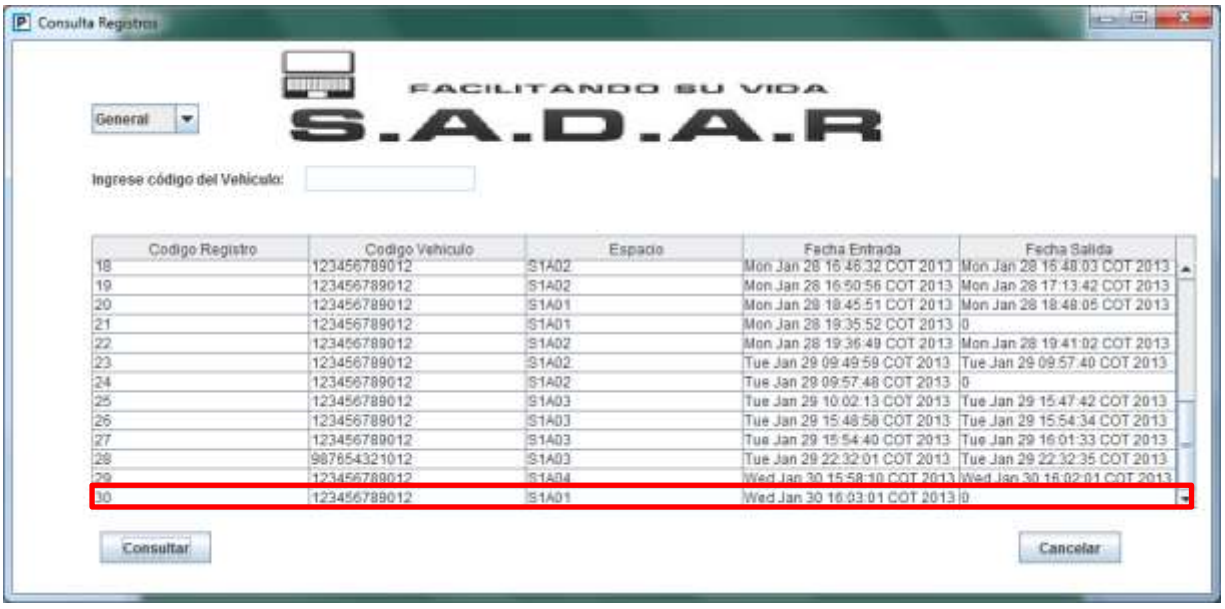
4.4 Pruebas de Funcionamiento

Proceso Asignación



Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Código Vehículo	Código Valido	Código No Válido	Guardar Registro de Uso Asignar espacio y quitar disponibilidad. Impresión Ticket de Asignación

Registro Actualizado



Codigo Registro	Codigo Vehiculo	Espacio	Fecha Entrada	Fecha Salida
18	123456789012	S1A02	Mon Jan 28 16:46:32 COT 2013	Mon Jan 28 16:48:03 COT 2013
19	123456789012	S1A02	Mon Jan 28 16:50:56 COT 2013	Mon Jan 28 17:13:42 COT 2013
20	123456789012	S1A01	Mon Jan 28 18:45:51 COT 2013	Mon Jan 28 18:48:05 COT 2013
21	123456789012	S1A01	Mon Jan 28 19:35:52 COT 2013	0
22	123456789012	S1A02	Mon Jan 28 19:36:49 COT 2013	Mon Jan 28 19:41:02 COT 2013
23	123456789012	S1A02	Tue Jan 29 09:49:59 COT 2013	Tue Jan 29 09:57:40 COT 2013
24	123456789012	S1A02	Tue Jan 29 09:57:48 COT 2013	0
25	123456789012	S1A03	Tue Jan 29 10:02:13 COT 2013	Tue Jan 29 15:47:42 COT 2013
26	123456789012	S1A03	Tue Jan 29 15:48:58 COT 2013	Tue Jan 29 15:54:34 COT 2013
27	123456789012	S1A03	Tue Jan 29 15:54:40 COT 2013	Tue Jan 29 16:01:33 COT 2013
28	987654321012	S1A03	Tue Jan 29 22:32:01 COT 2013	Tue Jan 29 22:32:35 COT 2013
29	123456789012	S1A04	Wed Jan 30 15:58:10 COT 2013	Wed Jan 30 16:02:01 COT 2013
30	123456789012	S1A01	Wed Jan 30 16:03:01 COT 2013	0

Impresión del Ticket



Proceso Reporte



Campo	Entradas Esperadas	Rango de Errores	Salidas Esperadas
Mes	Elegir un Mes	Ninguno Truncado por Combo Box	Calculo del porcentaje utilización del parqueadero por mes
Hora	Elegir una Hora	Ninguno Truncado por Combo Box	Calculo del porcentaje utilización del parqueadero por hora
			Calculo total de utilización total del Mes escogido

5. CAPITULO 5: Conclusiones y Recomendaciones

Conclusiones:

- ✓ El manejo de la metodología RAD en este proyecto se acopló de manera perfecta ya que por sus características está orientado al desarrollo de prototipos además, de que su tiempo de desarrollo se apegó a las necesidades del proyecto.
- ✓ El proceso de manejo de parqueaderos involucra a muchas áreas las cuales son fáciles de identificar dentro de la herramienta SIPOC, con ello involucrarse más con la naturaleza del sistema resulta más sencillo.
- ✓ La cadena de valor permite una mejor visualización de todas las actividades que se ven involucradas dentro del proceso de levantamiento de requerimientos y los posibles actores que tendrán que involucrarse.
- ✓ El diagramado UML es esencial al momento de visualizar como se diseñará el sistema ya que nos ubica todos los elementos necesarios dentro del desarrollo así como todas las clases involucradas en determinado proceso.
- ✓ El levantamiento de este sistema facilitará la administración del parqueadero por ofrecer reportes completos de uso, asignación totalmente automática y facilitación de manejo para el usuario final.

Recomendaciones:

- ➢ Se recomienda la implementación de este prototipo dentro de la universidad para el mejoramiento de seguridad y servicio de los parqueaderos para los estudiantes docentes y administrativos de la universidad.
- ➢ Se considera prudente realizar back up’s mensuales de la tabla registros con la finalidad de llevar un completo informe de todos los registros realizados durante el mes, y poder obtener datos estadísticos.
- ➢ La implementación de una auditoria de todas las actividades realizadas dentro de la base de datos todas estas asociadas al usuario que se conecta a dicho momento.
- ➢ El uso de código de barras al proyecto se enfocó básicamente para obviar la digitación del usuario final, pero se podría utilizar aparatos de proximidad en futuras actualizaciones del sistema para facilitar aún más el manejo del mismo.
- ➢ Es prudencial sugerir que para un óptimo uso del sistema el usuario debe respetar las políticas y normas de uso de plazas previamente establecidas.

BIBLIOGRAFÍA

INTERNET:

- http://es.wikipedia.org/wiki/Ingenier%C3%ADa_de_software, Enciclopedia Libre "Ingeniería de Software", WIKIPEDIA.ORG, 30/09/2012.
- http://es.wikipedia.org/wiki/Diagramas, Enciclopedia Libre "Diagramas", WIKIPEDIA.ORG, 30/09/2012.
- http://es.scribd.com/doc/5366642/02-Analisis-de-problemas-de-estacionamiento, Documento "ANÁLISIS DE PROBLEMAS DE ESTACIONAMIENTO EN GRANDES INSTITUCIONES: APLICACIÓN AL CASO DE UN CAMPOS UNIVERSITARIO", Patricio Donoso Ibáñez, 03/09/2012.
- http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema04.pdf, Diapositivas "Metodologías de Desarrollo de Software", Grupo Alarcos, 03/09/2012.
- http://www.buenastareas.com/ensayos/Sistema-De-Monitoreo-Para-Ingreso-De/582443.html, Ensayo "Sistema De Monitoreo Para Ingreso De Automoviles A Parqueaderos", "Sitio Web de Apoyo Educativo", ComScore, 16/10/2012.
- http://www.sznet.com.ar/sistemas/Informacion.php , Documento "Información sobre Código de Barras", sz.net Empresa ayuda Áreas Informáticas, 16/10/2012
- http://es.wikipedia.org/wiki/Fluctuaci%C3%B3n, Enciclopedia Libre "Fluctuación", WIKIPEDIA.ORG, 16/10/2012.
- http://es.wikipedia.org/wiki/Decodificador, Enciclopedia Libre "Decodificador", WIKIPEDIA.ORG, 16/10/2012.
- http://es.wikipedia.org/wiki/Diodo, Enciclopedia Libre "Diodo", WIKIPEDIA.ORG, 16/10/2012.
- http://www.docsolutions.com.mx/es/soluciones/servicios/digitalizacion.html, Web DocSolutions Documento "Digitalicación", DocSolutions, 16/10/2012.
- http://www.ia.uned.es/ia/asignaturas/adms/GuiaDidADMS/node9.html , Web de Información "Necesidad de una metodología", José R. Álvarez y Manuel Arias – UNED, 16/10/2012
- http://es.scribd.com/doc/51066630/64/Metodologias-orientadas-a-objetos, Biblioteca en Línea, Documento "SISTEMAS DE INFORMACIÓNEN LA ERA DIGITAL", Luis Alberto Mamani Pinto, 16/10/2012.
- http://glbrtlmb.blogdiario.com/i2006-09/, Blog de Opinion Libre Documento "ANALISIS DISEÑO ORIENTADO A OBJETOS", BlogDiario.com, 16/10/2012.

TRABAJOS CITADOS

[1] I. Wampy, «Prodigy Basico e Infinitum,» Scrib corp, 15 Mayo 2010. [En línea]. Available: http://es.scribd.com/doc/51688653/2006-Prodigy-Basico-e-Infinitum. [Último acceso: 19 Diciembre 2012].

[2] «spanishpmo.com,» © SpanishPMO, 28 Enero 2012. [En línea]. Available: http://spanishpmo.com/index.php/ciclos-de-vida-modelo-de-cascada/. [Último acceso: 19 Diciembre 2012].

[3] R. E. Gonzalez, «Polilibros,» Polilibros.com, 22 Agosto 2011. [En línea]. Available: http://148.204.211.134/polilibros/portal/Polilibros/P_externos/Administracion_informatica_de_las_organizaciones_Ramon_E_Enriquez_Gonzalez/AIO2_Mod_ESPIRAL.html. [Último acceso: 19 Diciembre 2012].

[4] S. Cruz, «Intro Ingeniería del Software,» blogspot.es, 16 Septiembre 2010. [En línea]. Available: http://scruz334.blogspot.es/1193169600/. [Último acceso: 19 Diciembre 2012].

[5] WikiPedia, «Rapid application development,» wikipedia.org, 12 Noviembre 2012. [En línea]. Available: http://en.wikipedia.org/wiki/Rapid_application_development. [Último acceso: 19 Diciembre 2012].

[6] G. Torossi, «Diseño de Sistemas,» chaco.gov.ar, 23 Agosto 2011. [En línea]. Available: http://www.chaco.gov.ar/utn/disenodesistemas/apuntes/oo/ApunteRUP.pdf. [Último acceso: 19 Diciembre 2012].

[7] ORACLE, «Qué es la tecnología JAVA,» java.com, 22 Noviembre 2011. [En línea]. Available: http://www.java.com/es/download/faq/whatis_java.xml. [Último acceso: 21 Diciembre 2012].

[8] Wikipedia, «Java (Lenguaje de Programación),» wikipedia.org, 17 Febrero 2011. [En línea]. Available: http://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n). [Último acceso: 21 Diciembre 2012].

[9] Wikipedia, «Oracle Database,» wikipedia.org, 8 Marzo 2011. [En línea]. Available: http://es.wikipedia.org/wiki/Oracle_Database. [Último acceso: 21 Diciembre 2012].

[10] J. Reyes, «Ventajas e inconvenientes de las diferentes bases de datos,» todoexpertos.com, 2 Abril 2010. [En línea]. Available: http://www.todoexpertos.com/categorias/tecnologia-e-internet/bases-de-datos/oracle/respuestas/14706/cuales-son-las-principales-ventajas-e-

inconvenientes-de-las-diferentes-bases-de-datos. [Último acceso: 21 Diciembre 2012].

[11] Wikipedia, «NetBeans,» wikipedia.org, 25 Abril 2011. [En línea]. Available: http://es.wikipedia.org/wiki/NetBeans. [Último acceso: 21 Diciembre 2012].

[12] Oracle, «NetBeans IDE Features,» http://netbeans.org, 14 Mayo 2012. [En línea]. Available: http://netbeans.org/features/index.html. [Último acceso: 21 Diciembre 2012].

[13] SZnet, «Información sobre códigos de Barras,» sznet.com, 17 Enero 2010. [En línea]. Available: http://www.sznet.com.ar/sistemas/Informacion.php. [Último acceso: 16 Octubre 2012].

[14] Wikipedia, «Procesos,» wikipedia.org, 26 Marzo 2012. [En línea]. Available: http://es.wikipedia.org/wiki/Proceso. [Último acceso: 21 Diciembre 2012].

[15] O. Espinosa, «Procesos,» Autoria Propia, Quito, 21012.

[16] Definicion.de, «Cadena de Valor,» definicion.de, 1 Abril 2008. [En línea]. Available: http://definicion.de/cadena-de-valor/#ixzz2GwB8iLwV. [Último acceso: 21 Diciembre 2012].

[17] Wikipedia, «Cadena de Valor,» wikipedia.org, 23 Junio 2011. [En línea]. Available: http://es.wikipedia.org/wiki/Cadena_de_valor. [Último acceso: 21 Diciembre 2012].

[18] SIG, «Mapa de Procesos,» solucionessig.com, 18 Abril 2012. [En línea]. Available: http://www.solucionessig.com/portal/mapa-de-procesos. [Último acceso: 21 Diciembre 2012].

[19] R. O. Grazon, «Mapa de Procesos,» scribd.com, 26 Julio 2012. [En línea]. Available: http://es.scribd.com/doc/84774712/MAPAS-DE-PROCESOS. [Último acceso: 21 Diciembre 2012].

[20] T. Bozheva, «Gestion de Procesos,» blogspot.com, 3 Marzo 2011. [En línea]. Available: http://teodorabozheva.blogspot.com/2011/03/sipoc-la-definicion-de-tu-proceso-en.html. [Último acceso: 24 Enero 2013].

[21] J. Chinchilla, «Diagrama SIPOC,» engindustrial.blogspot.com, 9 Agosto 2009. [En línea]. Available: http://engindustrial.blogspot.com/2009/08/modelo-sipoc.html. [Último acceso: 24 Enero 2013].

ENTREVISTA CON ESPECIALISTAS:

- **CALDERÓN SERRANO JORGE ALFREDO**, Ingeniero de Sistemas, Masterado: Master en Filosofía, Sistemas Operativos, Herramientas Case, Procesos de Negocio, Desarrollo de Sistemas, Filosofía de la Ciencia, acalderon@puce.edu.ec
- **LUIS OSWALDO ESPINOSA VITERI**, Ingeniero en Informática, Master en Ciencias - Mención Gestión Tecnológica, Estructuras de Datos, Compiladores, Sistemas Operativos, Inteligencia Artificial, Gestión de las Tecnologías de la Información, Marketing Tecnológico, Procesos Calidad y Mejoramiento Continúo, loespinosa@puce.edu.ec

INVESTIGADOR:

- Juan Carlos Andrade, Estudiante de Ingeniería en Sistemas, PUCE, Egresado Ingeniería

ANEXOS

ANEXO 1.1



On-Counter Presentation Omnidirectional Bar Code Reader

The Magellan™ 1100i omnidirectional presentation scanner is a high performing and versatile data collection tool for a variety of applications. It uses Datalogic ADC's latest digital imaging technology to provide intuitive and fast reading performance on 1D and 2D bar codes as well as supporting advanced features like image capture and EAS tag deactivation. A revolutionary External Read Indicator (ERI) allows the Magellan 1100i reader to integrate with other third party RFID or EAS systems for a truly "future-proof" solution.

Datalogic's Illumix™ intelligent illumination technology in the Magellan 1100i reader optimizes light levels to automatically capture images and read codes from mobile phones, PDAs, or computer display screens. Illumix technology also provides incredible motion tolerance, far beyond most other 2D scanning solutions, enabling both sweep and presentation scanning techniques.

The Magellan 1100i reader's compact size makes it the ideal solution for space-constrained environments. With a precise reading area, it is perfect for hands-free or handheld scanning and helps reduce accidental misreads of items caused by stray laser scan lines - improving productivity and customer satisfaction.

Datalogic ADC's patented Green Spot goodread indicator improves user feedback by projecting a visual green spot onto bar codes just read. The additional visual feedback makes it easy to use in noisy retail applications and in healthcare or library applications that benefit from silent reading. The Green Spot can also be used to target a single code from a group of codes or to position a single code inside the reading area.

A comprehensive multi-interface set and a powerful label-editing feature allows easy integration into existing or new host systems. Magellan Value Added Features such as Diagnostics Reporting, Host Download, plus optional Productivity Index Reporting™ allow management to use data from the scanner to improve throughput and manage maintenance. In addition, each scanner supports Remote Management Systems to streamline operations and ongoing support across an enterprise.

Features

• Aggressive performance on 1D and optional 2D codes increases productivity
• Digital imaging technology eliminates moving parts for improved reliability
• Illumix™ intelligent illumination technology auto-detects bar codes from mobile phones, laptops or PDAs
• Compact, ergonomic design is ideal for hands-free or handheld operation
• Green Spot good-read indicator provides additional user feedback on button equipped models
• Precise imaging scan volume eliminates misreads caused by stray laser scan lines
• Button model for image capture or targeting applications like Price Look Up (PLU) lists
• Checkpoint® Systems EAS and Interlock compatible
• WMI Compliant; Integrates with IBM Director through RMA

Specifications

Deco ding Capability

1D / Linear Codes Autodiscriminates all standard 1D codes including GS1
DataBar™ linear codes.
2D Codes Aztec Code, Data Matrix, MaxiCode, QR Code
Stacked CodE s GS1 DataBar Composites; GS1 DataBar Composites; GS1
DataBar Expanded Stacked; GS1 DataBar Stacked; GS1 DataBar
Stacked Omnidirectional; Micro PDF417, PDF417

Electrical

Current Operating (Typical): < 400 mA
Standby/Idle (Typical): < 300 mA
Input Voltage 4.5 - 14.0 VDC

Environmental

Ambient Light Indoor: 0 to 6,000 lux; Outdoor: 0 to 86,100 lux
Drop Resistance Withstands repeated drops from 1.2 m / 4.0 ft onto a concrete
surface
ES D Protection (Air Discharge) 25 kV
Humidity (Non-Condensing) 5 - 95%
Particulate and Water Sealing IP52
Temperature Operating: 0 to 40 °C / 32 to 104 °F
Storage/Transport: -40 to 70 °C / -40 to 158 °F

Interfaces

Interfaces IBM 46xx; Keyboard Wedge; OEM (IBM) USB; RS-232; USB

Optional Features

EAS Features Checkpoint Systems: Integrated EAS hardware will work with
Counterpoint IX and Evolve D11 models
Sensormatic: Good Read Acknowledgement
Value Add ed Features Diagnostic Reporting; Host Download; Productivity Index
Reporting™ (PIR)

Physical Characteristics

Colors Available Black; Grey; Other colors and custom logo options are available
for minimum quantity purchase.
Dimensions Riser Stand: 5.9 x 7.6 x 9.7 cm / 2.3 x 3.0 x 3.8 in
Scanner: 8.4 x 7.1 x 9.4 cm / 3.3 x 2.8 x 3.7 in
Weight Riser Stand: 187.1 g / 6.6 oz
Scanner: 198.5 g / 7.0 oz

Reading Perfo rmance

Image Capture Optional: 752 x 480 pixels; Graphic Formats: JPEG
Print Contrast Ratio (Minimum) 25%
Read Rate (Maximum) 1,768 digital scan lines/sec.
Reading Angle Pitch: +/- 65°; Roll (Tilt): 0 - 360°; Skew (Yaw): +/- 75°
Reading Indicators Beeper (Adjustable Tone and Volume); Datalogic 'Green Spot'
Good Read Feedback; Good Read LED
Resolution (Maximum) 0.130 mm / 5 mils
Sc an Lines Digital: 136

Reading Ranges

Typical Depth of Field Minimum distance determined by symbol length and scan
angle.
Printing resolution, contrast, and ambient light dependent.

1D / Linear Codes

5 mils: 1.3 to 4.6 cm / 0.5 to 1.8 in
7.5 mils: 0.63 to 5.71 cm / 0.25 to 2.25 in
10 mils: 0 to 12.7 cm / 0 to 5.0 in
13 mils UPC: 0 to 17.7 cm / 0 to 7.0 in
20 mils: 0 to 17.7 cm / 0 to 7.0 in

Optional Codes

6.6 mils PDF: 3.8 to 7.6 cm / 1.5 to 3.0 in
10 mils Data Matrix: 0 to 4.5 cm / 0 to 1.8 in
10 mils PDF: 3.1 to 15.2 cm / 1.2 to 6.0 in
15 mils PDF: 2.5 to 15.2 cm / 1.0 to 6.0 in
24 mils Data Matrix: 0 to 12.7 cm / 0 to 5.0 in

Safety & Regulatory

Agency Approvals The product meets necessary safety and regulatory approvals
for its intended use.
The Quick Reference Guide for this product can be referred to
for a complete list of certifications.
Environmental Compliance Complies to EU RoHS
LED Classification IEC 60825 Class 1 LED

Utilities

OPOS / JavaPOS JavaPOS Utilities are available for download at no charge.
OPOS Utilities are available for download at no charge.
Remote Host Download Lowers service costs and improves operations.

Warranty

Warranty 2-Year Factory Warranty

ANEXO 1.2



On-Counter Presentation Omnidirectional Bar Code Reader

The Magellan™ 800i omnidirectional presentation scanner is a versatile data collection tool for many applications. It uses Datalogic ADC's latest digital imaging technology to provide intuitive and fast reading of 1D and 2D bar codes as well as supporting more advanced features like image capture and EAS tag deactivation. Despite its small size, the Magellan 800i reader provides big performance and reliability along with a rich feature set to provide excellent value in virtually any data collection system.

The Magellan 800i reader uses third generation Illumix™ intelligent illumination technology to optimize light levels for automatically reading bar codes from mobile phones and for capturing images. Illumix technology also provides incredible motion tolerance, exceeding other 2D scanning solutions, enabling both 'sweep' and 'presentation' style scanning techniques.

When space is limited but high performance is required, look no further than the Magellan 800i reader. Its compact size and footprint make it the ideal solution for space-constrained environments. Its precise imaging scan volume is perfect for hands-free scanning in confined spaces and helps reduce accidental misreads of items caused by stray laser scan lines, improving productivity and customer satisfaction. For large, heavy or awkward-sized items, the Magellan 800i reader can easily be picked up for handheld use because of its small, ergonomic enclosure.

Like all Magellan bar code scanners, the Magellan 800i reader includes a powerful label editing tool which can parse or manipulate label data and also allows easy integration into existing or new host systems. The Magellan family of value added features (including Diagnostics Reporting, Host Download, and Advanced Label Editing) allow management to use valuable data from the scanner to improve throughput and manage maintenance. In

addition, each Magellan 800i reader supports Remote Management Systems to streamline operations and ongoing support across an enterprise.
Improve your productivity and expand your data collection possibilities at the POS and beyond with the versatile Magellan 800i omnidirectional bar code reader.

Features

• Aggressive performance on 1D and 2D codes increases productivity
• Powerful Advanced Data Editing for custom host output
• No moving parts for improved reliability
• Illumix™ intelligent illumination technology for reading bar codes from mobile phones or laptops
• Ergonomically designed for hands-free or handheld operation
• Small footprint for easy installation into space constrained applications
• Precise imaging scan volume eliminates misreads caused by stray laser scan lines
• Configurable image capture mode to optimize host upload time
• Checkpoint Systems EAS and Interlock compatible

Specifications

Deco ding Capability

1D / Linear Codes Autodiscriminates all standard 1D codes including GS1 DataBar™
linear codes
2D Codes Aztec Code; Data Matrix; MaxiCode; QR Code
Stacked Codes GS1 DataBar Expanded Stacked; GS1 DataBar Stacked; GS1
DataBar Stacked Omnidirectional; MicroPDF417; PDF417

Electrical

Current Operating (Typical): < 350 mA
Standby/Idle (Typical): < 250 mA
Input Voltage 5 VDC (+/- 10%)

Environmental

Ambient Light Indoor: 0 to 6,000 lux; Outdoor: 0 to 86,000 lux
Drop Resistance Withstands repeated drops from 1.2 m / 4.0 ft onto a
concrete surface
ES D Protection (Air Discharge) 20 kV
Humidity (Non-Condensing) 5 - 95%
Particulate and Water Sealing IP52
Temperature Operating: 0 to 40 °C / 32 to 104 °F
Storage/Transport: -40 to 70 °C / -40 to 158 °F

Interfaces

Interfaces RS-232; USB: OEM USB, USB COM, USB HID Keyboard, USB
TEC

Optional Features

EAS Features Checkpoint Systems: Integrated EAS hardware will work
with Counterpoint IX and Evolve D11 models
Sensormatic: Good Read Acknowledgement
Value Add ed Features Advanced Label Editing; Diagnostic Reporting; Host
Download

Physical Characteristics

Colors Available Black; White; Other colors and custom logo options are
available for minimum quantity purchase.
Dimensions 15.1 x 7.6 x 10.9 cm / 5.9 x 3.0 x 4.3 in
Weight 306.2 g / 10.8 oz

Reading Perfo rmance

Image Capture Optional: 752 x 480 pixels; Graphic Formats: JPEG
Print Contrast Ratio (Minimum) 25%
Reading Angle Pitch: +/- 60°; Roll (Tilt): 0 - 360°; Skew (Yaw): +/- 60°
Reading Indicators Beeper (Adjustable Tone and Volume); Good Read LED
Resolution (Maximum) 0.130 mm / 5 mils

Reading Ranges

Typical Depth of Field Minimum distance determined by symbol length and scan
angle.
Printing resolution, contrast, and ambient light dependent.
1D / Linear Codes
5 mils: 0 to 2.5 cm / 0 to 1.0 in
7.5 mils: 0 to 8.2 cm / 0 to 3.2 in
10 mils: 0 to 12.7 cm / 0 to 5.0 in
13 mils UPC: 0 to 15.2 cm / 0 to 6.0 in
2D Codes
6.6 mils PDF417: 0 to 7.6 cm / 0 to 3.0 in
10 mils Data Matrix: 0.6 to 3.1 cm / 0.2 to 1.2 in
10 mils PDF417: 0 to 12.7 cm / 0 to 5.0 in
15 mils PDF417: 0 to 15.2 cm / 0 to 6.0 in
24 mils Data Matrix: 0 to 12.7 cm / 0 to 5.0 in

Safety & Regulatory

Agency Approvals The product meets necessary safety and regulatory
approvals for its intended use.
The Quick Reference Guide for this product can be referred
to for a complete list of certifications.
Environmental Compliance Complies to China RoHS; Complies to EU RoHS

Utilities

Magell an™ OSP Magellan On-Screen Programming (OSP) configuration
software is available for download at no charge.
OPOS / JavaPOS JavaPOS Utilities are available for download at no charge.
OPOS Utilities are available for download at no charge.
Remote Host Download Lowers service costs and improves operations.

Warranty

Warranty 3-Year Factory Warranty

MANUAL TÉCNICO

Empaquetado: DP **Componente:** EspaciosDP.java

```
package dp;
import md.EspacioMD;
import java.util.*;

public class EspaciosDP {
   private String codigoEspacio;
   private String codigoParqueadero;
   private String descripcion;
   private String disponibilidad;
   private EspacioMD espacioMD=new
EspacioMD(this);
   public EspaciosDP() {
   }
     public EspaciosDP(String codigoE, String
codigoP, String des, String disp) {
        this.codigoEspacio = codigoE;
        this.codigoParqueadero = codigoP;
        this.descripcion = des;
        this.disponibilidad = disp;
    }
     public EspaciosDP(String codigoE, String
codigoP,String des) {
        this.codigoEspacio = codigoE;
        this.codigoParqueadero = codigoP;
        this.descripcion = des;
     }
     public EspaciosDP(String codigo) {
        this.codigoEspacio = codigo;
     }
        public String getCodigoEspacio() {
      return codigoEspacio;
   }
   public void setCodigoEspacio(String
newCodigoEspacio) {
      this.codigoEspacio = newCodigoEspacio;
   }
   public String getCodigoParqueadero() {
      return codigoParqueadero;
   }
   public void setCodigoParqueadero(String
newCodigoParqueaderoVehiculo) {
      this.codigoParqueadero =
newCodigoParqueaderoVehiculo;
   }
   public String getDescripcion() {
      return descripcion;
   }
   public void setDescripcion(String newDescripcion)
{
      this.descripcion = newDescripcion;
   }
   public String getDisponibilidad() {
      return disponibilidad;
   }
   public void setDisponibilidad(String
newDisponibilidad) {
      this.disponibilidad = newDisponibilidad;
   }
      public boolean verificarDP(){
        if(espacioMD.verificarMD()==true){
            return true;
        }
        else{
            return false;
        }
    }
     public void ingresarDP(){
        espacioMD.ingresarMD();
    }
      public void eliminarDP() {
        espacioMD.eliminarMD();
    }
     public String[] verDatosDP() {
        return espacioMD.verDatosMD();
    }
      public void modificarDP() {
        espacioMD.modificarMD();
    }
  public EspaciosDP[] verServiciosDP(String
verespaciosP,int band) {
        return
espacioMD.verServiciosMD(verespaciosP,band);
    }

   public HashMap getHashParqueaderoDP(){
      return espacioMD.getHashParqueaderoMD();
   }
}
```

Empaquetado: DP **Componente:** ParqueaderoDP.java

```
package dp;
import java.util.*;
import md.ParqueaderoMD;

public class ParqueaderoDP {

   private ParqueaderoMD parqueaderoMD=new
ParqueaderoMD(this);
   private String codigoParqueadero;
   private String descripcion;

     public ParqueaderoDP() {
   }

  public ParqueaderoDP(String cod,String des)
  {
     this.codigoParqueadero=cod;
     this.descripcion=des;
  }

  public ParqueaderoDP(String cod)
  {
     this.codigoParqueadero=cod;
  }

   public String getCodigoParqueadero() {
       return codigoParqueadero;
   }
    public void setCodigoParqueadero(String
codigo) {
       this.codigoParqueadero = codigo;
   }
   public String getDescripcion() {
       return descripcion;
   }
    public void setDescripcion(String des) {
      this.descripcion = des;
   }
   public boolean verificarDP(){
      if(parqueaderoMD.verificarMD()==true){
         return true;
      }
      else{
         return false;
      }
   }

public void ingresarDP(){
      parqueaderoMD.ingresarMD();
   }
    public void eliminarDP() {
      parqueaderoMD.eliminarMD();
   }

   public String[] verDatosDP() {
      return parqueaderoMD.verDatosMD();
   }

   public void modificarDP() {
      parqueaderoMD.modificarMD();
   }

   public ParqueaderoDP[] verServiciosDP(String
verusuariosP,int band) {
      return
parqueaderoMD.verServiciosMD(verusuariosP,band
);
   }
}
```

Empaquetado: DP **Componente:** UsuarioDP.java

```
package dp;

import java.util.*;
import md.*;

public class UsuarioDP {
  private String codigoUsuario;
  private String userUsuario;
  private String passwordUsuario;
  private String tipoUsuario;
  private UsuarioMD usuarioMD=new
UsuarioMD(this);
  private IngresoMD ingresomd=new IngresoMD();

  public UsuarioDP() {

  }

  public UsuarioDP(String codigo, String user, String
password, String tipo) {
    this.codigoUsuario = codigo;
    this.userUsuario = user;
    this.passwordUsuario = password;
    this.tipoUsuario = tipo;
  }

  public UsuarioDP(String codigo, String user, String
password) {
      this.codigoUsuario = codigo;
      this.userUsuario = user;
      this.passwordUsuario = password;
   }
   public UsuarioDP(String codigo) {
      this.codigoUsuario = codigo;
   }

  public String getCodigoUsuario() {
    return codigoUsuario;
  }

  public void setCodigoUsuario(String
newCodigoUsuario) {
    codigoUsuario = newCodigoUsuario;
  }

  public String getUserUsuario() {
    return userUsuario;
  }

  public void setUserUsuario(String
newUserUsuario) {
    userUsuario = newUserUsuario;
  }

  public String getPasswordUsuario() {
    return passwordUsuario;
  }

  public void setPasswordUsuario(String
newPasswordUsuario) {
    passwordUsuario = newPasswordUsuario;
  }

  public String getTipoUsuario() {
    return tipoUsuario;
  }

  /** @param newRolUsuario
   * @pdOid 5a9f4df1-2aaa-4d5c-9b43-
aa7c1cda891e */
  public void setTipoUsuario(String newTipoUsuario)
{
    tipoUsuario = newTipoUsuario;
  }

  public boolean verificarDP(){
      if(usuarioMD.verificarMD()==true){
         return true;
      }
      else{
         return false;
      }
   }
    public void ingresarDP(){
      usuarioMD.ingresarMD();
   }
     public void eliminarDP() {
      usuarioMD.eliminarMD();
   }
    public String[] verDatosDP() {
      return usuarioMD.verDatosMD();
   }
     public void modificarDP() {
      usuarioMD.modificarMD();
   }
  public UsuarioDP[] verServiciosDP() {
      return usuarioMD.verServiciosMD();
   }
  public int verificarIngreso(String user, String
password){

      String[] verificar=new String[4];

verificar=ingresomd.verificarDatosMD(user);
            if(verificar[0].equals(user))
            {

if((verificar[0].equals(user))&&(verificar[2].equals(pa
ssword)))
                  if(verificar[3].equals("Administrador  "))
```

```
        return 1;
    else
    return 2;
}
```

```
        else return 0;
        return 0;
    }
}
```

Empaquetado: DP **Componente:** VehículoDP.java

```
package dp;
import md.VehiculoMD;
import java.util.*;

public class VehiculoDP {

  private String codigoVehiculo;
  private String codigoParqueadero;
  private String responsableVehiculo;
  private String marcaVehiculo;
  private String fechaFabVehiculo;
  private String placaVehiculo;
  private String colorVehiculo;
  private String modeloVehiculo;
  private String tipoParqVehiculo;
  private String periodoVehiculo;
  private VehiculoMD vehiculoMD=new
VehiculoMD(this);
  public VehiculoDP() {
  }
    public VehiculoDP(String codigoV, String
codigoP, String nombre, String marca, String fecha,
String placa, String color,String modelo, String tipo,
String periodo) {
       this.codigoVehiculo = codigoV;
       this.codigoParqueadero = codigoP;
       this.responsableVehiculo = nombre;
       this.marcaVehiculo = marca;
       this.fechaFabVehiculo = fecha;
       this.placaVehiculo = placa;
       this.colorVehiculo = color;
       this.modeloVehiculo = modelo;
       this.tipoParqVehiculo = tipo;
       this.periodoVehiculo = periodo;
   }
    public VehiculoDP(String codigoV, String
codigoP,String nombre, String marca) {
       this.codigoVehiculo = codigoV;
       this.codigoParqueadero = codigoP;
       this.responsableVehiculo = nombre;
       this.marcaVehiculo = marca;
    }
    public VehiculoDP(String codigo) {
       this.codigoVehiculo = codigo;
    }

  public String getCodigoVehiculo() {
     return codigoVehiculo;
  }
  public void setCodigoVehiculo(String
newCodigoVehiculo) {
     this.codigoVehiculo = newCodigoVehiculo;
  }
  public String getCodigoParqueaderoVehiculo() {
     return codigoParqueadero;
  }
  public void setCodigoParqueadero(String
newCodigoParqueaderoVehiculo) {
     this.codigoParqueadero =
newCodigoParqueaderoVehiculo;
  }
  public String getResponsableVehiculo() {
     return responsableVehiculo;
  }
  public void setResponsableVehiculo(String
newResponsableVehiculo) {
     this.responsableVehiculo =
newResponsableVehiculo;
  }
  public String getMarcaVehiculo() {
     return marcaVehiculo;
  }
  public void setMarcaVehiculo(String
newMarcaVehiculo) {
     this.marcaVehiculo = newMarcaVehiculo;
  }
  public String getFechaFabVehiculo() {
     return fechaFabVehiculo;
  }
  public void setFechaFabVehiculo(String
newFechaFabVehiculo) {
     this.fechaFabVehiculo = newFechaFabVehiculo;
  }
  public void setPlacaVehiculo(String
newPlacaVehiculo) {
     this.placaVehiculo = newPlacaVehiculo;
  }
  public String getPlacaVehiculo() {
     return placaVehiculo;
  }
  public void setColorVehiculo(String
newColorVehiculo) {
     this.colorVehiculo = newColorVehiculo;
  }
  public String getColorVehiculo() {
     return colorVehiculo;
  }
  public void setModeloVehiculo(String
newModeloVehiculo) {
     this.modeloVehiculo = newModeloVehiculo;
  }
  public String getModeloVehiculo() {
     return modeloVehiculo;
  }
  public void setTipoParqVehiculo(String
newTipoParqVehiculo) {
     this.tipoParqVehiculo = newTipoParqVehiculo;
  }
  public String getTipoParqVehiculo() {
     return tipoParqVehiculo;
  }
```

```
  public void setPeriodoVehiculo(String
newPeriodoVehiculo) {
    this.periodoVehiculo = newPeriodoVehiculo;
  }
  public String getPeriodoVehiculo() {
    return periodoVehiculo;
  }

  public boolean verificarDP(){
      if(vehiculoMD.verificarMD()==true){
          return true;
      }
      else{
          return false;
      }
   }
    public void ingresarDP(){
      vehiculoMD.ingresarMD();
   }
     public void eliminarDP() {
      vehiculoMD.eliminarMD();
     }
     public String[] verDatosDP() {
        return vehiculoMD.verDatosMD();
     }
      public void modificarDP() {
        vehiculoMD.modificarMD();
     }
    public VehiculoDP[] verServiciosDP(String
verusuariosP,int band) {
        return
vehiculoMD.verServiciosMD(verusuariosP,band);
    }
    public HashMap getHashParqueaderoDP(){
      return vehiculoMD.getHashParqueaderoMD();
    }

    public HashMap getHashVehiculoDP(){
      return vehiculoMD.getHashVehiculoMD();
          }
    }
```

Empaquetado: GUI **Componente:** AsignarGUI.java

```
package gui;
import java.sql.*;
import java.text.SimpleDateFormat;
import java.util.*;
import java.util.Date;
import javax.swing.JOptionPane;
import java.awt.*;
import java.awt.print.*;
import tesis.FondoJava;

/**
 *
 * @author  Juanka
 */
public class AsignarGUI extends
javax.swing.JFrame implements Printable{
    FondoJava as= new FondoJava(7);
    private Connection conexion;
    private Statement sentencia;
    private ResultSet resultset;
    String codVehImp;
    String codEspImp;
    String fechaImp;
    Date fecha = new Date();
public AsignarGUI() {
        initComponents();
        this.add(as,BorderLayout.CENTER);
        as.repaint();
    }
private void
btnCancelarActionPerformed(java.awt.event.ActionE
vent evt) {
    this.dispose();
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }

public void conexion(){
try
        {
            Class.forName
("oracle.jdbc.driver.OracleDriver");
        }
        catch(ClassNotFoundException e)
        {
            System.out.println(e.getMessage());
        }
        try
        {
            Locale.setDefault(Locale.ENGLISH);
            conexion =
DriverManager.getConnection("jdbc:oracle:thin:@loc
alhost:1521:XE","TESIS","abc123");
            sentencia =
conexion.createStatement(resultset.TYPE_SCROLL
_INSENSITIVE, resultset.CONCUR_READ_ONLY);
            //sentencia = conexion.createStatement();
        }
        catch(SQLException x)
        {
            System.out.println(x.getMessage());
        }
}

public int print(Graphics graphics, PageFormat
pageFormat, int pageIndex) throws PrinterException
{
        Image Croquis;
        Image Cabecera;
        Toolkit tool = Toolkit.getDefaultToolkit();
        //Toolkit tool1 = Toolkit.getDefaultToolkit();
        if (pageIndex == 0)
        {
            Graphics2D g2d = (Graphics2D) graphics;
            g2d.translate(pageFormat.getImageableX(),
pageFormat.getImageableY());

Cabecera=tool.getDefaultToolkit().getImage(ClassL
oader.getSystemResource("Gimagenes/Logo
Imp.png"));
            g2d.drawImage(Cabecera,100,10,this);
            g2d.drawString("Bienvenido al
Parqueadero", 100,100);
            g2d.drawString("Codigo Vehiculo:
"+codVehImp, 100,120);
            g2d.drawString("Su lugar asignado es el:
"+codEspImp, 100,140);
            g2d.drawString("Fecha de asignación:
"+fechaImp, 100,160);
            if(codEspImp.equals("S1A01"))
            {

Croquis=tool.getDefaultToolkit().getImage(ClassLoa
der.getSystemResource("Espacios/S1A01.jpg"));
                g2d.drawImage(Croquis,100,200,this);
            }
            if(codEspImp.equals("S1A02"))
            {

Croquis=tool.getDefaultToolkit().getImage(ClassLoa
der.getSystemResource("Espacios/S1A02.jpg"));
                g2d.drawImage(Croquis,100,200,this);
            }
            return PAGE_EXISTS;
        }
```

```
        else
            return NO_SUCH_PAGE;
    }

    private void
btnAsignarActionPerformed(java.awt.event.ActionEv
ent evt) {
        String codigoVehiculo=codVEH.getText();
        int codRegistro=0;
        String tipoParq="";
        String periodo="";
        int tot=0;
        int cont=0;
        conexion();
        try{
        resultset=sentencia.executeQuery("select
count(CODIGOESPACIO) m from ESPACIOS");
                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
        }catch(SQLException
x){System.out.println(x.getMessage());}
                String[][] Plazas = new String[tot][2];
            try{
                resultset =
sentencia.executeQuery("select
CODIGOESPACIO,DISPONIBILIDAD from
Espacios WHERE DISPONIBILIDAD='"+"Si"+"'");
                 while(resultset.next()){

Plazas[cont][0]=resultset.getString("CODIGOESPAC
IO");

Plazas[cont][1]=resultset.getString("DISPONIBILIDA
D");
                    System.out.println(Plazas[cont][0]);
                    System.out.println(Plazas[cont][1]);
                    cont++;
                }
                resultset =
sentencia.executeQuery("select
TIPOPARQ,PERIODO from VEHICULOS WHERE
CODIGOVEHICULO='"+codigoVehiculo+"'");
                 while(resultset.next())
                 {

tipoParq=resultset.getString("TIPOPARQ");

periodo=resultset.getString("PERIODO");
                 }
                 fecha = new Date();
                 SimpleDateFormat formateador = new
SimpleDateFormat("dd-MM-yyyy-hh:mm:ss");
                 formateador.format(fecha);

if(tipoParq.equals("Nocturno")&&periodo.equals("20
12-1")&&fecha.getHours()>6)
                 {
                    if(Plazas[0][1].equals("Si"))
                    {
                       try{
                          fecha = new Date();

resultset=sentencia.executeQuery("select
count(CODIGOREGISTRO) m from REGISTRO");
                          while(resultset.next()){

codRegistro=Integer.parseInt(resultset.getString("m"
)+1);
                          }

sentencia.executeUpdate("INSERT INTO
Registro(CodigoRegistro,CodigoVehiculo,CodigoEs
pacio,FechaEntrada,FechaSalida,Mes) VALUES
('"+codRegistro+"','"+codigoVehiculo+"','"+Plazas[0][
0]+"','"+fecha+"','"+0000+"','"+fecha.getMonth()+"')");

sentencia.executeUpdate("UPDATE ESPACIOS set
DISPONIBILIDAD='"+"No"+"' where
CODIGOESPACIO='"+Plazas[0][0]+"'");

JOptionPane.showMessageDialog(this, "Espacio
Asignado"+Plazas[0][0], "Confirmación de
Asignación",
JOptionPane.INFORMATION_MESSAGE);

JOptionPane.showMessageDialog(this, "Espacio
Asignado"+Plazas[0][0], "Actualización de Registro",
JOptionPane.INFORMATION_MESSAGE);
                          sentencia.close();
                          conexion.close();
                          codVehImp=codigoVehiculo;
                          codEspImp=Plazas[0][0];
                          fechaImp=""+fecha;
                          PrinterJob printerJob =
PrinterJob.getPrinterJob();
                          printerJob.setPrintable(this);
                          if ( printerJob.printDialog())
                          {
                             try
                                {
                                   printerJob.print();
                                }catch (PrinterException
exception) {

System.err.println("Printing error: " + exception);
                                }
                             }
                          }catch(SQLException x){

JOptionPane.showMessageDialog(null,"No existen
plazas libres!!!, Porfavor Espere","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
codVEH.setText("");
                            System.out.println("Error
es:"+x.getMessage());
                          }
```

```
                    codVEH.setText("");
                }else

{JOptionPane.showMessageDialog(null,"No existen
plazas libres!!!, Porfavor Espere","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
codVEH.setText(""); }
                    sentencia.close();
                    conexion.close();
            }

if(tipoParq.equals("Completo")||tipoParq.equals("Priv
ilegios"))
            {
              if(periodo.equals("2012-1"))
              {
                 if(Plazas[0][1]==null)

JOptionPane.showMessageDialog(null,"No existen
plazas libres!!!, Porfavor Espere","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
codVEH.setText("");
                 if(Plazas[0][1].equals("Si"))
                 {
                    try{
                       fecha = new Date();

resultset=sentencia.executeQuery("select
count(CODIGOREGISTRO) m from REGISTRO");
                       while(resultset.next()){

codRegistro=Integer.parseInt(resultset.getString("m"
));
                       codRegistro=codRegistro+1;
                       }

sentencia.executeUpdate("INSERT INTO
Registro(CodigoRegistro,CodigoVehiculo,CodigoEs
pacio,FechaEntrada,FechaSalida) VALUES
('"+codRegistro+"','"+codigoVehiculo+"','"+Plazas[0][
0]+"','"+fecha+"','"+0000+"')");

sentencia.executeUpdate("UPDATE ESPACIOS set
DISPONIBILIDAD='"+"No"+"' where
CODIGOESPACIO='"+Plazas[0][0]+"'");
                       codVehImp=codigoVehiculo;
                       codEspImp=Plazas[0][0];
                       fechaImp=""+fecha;
                       PrinterJob printerJob =
PrinterJob.getPrinterJob();
                       printerJob.setPrintable(this);
                       if ( printerJob.printDialog())
                       {
                          try
                             {
                                printerJob.print();
                             }catch (PrinterException
exception) {
System.err.println("Printing error: " + exception);
                                }
                             }
                          }catch(SQLException x){

System.out.println(x.getMessage());}
                          codVEH.setText("");
                          sentencia.close();
                          conexion.close();
                    }
                    if(Plazas[0][1].equals("No"))
                    {

JOptionPane.showMessageDialog(null,"No existen
plazas libres!!!, Porfavor Espere","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
codVEH.setText(""); }
                    }
                    if(periodo.equals("2012-1")==false)
                    {

JOptionPane.showMessageDialog(null,"Su vehiculo
no esta en periodo vigente consulte con el
Administrador!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
codVEH.setText(""); }
                 }
                }catch(SQLException
x){System.out.println(x.getMessage());}

}

   private void
btnSalirActionPerformed(java.awt.event.ActionEvent
evt) {
      String codigoVehiculo=codVEH.getText();
      String codRegistro="";
      String codigoEspacio="";
      conexion();
      try{
            resultset =
sentencia.executeQuery("select
CODIGOREGISTRO,CODIGOESPACIO from
REGISTRO WHERE
CODIGOVEHICULO='"+codigoVehiculo+"' order by
CODIGOREGISTRO asc");
            while(resultset.next())
            {
               if(resultset.isLast())

codRegistro=resultset.getString("CODIGOREGISTR
O");

codigoEspacio=resultset.getString("CODIGOESPAC
IO");
            }
            fecha = new Date();
```

```
            SimpleDateFormat formateador = new
SimpleDateFormat("dd-MM-yyyy-hh:mm:ss");
            formateador.format(fecha);
            sentencia.executeUpdate("UPDATE
REGISTRO set FECHASALIDA='"+fecha+"' where
CODIGOREGISTRO='"+codRegistro+"'");
            sentencia.executeUpdate("UPDATE
ESPACIOS set DISPONIBILIDAD='"+"Si"+"' where
CODIGOESPACIO='"+codigoEspacio+"'");
            codVEH.setText("");
            sentencia.close();
            conexion.close();
            JOptionPane.showMessageDialog(this,
"Espacio Liberado Gracias por Venir", "Confirmación
Salida", JOptionPane.INFORMATION_MESSAGE);
            }catch(SQLException x){
              System.out.println(x.getMessage());}
    }


    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new
Runnable() {
            public void run() {
                new AsignarGUI().setVisible(true);
            }
        });
    }
```

Empaquetado: GUI **Componente:** ConsultarEspacioGUI.java

```
package gui;
import dp.EspaciosDP;
import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import tesis.FondoJava;
/**
 *
 * @author  Juanka
 */
public class ConsultaEspaciosGUI extends
javax.swing.JFrame {
    FondoJava as= new FondoJava(5);
    /** Creates new form consultaServicios */
    public ConsultaEspaciosGUI() {

        initComponents();
        codConsulta.setEnabled(false);
        this.add(as,BorderLayout.CENTER);
        codConsultaPAR.setEnabled(false);
        as.repaint();
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }
boolean band=false;
    private void
consultarPROActionPerformed(java.awt.event.Actio
nEvent evt) {

        if(consulta.getSelectedIndex()==0){

            EspaciosDP esp=new EspaciosDP();
            EspaciosDP[]
espa=esp.verServiciosDP("",consulta.getSelectedIn
dex());
            String[][] elem=new String[espa.length][4];
            for(int i=0;i<espa.length;i++) {
                elem[i][0]=espa[i].getCodigoEspacio();

elem[i][1]=espa[i].getCodigoParqueadero();
                elem[i][2]=espa[i].getDescripcion();
                elem[i][3]=espa[i].getDisponibilidad();
            }

            String[] headers = {"Codigo
Espacio","Codigo
Parqueadero","Descripción","Disponibilidad"};
            DefaultTableModel model = new
DefaultTableModel(elem, headers);
            jTable1 = new JTable(model);
            jScrollPane1.setViewportView(jTable1);
            } else
            if(consulta.getSelectedIndex()==1)
            {
            codConsulta.setEditable(true);
            String cod=codConsulta.getText();
            EspaciosDP serv=new EspaciosDP(cod);
            band=false;
            band=serv.verificarDP();
           if(band==true) {
            JOptionPane.showMessageDialog(this,
"Registro Encontrado", "Actualización de Registro",
JOptionPane.INFORMATION_MESSAGE);
            this.repaint();
            String var=codConsulta.getText();
            EspaciosDP esp=new EspaciosDP();
            EspaciosDP[]
espa=esp.verServiciosDP(var,consulta.getSelectedI
ndex());
            String[][] elem=new String[espa.length][4];
            for(int i=0;i<espa.length;i++) {
                elem[i][0]=espa[i].getCodigoEspacio();

elem[i][1]=espa[i].getCodigoParqueadero();
                elem[i][2]=espa[i].getDescripcion();
                elem[i][3]=espa[i].getDisponibilidad();
            }
            String[] headers = {"Codigo
Espacio","Codigo
Parqueadero","Descripción","Disponibilidad"};
            DefaultTableModel model = new
DefaultTableModel(elem, headers);
            jTable1 = new JTable(model);
            jScrollPane1.setViewportView(jTable1);
            }else
            {JOptionPane.showMessageDialog(this,
"Registro No encontrado", "Actualización de
Registro",
JOptionPane.INFORMATION_MESSAGE);
            this.repaint();
            }
            }
}

    private void
CancelarActionPerformed(java.awt.event.ActionEve
nt evt) {
        this.dispose();
}

    private void
consultaActionPerformed(java.awt.event.ActionEven
t evt) {
```

```
String con=consulta.getSelectedItem().toString();
if(con.equals("Parámetro"))
{
    codConsulta.setEnabled(true);
    codConsulta.setEditable(true);
}
 else
 {
    codConsulta.setEnabled(false);
    codConsulta.setText("");
 }
 }
```

```
    /**
     *
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new
Runnable() {
            public void run() {
                new
ConsultaEspaciosGUI().setVisible(true);
            }
        });
    }
```

Empaquetado: GUI **Componente:** ConsultarRegistrosGUI.java

```
package gui;
import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import java.sql.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import tesis.FondoJava;
/**
 *
 * @author  Juanka
 */
public class ConsultaRegistrosGUI extends
javax.swing.JFrame {
    FondoJava as= new FondoJava(6);
    private Connection conexion;
    private Statement sentencia;
    private ResultSet resultset;
    /** Creates new form consultaServicios */
    public ConsultaRegistrosGUI() {
        initComponents();
        this.add(as,BorderLayout.CENTER);
        as.repaint();
        codConsulta.setEnabled(false);
        codConsultaPLA.setEnabled(false);
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }
boolean band=false;
public void conexion(){
try
        {
            Class.forName
("oracle.jdbc.driver.OracleDriver");
        }
        catch(ClassNotFoundException e)
        {
            System.out.println(e.getMessage());
        }
        try
        {
            Locale.setDefault(Locale.ENGLISH);
            conexion =
DriverManager.getConnection("jdbc:oracle:thin:@loc
alhost:1521:XE","TESIS","abc123");
            //sentencia =
conexion.createStatement(resultset.TYPE_SCROLL
_INSENSITIVE, resultset.CONCUR_READ_ONLY);
            sentencia = conexion.createStatement();
        }
        catch(SQLException x)
        {
            System.out.println(x.getMessage());
        }
}

public boolean verificarExistencia(String Cod){
        boolean band=false;

        try {
            resultset=sentencia.executeQuery("select *
from REGISTRO where
CODIGOVEHICULO='"+Cod+"'");
            while(resultset.next())
            {
                band= true;
            }
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
        return band;
    }

    private void
consultarPROActionPerformed(java.awt.event.Actio
nEvent evt) {
        if(consulta.getSelectedIndex()==0){
            int tot=0;
            int cont=0;
            conexion();
            try{
                resultset=sentencia.executeQuery("select
count(CODIGOREGISTRO) m from REGISTRO");
                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
            }catch(SQLException
x){System.out.println(x.getMessage());}
            String[][] Registros = new String[tot][5];
            try{
                resultset =
sentencia.executeQuery("select * from Registro
order by CODIGOREGISTRO asc");
                  while(resultset.next()){

Registros[cont][0]=resultset.getString("CODIGOREG
ISTRO");

Registros[cont][1]=resultset.getString("CODIGOVEH
ICULO");

Registros[cont][2]=resultset.getString("CODIGOESP
ACIO");
```

```
Registros[cont][3]=resultset.getString("FECHAENTR
ADA");

Registros[cont][4]=resultset.getString("FECHASALI
DA");
                    cont++;
                 }
                }catch(SQLException
x){System.out.println(x.getMessage());}
          String[] headers = {"Codigo
Registro","Codigo Vehiculo","Espacio","Fecha
Entrada","Fecha Salida"};
          DefaultTableModel model = new
DefaultTableModel(Registros, headers);
          jTable1 = new JTable(model);
          jScrollPane1.setViewportView(jTable1);
          } else
          if(consulta.getSelectedIndex()==1)
          {
          codConsulta.setEditable(true);
          String cod=codConsulta.getText();
          band=false;
          band=verificarExistencia(cod);
         if(band==true) {
          JOptionPane.showMessageDialog(this,
"Registro(s) Encontrado(s)", "Actualización de
Registro",
JOptionPane.INFORMATION_MESSAGE);
          this.repaint();
          String var=codConsulta.getText();
          int tot=0;
          int cont=0;
          conexion();
          try{
             resultset=sentencia.executeQuery("select
count(CODIGOREGISTRO) m from REGISTRO
where CODIGOVEHICULO='"+var+"'");
             while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
             }
          }catch(SQLException
x){System.out.println(x.getMessage());}
          String[][] Registros = new String[tot][5];
          try{
             resultset =
sentencia.executeQuery("select * from Registro
where CODIGOVEHICULO='"+var+"' order by
CODIGOREGISTRO asc");
               while(resultset.next()){

Registros[cont][0]=resultset.getString("CODIGOREG
ISTRO");

Registros[cont][1]=resultset.getString("CODIGOVEH
ICULO");

Registros[cont][2]=resultset.getString("CODIGOESP
ACIO");

Registros[cont][3]=resultset.getString("FECHAENTR
ADA");

Registros[cont][4]=resultset.getString("FECHASALI
DA");
                    cont++;
                 }
                }catch(SQLException
x){System.out.println(x.getMessage());}
          String[] headers = {"Codigo
Registro","Codigo Vehiculo","Espacio","Fecha
Entrada","Fecha Salida"};
          DefaultTableModel model = new
DefaultTableModel(Registros, headers);
          jTable1 = new JTable(model);
          jScrollPane1.setViewportView(jTable1);
          }else
          {JOptionPane.showMessageDialog(this,
"Registro No encontrado", "Actualización de
Registro",
JOptionPane.INFORMATION_MESSAGE);
          this.repaint();
          }
          }
}

   private void
CancelarActionPerformed(java.awt.event.ActionEve
nt evt) {
       this.dispose();
}

   private void
consultaActionPerformed(java.awt.event.ActionEven
t evt) {
    String con=consulta.getSelectedItem().toString();
  if(con.equals("Parámetro"))
  {
     codConsulta.setEnabled(true);
     codConsulta.setEditable(true);
  }
   else
   {
     codConsulta.setEnabled(false);
     codConsulta.setText("");
   }
   }

private void
consultaFocusGained(java.awt.event.FocusEvent
evt) {

}
```

```
private void
consultaMouseClicked(java.awt.event.MouseEvent
evt) {

}

    /**
     *
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new
Runnable() {
            public void run() {
                new
ConsultaRegistrosGUI().setVisible(true);
            }
        });
    }
```

Empaquetado: GUI **Componente:** ConsultarUsuarioGUI.java

```
package gui;
import dp.UsuarioDP;
import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import tesis.FondoJava;
/**
 *
 * @author Juanka
 */
public class ConsultaUsuarioGUI extends
javax.swing.JFrame {
    FondoJava as= new FondoJava(3);
    /** Creates new form ConsultaUsuarioGUI */
    public ConsultaUsuarioGUI() {
        initComponents();
        this.add(as,BorderLayout.CENTER);
        as.repaint();
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }
private void
CancelarActionPerformed(java.awt.event.ActionEve
nt evt) {
        this.dispose();
}

private void
consultarActionPerformed(java.awt.event.ActionEve
nt evt) {

            UsuarioDP registro=new UsuarioDP();
            UsuarioDP[] arr=registro.verServiciosDP();
            String[][] mat=new String[arr.length][4];
            for(int i=0;i<arr.length;i++) {
                mat[i][0]=arr[i].getCodigoUsuario();
                mat[i][1]=arr[i].getUserUsuario();
                mat[i][2]=arr[i].getPasswordUsuario();
                mat[i][3]=arr[i].getTipoUsuario();
            }
            String[] headers =
{"Codigo","Usuario","Password","Tipo"};
            DefaultTableModel model = new
DefaultTableModel(mat, headers);
            jTable1 = new JTable(model);
            jScrollPane1.setViewportView(jTable1);
}
    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc="
Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not
available, stay with the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lo
okandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo
info :
javax.swing.UIManager.getInstalledLookAndFeels())
{
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getCla
ssName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(ConsultaUsuario
GUI.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(ConsultaUsuario
GUI.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(ConsultaUsuario
GUI.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch
(javax.swing.UnsupportedLookAndFeelException
ex) {

java.util.logging.Logger.getLogger(ConsultaUsuario
GUI.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new
Runnable() {

            public void run() {
```

```
            new                                                    });
ConsultaUsuarioGUI().setVisible(true);                    }
        }
```

Empaquetado: GUI **Componente:** ConsultarUsuarioGUI.java

```
package gui;
import dp.VehiculoDP;
import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import tesis.FondoJava;
/**
 *
 * @author  Juanka
 */
public class ConsultaVehiculoGUI extends
javax.swing.JFrame {
    FondoJava as= new FondoJava(5);
    /** Creates new form consultaServicios */
    public ConsultaVehiculoGUI() {
        initComponents();
        this.add(as,BorderLayout.CENTER);
        as.repaint();
        codConsulta.setEditable(false);
        codConsultaPLA.setEditable(false);
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }
boolean band=false;
    private void
consultarPROActionPerformed(java.awt.event.Actio
nEvent evt) {

        if(consulta.getSelectedIndex()==0){

            VehiculoDP usu=new VehiculoDP();
            VehiculoDP[]
usua=usu.verServiciosDP("",consulta.getSelectedIn
dex());
            String[][] elem=new String[usua.length][10];
            for(int i=0;i<usua.length;i++) {
                elem[i][0]=usua[i].getCodigoVehiculo();

elem[i][1]=usua[i].getCodigoParqueaderoVehiculo();

elem[i][2]=usua[i].getResponsableVehiculo();
                elem[i][3]=usua[i].getMarcaVehiculo();
                elem[i][4]=usua[i].getFechaFabVehiculo();
                elem[i][5]=usua[i].getPlacaVehiculo();
                elem[i][6]=usua[i].getColorVehiculo();
                elem[i][7]=usua[i].getModeloVehiculo();
                elem[i][8]=usua[i].getTipoParqVehiculo();
                elem[i][9]=usua[i].getPeriodoVehiculo();
            }

            String[] headers = {"Codigo
Vehiculo","Codigo
Parqueadero","Responsable","Marca","Fecha
Fabricación","Placa","Color","Modelo","Tipo de
Parqueo","Periodo"};
            DefaultTableModel model = new
DefaultTableModel(elem, headers);
            jTable1 = new JTable(model);
            jScrollPane1.setViewportView(jTable1);
            } else
            if(consulta.getSelectedIndex()==1)
            {
            codConsulta.setEditable(true);
            String cod=codConsulta.getText();
            VehiculoDP serv=new VehiculoDP(cod);
            band=false;
            band=serv.verificarDP();
           if(band==true) {
            JOptionPane.showMessageDialog(this,
"Registro Encontrado", "Actualización de Registro",
JOptionPane.INFORMATION_MESSAGE);
            this.repaint();
            String var=codConsulta.getText();
            VehiculoDP usu=new VehiculoDP();
            VehiculoDP[]
usua=usu.verServiciosDP(var,consulta.getSelectedI
ndex());
            String[][] elem=new String[usua.length][10];
            for(int i=0;i<usua.length;i++) {
                elem[i][0]=usua[i].getCodigoVehiculo();

elem[i][1]=usua[i].getCodigoParqueaderoVehiculo();

elem[i][2]=usua[i].getResponsableVehiculo();
                elem[i][3]=usua[i].getMarcaVehiculo();
                elem[i][4]=usua[i].getFechaFabVehiculo();
                elem[i][5]=usua[i].getPlacaVehiculo();
                elem[i][6]=usua[i].getColorVehiculo();
                elem[i][7]=usua[i].getModeloVehiculo();
                elem[i][8]=usua[i].getTipoParqVehiculo();
                elem[i][9]=usua[i].getPeriodoVehiculo();
            }
            String[] headers = {"Codigo
Vehiculo","Codigo
Parqueadero","Responsable","Marca","Fecha
Fabricación","Placa","Color","Modelo","Tipo de
Parqueo","Periodo"};
            DefaultTableModel model = new
DefaultTableModel(elem, headers);
            jTable1 = new JTable(model);
            jScrollPane1.setViewportView(jTable1);
            }else
```

```
        {JOptionPane.showMessageDialog(this,
"Registro No encontrado", "Actualización de
Registro",
JOptionPane.INFORMATION_MESSAGE);
        this.repaint();
        }
        }
}

    private void
CancelarActionPerformed(java.awt.event.ActionEve
nt evt) {
       this.dispose();
}

    private void
consultaActionPerformed(java.awt.event.ActionEven
t evt) {
   String con=consulta.getSelectedItem().toString();
   if(con.equals("Parámetro"))
   {
      codConsulta.setEnabled(true);
      codConsulta.setEditable(true);
   }
    else
    {
      codConsulta.setEnabled(false);
      codConsulta.setText("");
    }
    }

    /**
     *
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new
Runnable() {
            public void run() {
                new
ConsultaVehiculoGUI().setVisible(true);
            }
        });
    }
```

Empaquetado: GUI **Componente:** EspaciosGUI.java

```
package gui;
import dp.EspaciosDP;
import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import java.awt.event.KeyEvent;
import java.text.DecimalFormat;
import javax.swing.JOptionPane;
import java.util.*;
import tesis.FondoJava;
/**
 *
 * @author  Juanka
 */
public class EspaciosGUI extends
javax.swing.JFrame {
     FondoJava as= new FondoJava(3);
     EspaciosDP espacioDP = new EspaciosDP();
     int existe=0, noexec=0;
     private HashMap codparq=new HashMap();

   public EspaciosGUI() {
      initComponents();
      this.add(as,BorderLayout.CENTER);
      as.repaint();
      codparq =
espacioDP.getHashParqueaderoDP();
      cargarMap(codparq);
   }

@Override
   public Image getIconImage() {
      Toolkit tool = Toolkit.getDefaultToolkit();
      Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
      return retValue;
   }

    public void cargarMap(HashMap map){
      int cont=0;
      String[] parqueaderos=new String[map.size()];
      Set set = map.entrySet();
      Iterator i = set.iterator();
      while(i.hasNext()){
      Map.Entry me = (Map.Entry)i.next();
      parqueaderos[cont]=(String)me.getKey();
      cont++;
      }
      for(int a=0;a<parqueaderos.length;a++)

comboParqueaderos.addItem(parqueaderos[a]);
   }
```

```
boolean band=false;
   private void
guardarPROActionPerformed(java.awt.event.Action
Event evt) {
       String codigo=codESP.getText();
       String
codigoParqueadero=(String)comboParqueaderos.ge
tSelectedItem();
       String descripcion=descrip.getText();
       String
disponibilidad=(String)disp.getSelectedItem();
       EspaciosDP serv=new
EspaciosDP(codigo,codigoParqueadero,descripcion,
disponibilidad);
       band=serv.verificarDP();

       if(band==true)
       {  int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
modificar el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
          if (seleccion == 0)
          {
          serv.modificarDP();
          codESP.setText("");
          descrip.setText("");
        } }  else
          {
             int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Ingresar el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
          if (seleccion == 0){
          serv.ingresarDP();
          codESP.setEditable(true);
          descrip.setEditable(false);
          codESP.setText("");
          descrip.setText("");
          guardarPRO.setEnabled(false);
          JOptionPane.showMessageDialog(this,
"Espacio Ingresado con exito!!", "Ingreso de
Registro",
JOptionPane.INFORMATION_MESSAGE);
          this.repaint();
       }}
   }

   private void
codESPActionPerformed(java.awt.event.ActionEven
t evt) {


   }

   private void
cancelarPROActionPerformed(java.awt.event.Action
Event evt) {
       this.dispose();
   }

   private void
codESPFocusLost(java.awt.event.FocusEvent evt) {

       String cod=codESP.getText();
       band=false;
       EspaciosDP serv=new EspaciosDP(cod);
       band=serv.verificarDP();
       if(cod.equals(""))
       {

JOptionPane.showMessageDialog(null,"Debe
ingresar un código!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
       }else
       if(band==true) {
            int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Modificar el registro?","Modificar
Espacio",JOptionPane.YES_NO_CANCEL_OPTION
,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
          if (seleccion == 0)
          {
          codESP.setEditable(false);
          descrip.setEditable(true);
          descrip.setText(serv.verDatosDP()[2]);
          String disp1=""+disp.getSelectedItem();
          String disp2=serv.verDatosDP()[6];
          if(disp1.equals(disp2))
          {
             disp.removeAllItems();
             disp.addItem("Si");
             disp.addItem("No");
             this.repaint();
          }
          else
          {
                disp.removeAllItems();
                disp.addItem("No");
                disp.addItem("Si");
                this.repaint();
             }
          this.repaint();
          guardarPRO.setEnabled(true);
          borrarPRO.setEnabled(true);

          }
         } else
```

```
        {  int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Ingresar nuevo registro?","Modificar
Espacio",JOptionPane.YES_NO_CANCEL_OPTION
,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
            if (seleccion == 0)
            {
            guardarPRO.setEnabled(true);
            descrip.setEditable(true);
            codESP.setEditable(false);
            descrip.setText("");
            }
        }


    }

    private void
borrarPROActionPerformed(java.awt.event.ActionEv
ent evt) {
        String cod=codESP.getText();
        EspaciosDP serv=new EspaciosDP(cod);
        int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea Borrar
el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
            if(seleccion == 0)
            {
            serv.eliminarDP();
            codESP.setEditable(true);
            codESP.setText("");
            descrip.setText("");
            descrip.setEditable(false);
            guardarPRO.setEnabled(false);
            borrarPRO.setEnabled(false);
        }
    }

    private void
codESPKeyTyped(java.awt.event.KeyEvent evt) {
        int limite = 6;
        if (codESP.getText().length()== limite){
            evt.consume();

JOptionPane.showMessageDialog(null,"Escriba
máximo 6 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
        }
    }

    private void
descripKeyTyped(java.awt.event.KeyEvent evt) {
        int limite = 25;
        if (descrip.getText().length()== limite)
        {  evt.consume();
        JOptionPane.showMessageDialog(null,"Escriba
máximo 25 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
        }
    }

private void
comboParqueaderosActionPerformed(java.awt.even
t.ActionEvent evt) {
// TODO add your handling code here:
}
     public void cargarDatos() {

codESP.setText(espacioDP.getCodigoEspacio());
         descrip.setText(espacioDP.getDescripcion());
         codESP.setEditable(false);
         guardarPRO.setEnabled(true);
     }
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new
Runnable() {
            public void run() {
                new EspaciosGUI().setVisible(true);
            }
        });
    }
```

Empaquetado: GUI **Componente:** ParqueaderoGUI.java

```
package gui;
import dp.ParqueaderoDP;
import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import java.awt.event.KeyEvent;
import java.text.DecimalFormat;
import javax.swing.JOptionPane;
import java.util.*;
import tesis.FondoJava;
/**
 *
 * @author  Juanka
 */
public class ParqueaderoGUI extends
javax.swing.JFrame {
    FondoJava as= new FondoJava(4);
    ParqueaderoDP parqueaderoDP = new
ParqueaderoDP();
    int existe=0, noexec=0;

   /** Creates new form VehiculoGUI */
   public ParqueaderoGUI() {
      initComponents();
      this.add(as,BorderLayout.CENTER);
      as.repaint();
   }

@Override
   public Image getIconImage() {
      Toolkit tool = Toolkit.getDefaultToolkit();
      Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
      return retValue;
   }
boolean band=false;
   private void
guardarPROActionPerformed(java.awt.event.Action
Event evt) {
      String codigoParqueadero=codPAR.getText();
      String descripcion=descripPAR.getText();
      ParqueaderoDP serv=new
ParqueaderoDP(codigoParqueadero,descripcion);
      band=serv.verificarDP();

      if(band==true)
      {  int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
modificar el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
         if (seleccion == 0)
         {
         serv.modificarDP();
         codPAR.setText("");
         descripPAR.setText("");
      } }  else
      {   int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Ingresar el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
         if (seleccion == 0)
         {
         serv.ingresarDP();
         codPAR.setEditable(true);
         descripPAR.setEditable(false);
         codPAR.setText("");
         descripPAR.setText("");
         guardarPRO.setEnabled(false);
         JOptionPane.showMessageDialog(this,
"Parqueadero Ingresado con exito!!", "Ingreso de
Registro",
JOptionPane.INFORMATION_MESSAGE);
         this.repaint();
      }}
   }

   private void
codPARActionPerformed(java.awt.event.ActionEven
t evt) {


   }

   private void
cancelarPROActionPerformed(java.awt.event.Action
Event evt) {
      this.dispose();
   }

   private void
codPARFocusLost(java.awt.event.FocusEvent evt) {

      String cod=codPAR.getText();
      band=false;
      ParqueaderoDP serv=new
ParqueaderoDP(cod);
      band=serv.verificarDP();
      if(cod.equals(""))
      {

JOptionPane.showMessageDialog(null,"Debe
ingresar un código!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
      }else
      if(band==true) {
          int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
```

```
Modificar el registro?","Modificar
Parqueadero",JOptionPane.YES_NO_CANCEL_OP
TION,JOptionPane.QUESTION_MESSAGE,null,ne
w Object[] { "Si", "No", "Cancelar" }, "No");
          if (seleccion == 0)
          {
          codPAR.setEditable(false);
          descripPAR.setEditable(true);
          descripPAR.setText(serv.verDatosDP()[1]);
          }
          guardarPRO.setEnabled(true);
          borrarPRO.setEnabled(true);

        } else
       {  int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Ingresar nuevo registro?","Ingresar
Parqueadero",JOptionPane.YES_NO_CANCEL_OP
TION,JOptionPane.QUESTION_MESSAGE,null,ne
w Object[] { "Si", "No", "Cancelar" }, "No");
          if (seleccion == 0)
          {
          guardarPRO.setEnabled(true);
          descripPAR.setEditable(true);
          codPAR.setEditable(false);
          descripPAR.setText("");
       }}



   }

   private void
borrarPROActionPerformed(java.awt.event.ActionEv
ent evt) {
       String cod=codPAR.getText();
       ParqueaderoDP serv=new
ParqueaderoDP(cod);
       int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea Borrar
el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
          if (seleccion == 0)
          {
          serv.eliminarDP();
          codPAR.setEditable(true);
          codPAR.setText("");
          descripPAR.setText("");
          descripPAR.setEditable(false);
          guardarPRO.setEnabled(false);
          borrarPRO.setEnabled(false);
       }
   }

   private void
codPARKeyTyped(java.awt.event.KeyEvent evt) {
       int limite = 25;
       if (codPAR.getText().length()== limite){
          evt.consume();

JOptionPane.showMessageDialog(null,"Escriba
máximo 25 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
       }
   }

   private void
descripPARKeyTyped(java.awt.event.KeyEvent evt)
{
      int limite = 25;
       if (descripPAR.getText().length()== limite)
       {  evt.consume();
       JOptionPane.showMessageDialog(null,"Escriba
máximo 25 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
       }
   }

   public static void main(String args[]) {
       java.awt.EventQueue.invokeLater(new
Runnable() {
          public void run() {
             new ParqueaderoGUI().setVisible(true);
          }
       });
   }
```

Empaquetado: GUI **Componente:** ReporteGUI.java

```
package gui;
import java.sql.*;
import java.text.SimpleDateFormat;
import java.util.*;
import java.util.Date;
import javax.swing.JOptionPane;
import java.awt.*;
import java.awt.print.*;
import java.text.DecimalFormat;
import md.VerificarMD;
import tesis.FondoJava;

/**
 *
 * @author  Juanka
 */
public class ReporteGUI extends
javax.swing.JFrame implements Printable{
    FondoJava as= new FondoJava(8);
    private Connection conexion;
    private Statement sentencia;
    private ResultSet resultset;
    String codVehImp;
    String codEspImp;
    String fechaImp;
    Date fecha = new Date();

    /** Creates new form FacturacionGUI */
    public ReporteGUI() {
        initComponents();
        this.add(as,BorderLayout.CENTER);
        as.repaint();
    }
private void
btnCancelarActionPerformed(java.awt.event.ActionE
vent evt) {
    this.dispose();
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }

public void conexion(){
try
        {
            Class.forName
("oracle.jdbc.driver.OracleDriver");
        }
        catch(ClassNotFoundException e)
        {
            System.out.println(e.getMessage());
        }
        try
        {
            Locale.setDefault(Locale.ENGLISH);
            conexion =
DriverManager.getConnection("jdbc:oracle:thin:@loc
alhost:1521:XE","TESIS","abc123");
            sentencia =
conexion.createStatement(resultset.TYPE_SCROLL
_INSENSITIVE, resultset.CONCUR_READ_ONLY);
            //sentencia = conexion.createStatement();
        }
        catch(SQLException x)
        {
            //System.out.println(x.getMessage());
        }
}

public int print(Graphics graphics, PageFormat
pageFormat, int pageIndex) throws PrinterException
{
        Image Croquis;
        Image Cabecera;
        Toolkit tool = Toolkit.getDefaultToolkit();
        //Toolkit tool1 = Toolkit.getDefaultToolkit();
        if (pageIndex == 0)
        {
            Graphics2D g2d = (Graphics2D) graphics;
            g2d.translate(pageFormat.getImageableX(),
pageFormat.getImageableY());

Cabecera=tool.getDefaultToolkit().getImage(ClassL
oader.getSystemResource("Gimagenes/Logo
Imp.png"));
            g2d.drawImage(Cabecera,100,10,this);
            g2d.drawString("Bienvenido al
Parqueadero", 100,100);
            g2d.drawString("Codigo Vehiculo:
"+codVehImp, 100,120);
            g2d.drawString("Su lugar asignado es el:
"+codEspImp, 100,140);
            g2d.drawString("Fecha de asignación:
"+fechaImp, 100,160);
            if(codEspImp.equals("S1A01"))
            {

Croquis=tool.getDefaultToolkit().getImage(ClassLoa
der.getSystemResource("Espacios/S1A01.jpg"));
                g2d.drawImage(Croquis,100,200,this);
            }
            if(codEspImp.equals("S1A02"))
            {

Croquis=tool.getDefaultToolkit().getImage(ClassLoa
der.getSystemResource("Espacios/S1A02.jpg"));
```

```
            g2d.drawImage(Croquis,100,200,this);
          }
          return PAGE_EXISTS;
       }
       else
          return NO_SUCH_PAGE;
    }

    private void
btnAsignarActionPerformed(java.awt.event.ActionEv
ent evt) {
       String mesI="";
       switch(mes.getSelectedIndex())
       {
          case 0: mesI="Jan";
          break;
          case 1: mesI="Feb";
          break;
          case 2: mesI="Mar";
          break;
          case 3: mesI="Apr";
          break;
          default:
          break;
       }
       String horaI="
"+(String)HoraI.getSelectedItem();
       int tot=0;
       int cont=0;
       conexion();
       VerificarMD v=new VerificarMD();
       int
mesC=v.verificarMes(mes.getSelectedIndex()+1);
       try{
       resultset=sentencia.executeQuery("select
count(CODIGOREGISTRO) m from REGISTRO");
             while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
             }
       }catch(SQLException x){}
             String[][] Registros = new String[tot][2];
             int c=0;
             cont=0;
          try{
             resultset =
sentencia.executeQuery("select
CODIGOREGISTRO,FECHAENTRADA from
REGISTRO where FECHAENTRADA
LIKE'%"+mesI+"%' AND FECHAENTRADA
LIKE'%"+horaI+"%'");
                while(resultset.next()){

Registros[cont][0]=resultset.getString("CODIGOREG
ISTRO");

Registros[cont][1]=resultset.getString("FECHAENTR
ADA");
                   c++;
                   cont++;
                }
             sentencia.close();
             conexion.close();
          }catch(SQLException x){}
          System.out.println("El numero de vehiculos
en mes de "+(String)mes.getSelectedItem()+" es de:
"+c);
          System.out.println("El numero de registros
totales es de: "+tot);
          System.out.println("El numero de registros
del mes es: "+mesC);
          DecimalFormat df = new
DecimalFormat("0.00");

MesM.setText((String)mes.getSelectedItem());
          HoraM.setText(horaI);
          double porcM=(double)c/mesC*100;
          PorcM.setText(df.format(porcM));

MesT.setText((String)mes.getSelectedItem());
          double porcT=(double)mesC/tot*100;
          PorcT.setText(df.format(porcT));
          System.out.println("El valor es de: "+porcT);
}

    public static void main(String args[]) {
       java.awt.EventQueue.invokeLater(new
Runnable() {
          public void run() {
             new ReporteGUI().setVisible(true);
          }
       });
    }
```

Empaquetado: GUI **Componente:** UsuarioGUI.java

```
package gui;
import dp.UsuarioDP;
import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import javax.swing.JOptionPane;
import tesis.FondoJava;


/**
 *
 * @author Juanka
 */
public class UsuarioGUI extends
javax.swing.JFrame {
    UsuarioDP usuarioDP = new UsuarioDP();
    FondoJava as= new FondoJava(3);
    boolean band=false;
    int existe=0, noexec=0;
    public UsuarioGUI() {
        initComponents();
        this.add(as,BorderLayout.CENTER);
        as.repaint();
        userTxt.setEditable(false);
        passwordTxt.setEditable(false);
        guardar.setEnabled(false);
        borrar.setEnabled(false);
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }
private void
cancelarActionPerformed(java.awt.event.ActionEven
t evt) {
    this.dispose();
}

private void
guardarActionPerformed(java.awt.event.ActionEvent
evt) {
        String codigo=codigoTxt.getText();
        String user=userTxt.getText();
        String password=passwordTxt.getText();
        String tipo="";

if(comboTipo.getSelectedItem()=="Administrador")
            tipo="Administrador";
        if(comboTipo.getSelectedItem()=="Guardia")
            tipo="Guardia";
        UsuarioDP registro=new
UsuarioDP(codigo,user,password,tipo);
        band=registro.verificarDP();
        if(band==true)
        {
            int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
modificar el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
            if (seleccion == 0)
            {
            registro.modificarDP();
            codigoTxt.setText("");
            userTxt.setText("");
            passwordTxt.setText("");
            codigoTxt.setEditable(false);
        } }
        else{
            int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Ingresar el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
            if (seleccion == 0)
            {
            registro.ingresarDP();
            codigoTxt.setText("");
            userTxt.setText("");
            passwordTxt.setText("");
            JOptionPane.showMessageDialog(this,
"Producto Ingresado con exito!!", "Ingreso de
Registro",
JOptionPane.INFORMATION_MESSAGE);
            this.repaint();
        } }
        codigoTxt.setEditable(true);
}

private void
borrarActionPerformed(java.awt.event.ActionEvent
evt) {
    int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea Borrar
el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
            if (seleccion == 0)
            {
                String codigo=codigoTxt.getText();
                UsuarioDP registro=new
UsuarioDP(codigo);
                registro.eliminarDP();
```

```
            codigoTxt.setText("");
            userTxt.setText("");
            passwordTxt.setText("");
            codigoTxt.setEditable(true);
            guardar.setEnabled(false);
            borrar.setEnabled(false);
        }
}

private void
codigoTxtFocusLost(java.awt.event.FocusEvent evt)
{
      String codigo=codigoTxt.getText();
      band=false;
      UsuarioDP registro=new UsuarioDP(codigo);
      band=registro.verificarDP();
      if(codigo.equals(""))
      {

JOptionPane.showMessageDialog(null,"Debe
ingresar un código!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
      }else
      if(band==true) {
        int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Modificar el registro?","Modificar
Producto",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
        if (seleccion == 0)
        {
        codigoTxt.setEditable(false);
        userTxt.setEditable(true);
        passwordTxt.setEditable(true);

        codigoTxt.setText(registro.verDatosDP()[0]);
        userTxt.setText(registro.verDatosDP()[1]);

passwordTxt.setText(registro.verDatosDP()[2]);
        String
tipo1=""+comboTipo.getSelectedItem();
        String tipo2=registro.verDatosDP()[3];
        if(tipo1.equals(tipo2))
        {
           comboTipo.removeAllItems();
           comboTipo.addItem("Administrador");
           comboTipo.addItem("Guardia");
           this.repaint();
        }
        else
        {
             comboTipo.removeAllItems();
             comboTipo.addItem("Guardia");
             comboTipo.addItem("Administrador");
             this.repaint();
           }
        this.repaint();
        guardar.setEnabled(true);
            borrar.setEnabled(true);

      } } else
      { JOptionPane.showMessageDialog(this, "El
usuario no existe", "Actualización de Registro",
JOptionPane.INFORMATION_MESSAGE);
            this.repaint();
            guardar.setEnabled(true);
            borrar.setEnabled(false);
            codigoTxt.setEditable(false);
            userTxt.setEditable(true);
            passwordTxt.setEditable(true);
            userTxt.setText("");
            passwordTxt.setText("");
      }
}

private void
codigoTxtKeyTyped(java.awt.event.KeyEvent evt) {
      int limite = 5;
      if (codigoTxt.getText().length()== limite)
      {  evt.consume();
      JOptionPane.showMessageDialog(null,"Escriba
máximo 5 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
      }
}

private void
userTxtKeyTyped(java.awt.event.KeyEvent evt) {
int limite = 15;
      if (userTxt.getText().length()== limite)
      {  evt.consume();
      JOptionPane.showMessageDialog(null,"Escriba
máximo 15 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
      }
}

private void
passwordTxtKeyTyped(java.awt.event.KeyEvent
evt) {
      int limite = 10;
      if (passwordTxt.getText().length()== limite)
      {  evt.consume();
      JOptionPane.showMessageDialog(null,"Escriba
máximo 10 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
      }
}

   public void cargarDatos() {

codigoTxt.setText(usuarioDP.getCodigoUsuario());
       userTxt.setText(usuarioDP.getUserUsuario());

passwordTxt.setText(usuarioDP.getPasswordUsuari
o());
       codigoTxt.setEditable(false);
```

```
        guardar.setEnabled(true);
    }

    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc="
Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not
available, stay with the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lo
okandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo
info :
javax.swing.UIManager.getInstalledLookAndFeels())
{
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getCla
ssName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(UsuarioGUI.class
.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(UsuarioGUI.class
.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(UsuarioGUI.class
.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        } catch
(javax.swing.UnsupportedLookAndFeelException
ex) {

java.util.logging.Logger.getLogger(UsuarioGUI.class
.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new
Runnable() {

            public void run() {
                new UsuarioGUI().setVisible(true);
            }
        });
    }
```

Empaquetado: GUI **Componente:** VehiculoGUI.java

```
package gui;
import dp.VehiculoDP;
import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import java.awt.event.KeyEvent;
import java.text.DecimalFormat;
import javax.swing.JOptionPane;
import java.util.*;
import tesis.FondoJava;
/**
 *
 * @author  Juanka
 */
public class VehiculoGUI extends
javax.swing.JFrame {
    FondoJava as= new FondoJava(4);
    VehiculoDP vehiculoDP = new VehiculoDP();
    int existe=0, noexec=0;
    private HashMap codparq=new HashMap();
   /** Creates new form VehiculoGUI */
   public VehiculoGUI() {
      initComponents();
      this.add(as,BorderLayout.CENTER);
      as.repaint();
      codparq =
vehiculoDP.getHashParqueaderoDP();
      cargarMap(codparq);
   }

@Override
   public Image getIconImage() {
      Toolkit tool = Toolkit.getDefaultToolkit();
      Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
      return retValue;
   }

    public void cargarMap(HashMap map){
      int cont=0;
      String[] parqueaderos=new String[map.size()];
      Set set = map.entrySet();
      Iterator i = set.iterator();
      while(i.hasNext()){
      Map.Entry me = (Map.Entry)i.next();
      parqueaderos[cont]=(String)me.getKey();
      cont++;
      }
      for(int a=0;a<parqueaderos.length;a++)

comboParqueaderos.addItem(parqueaderos[a]);
   }
boolean band=false;
   private void
guardarPROActionPerformed(java.awt.event.Action
Event evt) {
      String codigo=codVEH.getText();
      String
codigoParqueadero=(String)comboParqueaderos.ge
tSelectedItem();
      String responsable=nombResp.getText();
      String marca=marcaVEH.getText();
      String fechaFab=fechFabVEH.getText();
      String placa=placaVEH.getText();
      String color=colorVEH.getText();
      String
modelo=(String)comboModelo.getSelectedItem();
      String
tipo=(String)comboTipo.getSelectedItem();
      String periodo=periodoVEH.getText();
      VehiculoDP serv=new
VehiculoDP(codigo,codigoParqueadero,responsable
,marca,fechaFab,placa,color,modelo,tipo,periodo);
      band=serv.verificarDP();

      if(band==true)
      {  int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
modificar el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
         if (seleccion == 0)
         {
         serv.modificarDP();
         codVEH.setText("");
         nombResp.setText("");
         marcaVEH.setText("");
         fechFabVEH.setText("");
         placaVEH.setText("");
         colorVEH.setText("");
         periodoVEH.setText("");
       } }  else
      {   int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Ingresar el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
         if (seleccion == 0)
         {
         serv.ingresarDP();
         codVEH.setEditable(true);
         nombResp.setEditable(false);
         marcaVEH.setEditable(false);
         fechFabVEH.setEditable(false);
         placaVEH.setEditable(false);
         colorVEH.setEditable(false);
         periodoVEH.setEditable(false);
```

```
            codVEH.setText("");
            nombResp.setText("");
            marcaVEH.setText("");
            fechFabVEH.setText("");
            placaVEH.setText("");
            colorVEH.setText("");
            periodoVEH.setText("");
            guardarPRO.setEnabled(false);
            JOptionPane.showMessageDialog(this,
"Vehiculo Ingresado con exito!!", "Ingreso de
Registro",
JOptionPane.INFORMATION_MESSAGE);
            this.repaint();
        }}
    }

    private void
codVEHActionPerformed(java.awt.event.ActionEven
t evt) {



    }

    private void
cancelarPROActionPerformed(java.awt.event.Action
Event evt) {
        this.dispose();
    }

    private void
codVEHFocusLost(java.awt.event.FocusEvent evt) {

        String cod=codVEH.getText();
        band=false;
        VehiculoDP serv=new VehiculoDP(cod);
        band=serv.verificarDP();
        if(cod.equals(""))
        {

JOptionPane.showMessageDialog(null,"Debe
ingresar un código!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
        }else
        if(band==true) {
            int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Modificar el registro?","Modificar
Vehiculo",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
            if (seleccion == 0)
            {
            codVEH.setEditable(false);
            nombResp.setEditable(true);
            marcaVEH.setEditable(true);
            fechFabVEH.setEditable(true);
            placaVEH.setEditable(true);
            colorVEH.setEditable(true);
            periodoVEH.setEditable(true);

//comboParqueaderos.setSelectedItem(serv.verDato
sDP()[0]);
            nombResp.setText(serv.verDatosDP()[1]);
            marcaVEH.setText(serv.verDatosDP()[2]);
            fechFabVEH.setText(serv.verDatosDP()[3]);
            placaVEH.setText(serv.verDatosDP()[4]);
            colorVEH.setText(serv.verDatosDP()[5]);
            String
modelo1=""+comboModelo.getSelectedItem();
            String modelo2=serv.verDatosDP()[6];
            if(modelo1.equals(modelo2))
            {
               comboModelo.removeAllItems();
               comboModelo.addItem("Sedan");
               comboModelo.addItem("Hatchback");
               this.repaint();
            }
            else
            {
                  comboModelo.removeAllItems();
                  comboModelo.addItem("HatchBack");
                  comboModelo.addItem("Sedan");
                  this.repaint();
               }
            this.repaint();
            String tipo1="Completo";
            String tipo2=serv.verDatosDP()[7];
            if(tipo1.equals(tipo2))
            {
               comboTipo.removeAllItems();
               comboTipo.addItem("Completo");
               comboTipo.addItem("Nocturno");
               comboTipo.addItem("Privilegios");
               this.repaint();
            }
            else
            {
                  if(tipo2.equals("Nocturno"))
                  {
                  comboTipo.addItem("Nocturno");
                  comboTipo.addItem("Completo");
                  comboTipo.addItem("Privilegios");
                  this.repaint();
                  }
                  if(tipo2.equals("Privilegios"))
                  {
                  comboTipo.addItem("Privilegios");
                  comboTipo.addItem("Nocturno");
                  comboTipo.addItem("Completo");
                  this.repaint();
                  }
               }
            this.repaint();
            periodoVEH.setText(serv.verDatosDP()[8]);
            guardarPRO.setEnabled(true);
            borrarPRO.setEnabled(true);
            }
         } else
```

```
      {  int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea
Ingresar nuevo registro?","Modificar
Vehiculo",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
          if (seleccion == 0)
          {
          guardarPRO.setEnabled(true);
          nombResp.setEditable(true);
          codVEH.setEditable(false);
          marcaVEH.setEditable(true);
          fechFabVEH.setEditable(true);
          placaVEH.setEditable(true);
          colorVEH.setEditable(true);
          periodoVEH.setEditable(true);
          nombResp.setText("");
          marcaVEH.setText("");
          fechFabVEH.setText("");
          placaVEH.setText("");
          colorVEH.setText("");
          periodoVEH.setText("");
      }}


   }

   private void
borrarPROActionPerformed(java.awt.event.ActionEv
ent evt) {
       String cod=codVEH.getText();
       VehiculoDP serv=new VehiculoDP(cod);
       int seleccion =
JOptionPane.showOptionDialog(this,"¿Desea Borrar
el registro?","Selector de
opciones",JOptionPane.YES_NO_CANCEL_OPTIO
N,JOptionPane.QUESTION_MESSAGE,null,new
Object[] { "Si", "No", "Cancelar" }, "No");
          if (seleccion == 0)
          {
          serv.eliminarDP();
          codVEH.setEditable(true);
          codVEH.setText("");
          nombResp.setText("");
          marcaVEH.setText("");
          fechFabVEH.setText("");
          placaVEH.setText("");
          colorVEH.setText("");
          periodoVEH.setText("");
          nombResp.setEditable(false);
          marcaVEH.setEditable(false);
          fechFabVEH.setEditable(false);
          placaVEH.setEditable(false);
          colorVEH.setEditable(false);
          periodoVEH.setEditable(false);
          guardarPRO.setEnabled(false);
          borrarPRO.setEnabled(false);
      }
   }

   private void
marcaVEHKeyTyped(java.awt.event.KeyEvent evt) {

}

   private void
colorVEHKeyTyped(java.awt.event.KeyEvent evt) {

   }

   private void
codVEHKeyTyped(java.awt.event.KeyEvent evt) {
       int k=(int)evt.getKeyChar();
          if (k >= 97 && k <= 122 || k>=65 && k<=90){

evt.setKeyChar((char)KeyEvent.VK_CLEAR);

JOptionPane.showMessageDialog(null,"No puede
ingresar letras!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
              }
          if(k==241 || k==209){

evt.setKeyChar((char)KeyEvent.VK_CLEAR);

JOptionPane.showMessageDialog(null,"No puede
ingresar letras!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
              }
       int limite = 12;
       if (codVEH.getText().length()== limite){
          evt.consume();

JOptionPane.showMessageDialog(null,"Escriba
máximo 12 números!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
       }
   }

   private void
nombRespKeyTyped(java.awt.event.KeyEvent evt) {
      int limite = 40;
       if (nombResp.getText().length()== limite)
       {  evt.consume();
       JOptionPane.showMessageDialog(null,"Escriba
máximo 40 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
       }
   }

   private void
fechFabVEHKeyTyped(java.awt.event.KeyEvent
evt) {
        int k=(int)evt.getKeyChar();
          if (k >= 97 && k <= 122 || k>=65 && k<=90){

evt.setKeyChar((char)KeyEvent.VK_CLEAR);
```

```
JOptionPane.showMessageDialog(null,"No puede
ingresar letras!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
            }
         if(k==241 || k==209){

evt.setKeyChar((char)KeyEvent.VK_CLEAR);

JOptionPane.showMessageDialog(null,"No puede
ingresar letras!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
            }
      int limite = 4;
      if (fechFabVEH.getText().length()== limite)
      {  evt.consume();
      JOptionPane.showMessageDialog(null,"Escriba
máximo 4 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
      }
   }

   private void
placaVEHKeyTyped(java.awt.event.KeyEvent evt) {
      int limite = 7;
      if (placaVEH.getText().length()== limite)
      {  evt.consume();
      JOptionPane.showMessageDialog(null,"Escriba
máximo 7 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
      }
   }

private void
comboParqueaderosActionPerformed(java.awt.even
t.ActionEvent evt) {
// TODO add your handling code here:
}

private void
periodoVEHKeyTyped(java.awt.event.KeyEvent evt)
{
   int k=(int)evt.getKeyChar();
         if (k >= 97 && k <= 122 || k>=65 && k<=90){

evt.setKeyChar((char)KeyEvent.VK_CLEAR);

JOptionPane.showMessageDialog(null,"No puede
ingresar letras!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
            }
         if(k==241 || k==209){

evt.setKeyChar((char)KeyEvent.VK_CLEAR);

JOptionPane.showMessageDialog(null,"No puede
ingresar letras!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
            }
      int limite = 6;
      if (fechFabVEH.getText().length()== limite)
      {  evt.consume();
      JOptionPane.showMessageDialog(null,"Escriba
máximo 6 caracteres!!!","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
      }
}

private void
comboTipoActionPerformed(java.awt.event.ActionEv
ent evt) {

}
    public void cargarDatos() {

codVEH.setText(vehiculoDP.getCodigoVehiculo());

nombResp.setText(vehiculoDP.getResponsableVehi
culo());

marcaVEH.setText(vehiculoDP.getMarcaVehiculo());

colorVEH.setText(vehiculoDP.getColorVehiculo());

placaVEH.setText(vehiculoDP.getPlacaVehiculo());

fechFabVEH.setText(vehiculoDP.getCodigoVehiculo
());
       codVEH.setEditable(false);
       guardarPRO.setEnabled(true);
    }
   public static void main(String args[]) {
      java.awt.EventQueue.invokeLater(new
Runnable() {
         public void run() {
            new VehiculoGUI().setVisible(true);
         }
      });
   }
```

Empaquetado: GUI **Componente:** VentanaAutorizaciónGUI.java

```
package gui;
import dp.UsuarioDP;
import java.awt.*;
import javax.swing.*;
import tesis.*;
/**
 *
 * @author Juanka
 */
public class VentanaAutorizacionGUI extends
javax.swing.JFrame {
    VentanaPrincipal P1 = new VentanaPrincipal();
    VentanaPrincipal2 P2 = new VentanaPrincipal2();
    FondoJava ap = new FondoJava(1);

    /** Creates new form VentanaAutorizacionGUI */
    public VentanaAutorizacionGUI() {
        initComponents();
        this.setTitle("Sistema Automático de
Asignación y Registro");
        this.add(ap,BorderLayout.CENTER);
        ap.repaint();
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }
private void
IngresarActionPerformed(java.awt.event.ActionEven
t evt) {
    String usuario=User.getText();
    String pass=Password.getText();
    UsuarioDP registro=new UsuarioDP(usuario);
    int opcion=registro.verificarIngreso(usuario,pass);
    if(opcion==0)
        JOptionPane.showMessageDialog(null,"El
usuario no existe","Ventana Error
Datos",JOptionPane.ERROR_MESSAGE);
    else
    {
        if(opcion==1)
        {
            P1.setLocationRelativeTo(null);
            P1.setVisible(true);
        }
        if(opcion==2)
        {
            P2.setLocationRelativeTo(null);
            P2.setVisible(true);
        }
    }
    this.dispose();
}

private void
SalirActionPerformed(java.awt.event.ActionEvent
evt) {
    this.dispose();
}

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc="
Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not
available, stay with the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lo
okandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo
info :
javax.swing.UIManager.getInstalledLookAndFeels())
{
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getCla
ssName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(VentanaAutorizac
ionGUI.class.getName()).log(java.util.logging.Level.
SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(VentanaAutorizac
ionGUI.class.getName()).log(java.util.logging.Level.
SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(VentanaAutorizac
ionGUI.class.getName()).log(java.util.logging.Level.
SEVERE, null, ex);
        } catch
(javax.swing.UnsupportedLookAndFeelException
ex) {

java.util.logging.Logger.getLogger(VentanaAutorizac
ionGUI.class.getName()).log(java.util.logging.Level.
SEVERE, null, ex);
```

```
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new
Runnable() {
            public void run() {
                new
VentanaAutorizacionGUI().setVisible(true);
            }
        });
    }
```

Empaquetado: GUI **Componente:** VentanaPrincipal.java

```
package gui;

import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import tesis.*;

/**
 *
 * @author  Juanka
 */
public class VentanaPrincipal extends
javax.swing.JFrame {
    //FondoMenuPrincipal menuP=new
FondoMenuPrincipal();
    FondoJava as= new FondoJava(2);
    /** Creates new form general */
    public VentanaPrincipal() {
        initComponents();
        this.add(as,BorderLayout.CENTER);
        as.repaint();
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }
private void
usuarioActionPerformed(java.awt.event.ActionEvent
evt) {
    UsuarioGUI C1 = new UsuarioGUI();
     C1.setLocationRelativeTo(null);
     C1.setVisible(true);
}

    private void
vehiculosActionPerformed(java.awt.event.ActionEve
nt evt) {
     VehiculoGUI C1 = new VehiculoGUI();
     C1.setLocationRelativeTo(null);
     C1.setVisible(true);

}

    private void
convehiculoActionPerformed(java.awt.event.ActionE
vent evt) {
     ConsultaVehiculoGUI C1 = new
ConsultaVehiculoGUI();
     C1.setLocationRelativeTo(null);
     C1.setVisible(true);  // TODO add your handling
code here:
}

    private void
conusuarioActionPerformed(java.awt.event.ActionEv
ent evt) {
      ConsultaUsuarioGUI C1 = new
ConsultaUsuarioGUI();
     C1.setLocationRelativeTo(null);
     C1.setVisible(true);  // TODO add your handling
code here:
}

    private void
conregistroActionPerformed(java.awt.event.ActionEv
ent evt) {
     ConsultaRegistrosGUI CR1 = new
ConsultaRegistrosGUI();
     CR1.setLocationRelativeTo(null);
     CR1.setVisible(true);  // TODO add your handling
code here:
    }

    private void
asignarActionPerformed(java.awt.event.ActionEvent
evt) {
     AsignarGUI C1 = new AsignarGUI();
     C1.setLocationRelativeTo(null);
     C1.setVisible(true);   // TODO add your handling
code here:
    }

private void
espaciosActionPerformed(java.awt.event.ActionEve
nt evt) {
     EspaciosGUI E1= new EspaciosGUI();
     E1.setLocationRelativeTo(null);
     E1.setVisible(true);
}

private void
conespacioActionPerformed(java.awt.event.ActionE
vent evt) {
     ConsultaEspaciosGUI CE1= new
ConsultaEspaciosGUI();
     CE1.setLocationRelativeTo(null);
     CE1.setVisible(true);
}

    private void
jMenuItem1ActionPerformed(java.awt.event.ActionE
vent evt) {
     ReporteGUI R1= new ReporteGUI();
     R1.setLocationRelativeTo(null);
     R1.setVisible(true);
    }
```

```
    private void
parqueaderosActionPerformed(java.awt.event.Actio
nEvent evt) {
     ParqueaderoGUI P1= new ParqueaderoGUI();
     P1.setLocationRelativeTo(null);
     P1.setVisible(true);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new
Runnable() {
            public void run() {
                new VentanaPrincipal().setVisible(true);
            }
        });
    }
```

Empaquetado: GUI **Componente:** VentanaPrincipal2.java

```
package gui;

import java.awt.BorderLayout;
import java.awt.Image;
import java.awt.Toolkit;
import tesis.FondoJava;

/**
 *
 * @author Juanka
 */
public class VentanaPrincipal2 extends
javax.swing.JFrame {
    FondoJava as= new FondoJava(2);
    /** Creates new form VentanaPrincipal2 */
    public VentanaPrincipal2() {
        initComponents();
        this.add(as,BorderLayout.CENTER);
        as.repaint();
    }

@Override
    public Image getIconImage() {
        Toolkit tool = Toolkit.getDefaultToolkit();
        Image retValue =
tool.getDefaultToolkit().getImage(ClassLoader.getSy
stemResource("Gimagenes/Parqueo.png"));
        return retValue;
    }
private void
convehiculoActionPerformed(java.awt.event.ActionE
vent evt) {
    ConsultaVehiculoGUI C1 = new
ConsultaVehiculoGUI();
    C1.setLocationRelativeTo(null);
    C1.setVisible(true);
}

private void
conespaciosActionPerformed(java.awt.event.Action
Event evt) {
    ConsultaEspaciosGUI CE1 = new
ConsultaEspaciosGUI();
    CE1.setLocationRelativeTo(null);
    CE1.setVisible(true);
}

private void
conregistrosActionPerformed(java.awt.event.ActionE
vent evt) {
// TODO add your handling code here:
}

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc="
Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not
available, stay with the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lo
okandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo
info :
javax.swing.UIManager.getInstalledLookAndFeels())
{
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getCla
ssName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(VentanaPrincipal
2.class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(VentanaPrincipal
2.class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(VentanaPrincipal
2.class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
        } catch
(javax.swing.UnsupportedLookAndFeelException
ex) {

java.util.logging.Logger.getLogger(VentanaPrincipal
2.class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new
Runnable() {

            public void run() {
                new VentanaPrincipal2().setVisible(true);
            }
        });
    }
```

Empaquetado: MD **Componente:** EspacioMD.java

```
package md;
import java.sql.*;
import dp.*;
import java.util.*;

public class EspacioMD {

   private Connection conexion;
   private Statement sentencia;
   private ResultSet resultset;
   private String cadena;
   private EspaciosDP espaciosdp;


    public EspacioMD(EspaciosDP newEspaciosDP)
{
       espaciosdp=newEspaciosDP;
       conexion=null;
       try {
          Class.forName
("oracle.jdbc.driver.OracleDriver");
       }catch(ClassNotFoundException
e){System.out.println(e.getMessage());}
       try {
          Locale.setDefault(Locale.ENGLISH);
          conexion =
DriverManager.getConnection("jdbc:oracle:thin:@loc
alhost:1521:XE","TESIS","abc123");
          sentencia = conexion.createStatement();

       }catch(SQLException x){
          System.out.println(x.getMessage());}

   }

   public HashMap getHashParqueaderoMD() {
       HashMap map=new HashMap();
       try {
          resultset=sentencia.executeQuery("select
CodigoParqueadero,Descripcion from
Parqueadrero");
          while(resultset.next())
          {

map.put(resultset.getString(1),resultset.getString(2))
;
          }
       }catch(SQLException x){
          System.out.println(x.getMessage());
       }
       return map;
   }


   public void ingresarMD() {
       try{
          sentencia.executeUpdate("INSERT INTO
Espacios(CodigoEspacio,CodigoParqueadero,Descr
ipcion,Disponibilidad) VALUES
('"+espaciosdp.getCodigoEspacio()+"','"+espaciosdp
.getCodigoParqueadero()+"','"+espaciosdp.getDescr
ipcion()+"','"+espaciosdp.getDisponibilidad()+"')");
          sentencia.close();
          conexion.close();
       }catch(SQLException x){
          System.out.println(x.getMessage());}
   }

  public void eliminarMD(){

       try {
          sentencia.executeUpdate("delete from
Espacios where CodigoEspacio =
'"+espaciosdp.getCodigoEspacio()+"'");
          sentencia.close();
          conexion.close();
       }catch(SQLException x){
          System.out.println(x.getMessage());
       }
   }
  public String[] verDatosMD() {
       String[] datos = new String[9];

       try {

          resultset =
sentencia.executeQuery("SELECT * FROM
ESPACIOS where
CODIGOESPACIO='"+espaciosdp.getCodigoEspaci
o()+"'");

          while(resultset.next()){

datos[0]=resultset.getString("CODIGOESPACIO");

datos[1]=resultset.getString("CODIGOPARQUEADE
RO");

datos[2]=resultset.getString("DESCRIPCION");

datos[3]=resultset.getString("DISPONIBILIDAD");
          }

       }catch(SQLException x){
          System.out.println(x.getMessage());
       }

       return datos;
   }

   public void modificarMD() {

       try {
```

```
            sentencia.executeUpdate("update espacios
set
CODIGOPARQUEADERO="'+espaciosdp.getCodig
oParqueadero()+"',DESCRIPCION="'+espaciosdp.g
etDescripcion()+"',DISPONIBILIDAD="'+espaciosdp.
getDisponibilidad()+"' WHERE
CODIGOESPACIO="'+espaciosdp.getCodigoEspaci
o()+"'");
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
    }

     public boolean verificarMD(){
        boolean band=false;

        try {
            resultset=sentencia.executeQuery("select *
from ESPACIOS where CODIGOESPACIO =
'"+espaciosdp.getCodigoEspacio()+"'");
            while(resultset.next())
            {
                band= true;
            }
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
        return band;
    }

      public EspaciosDP[] verServiciosMD(String
verservicios,int band) {
        int cont=0;
        int tot=0;
        EspaciosDP[] servicios=new EspaciosDP[0];

        try {

            resultset = sentencia.executeQuery("select *
from ESPACIOS");
            if(band==0)
            {
                resultset=sentencia.executeQuery("select
count(CODIGOESPACIO) m from ESPACIOS");

                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
                servicios = new EspaciosDP[tot];
                resultset =
sentencia.executeQuery("select * from
ESPACIOS");
                while(resultset.next()){
                    servicios[cont]=new
EspaciosDP(resultset.getString("CODIGOESPACIO
"),resultset.getString("CODIGOPARQUEADERO"),r
esultset.getString("DESCRIPCION"),resultset.getStri
ng("DISPONIBILIDAD"));
                    cont++;
                }
            }
            if(band==1)
            {
                resultset=sentencia.executeQuery("select
count(CODIGOESPACIO) m from ESPACIOS
where CODIGOESPACIO="'+verservicios+"'");

                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
                servicios = new EspaciosDP[tot];
                resultset =
sentencia.executeQuery("select * from ESPACIOS
where CODIGOESPACIO="'+verservicios+"'");

                while(resultset.next()){
                    servicios[cont]=new
EspaciosDP(resultset.getString("CODIGOESPACIO
"),resultset.getString("CODIGOPARQUEADERO"),r
esultset.getString("DESCRIPCION"),resultset.getStri
ng("DISPONIBILIDAD"));
                    cont++;
                }
            }
            if(band==2)
            {
                resultset=sentencia.executeQuery("select
count(CODIGOESPACIO) m from ESPACIOS
where
CODIGOPARQUEADERO="'+verservicios+"'");

                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
                servicios = new EspaciosDP[tot];
                resultset =
sentencia.executeQuery("select * from ESPACIOS
where
CODIGOPARQUEADERO="'+verservicios+"'");

                while(resultset.next()){
                    servicios[cont]=new
EspaciosDP(resultset.getString("CODIGOESPACIO
"),resultset.getString("CODIGOPARQUEADERO"),r
esultset.getString("DESCRIPCION"),resultset.getStri
ng("DISPONIBILIDAD"));
                    cont++;
                }
            }
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());
```

```
    }
    return servicios;                                    }
}
```

Empaquetado: MD **Componente:** IngresoMD.java

```
package md;
import java.util.*;
import java.sql.*;
import java.util.Locale;
import dp.*;

/**
 *
 * @author Juanka
 */
public class IngresoMD {
   private Connection conexion;
   private Statement sentencia;
   private ResultSet resultset;
   private String cadena;
   public IngresoMD() {
       conexion=null;
       try {
          Class.forName
("oracle.jdbc.driver.OracleDriver");
       }catch(ClassNotFoundException
e){System.out.println(e.getMessage());}
       try {
          Locale.setDefault(Locale.ENGLISH);
          conexion =
DriverManager.getConnection("jdbc:oracle:thin:@loc
alhost:1521:XE","TESIS","abc123");
          sentencia = conexion.createStatement();
       }catch(SQLException x){
          System.out.println(x.getMessage());}
   }

public String[] verificarDatosMD(String user) {
       String[] datos = new String[4];
       try {

             resultset =
sentencia.executeQuery("SELECT * FROM
USUARIOS WHERE CODIGOUSUARIO =
'"+user+"'");
             while(resultset.next()){

datos[0]=resultset.getString("CODIGOUSUARIO");
               datos[1]=resultset.getString("USER");

datos[2]=resultset.getString("PASSWORD");
               datos[3]=resultset.getString("TIPO");
          }
       }catch(SQLException x){
          System.out.println(x.getMessage());
       }
       return datos;
   }
}
```

Empaquetado: MD **Componente:** ParqueaderoMD.java

```
package md;
import dp.*;
import java.sql.*;
import java.util.*;

/**
 *
 * @author Juanka
 */
public class ParqueaderoMD {
    private String estado="activo";
    private String estado1="desactivado";
    private Connection conexion;
    private Statement sentencia;
    private ResultSet resultset;
    private ParqueaderoDP parqueaderoDP;

    public ParqueaderoMD(){
    }

    public ParqueaderoMD(ParqueaderoDP
parqueaderodp)
    {
        parqueaderoDP=parqueaderodp;
        try
        {
            Class.forName
("oracle.jdbc.driver.OracleDriver");
        }
        catch(ClassNotFoundException e)
        {
            System.out.println(e.getMessage());
        }
        try
        {
            Locale.setDefault(Locale.ENGLISH);
            conexion =
DriverManager.getConnection("jdbc:oracle:thin:@loc
alhost:1521:XE","TESIS","abc123");
            sentencia = conexion.createStatement();
        }
        catch(SQLException x)
        {
            System.out.println(x.getMessage());
        }
    }

    public void ingresarMD() {
        try{
            sentencia.executeUpdate("INSERT INTO
PARQUEADRERO(CodigoParqueadero,Descripcion
) VALUES
('"+parqueaderoDP.getCodigoParqueadero()+"','"+p
arqueaderoDP.getDescripcion()+"')");
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());}
    }

public void eliminarMD(){

        try {
            sentencia.executeUpdate("delete from
PARQUEADRERO where CodigoParqueadero =
'"+parqueaderoDP.getCodigoParqueadero()+"'");
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
    }

    public String[] verDatosMD(){
        String[] datos = new String[4];
        try {

            resultset = sentencia.executeQuery("select *
from PARQUEADRERO");

            while(resultset.next()){

datos[0]=resultset.getString("CODIGOPARQUEADE
RO");

datos[1]=resultset.getString("DESCRIPCION");
            }
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }

        return datos;
    }

public void modificarMD() {

        try {
            sentencia.executeUpdate("update
PARQUEADRERO set
DESCRIPCION='"+parqueaderoDP.getDescripcion()
+"' where
CODIGOVEHICULO='"+parqueaderoDP.getCodigo
Parqueadero()+"'");
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
    }

     public boolean verificarMD(){
        boolean band=false;

        try {
```

```
            resultset=sentencia.executeQuery("select *
from PARQUEADRERO where
CODIGOPARQUEADERO =
'"+parqueaderoDP.getCodigoParqueadero()+"'");
            while(resultset.next())
            {
                band= true;
            }
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
        return band;
    }

public ParqueaderoDP[] verServiciosMD(String
verservicios,int band) {
        int cont=0;
        int tot=0;
        ParqueaderoDP[] servicios=new
ParqueaderoDP[0];

        try {

            resultset = sentencia.executeQuery("select *
from PARQUEADRERO");
            if(band==0)
            {
                resultset=sentencia.executeQuery("select
count(CODIGOPARQUEADERO) m from
PARQUEADRERO");

                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
                servicios = new ParqueaderoDP[tot];
                resultset =
sentencia.executeQuery("select * from
PARQUEADRERO");
                while(resultset.next()){
                    servicios[cont]=new
ParqueaderoDP(resultset.getString("CODIGOPARQ
UEADERO"),resultset.getString("DESCRIPCION"));
                    cont++;
                }
            }
            if(band==1)
            {
                resultset=sentencia.executeQuery("select
count(CODIGOPARQUEADERO) m from
PARQUEADRERO where
CODIGOPARQUEADERO='"+verservicios+"'");
                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
                servicios = new ParqueaderoDP[tot];
                resultset =
sentencia.executeQuery("select * from
PARQUEADRERO where
CODIGOPARQUEADERO='"+verservicios+"'");

                while(resultset.next()){
                    servicios[cont]=new
ParqueaderoDP(resultset.getString("CODIGOPARQ
UEADERO"),resultset.getString("DESCRIPCION"));
                    cont++;
                }

            }
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
        return servicios;
    }

}
```

Empaquetado: MD **Componente:** UsuarioMD.java

```
package md;

import dp.*;
import java.util.*;
import java.sql.*;
import java.util.Locale;

/** @pdOid b9d37506-b434-40df-9c53-
32558eed0235 */
public class UsuarioMD {
   private Connection conexion;
   private Statement sentencia;
   private ResultSet resultset;
   private String cadena;
   private UsuarioDP usuariodp;
   public UsuarioMD() {

   }
   public UsuarioMD(UsuarioDP nuevoUsuarioDP) {
       usuariodp=nuevoUsuarioDP;
       conexion=null;
       try {
          Class.forName
("oracle.jdbc.driver.OracleDriver");
       }catch(ClassNotFoundException
e){System.out.println(e.getMessage());}
       try {
            Locale.setDefault(Locale.ENGLISH);
            conexion =
DriverManager.getConnection("jdbc:oracle:thin:@loc
alhost:1521:XE","TESIS","abc123");
            sentencia = conexion.createStatement();
       }catch(SQLException x){
          System.out.println(x.getMessage());}
    }

   public void ingresarMD() {
     try{
         sentencia.executeUpdate("INSERT INTO
USUARIOS VALUES
('"+usuariodp.getCodigoUsuario()+"','"+usuariodp.ge
tUserUsuario()+"','"+usuariodp.getPasswordUsuario(
)+"','"+usuariodp.getTipoUsuario()+"')");
          sentencia.close();
          conexion.close();
       }catch(SQLException x){
          System.out.println(x.getMessage());}
   }

   public void eliminarMD() {
     try {
          sentencia.executeUpdate("delete from
usuarios where codigousuario =
'"+usuariodp.getCodigoUsuario()+"'");
          sentencia.close();
          conexion.close();
       }catch(SQLException x){
          System.out.println(x.getMessage());
       }
   }

   public void modificarMD() {
     try {
          sentencia.executeUpdate("update usuario
set
USER='"+usuariodp.getUserUsuario()+"',PASSWOR
D='"+usuariodp.getPasswordUsuario()+"',TIPO='"+u
suariodp.getTipoUsuario()+"' where
CODIGOUSUARIO='"+usuariodp.getCodigoUsuario(
)+"'");
          sentencia.close();
          conexion.close();
       }catch(SQLException x){
          System.out.println(x.getMessage());
       }
   }

   public String[] verDatosMD() {
       String[] datos = new String[4];

       try {
          resultset = sentencia.executeQuery(
"SELECT * FROM USUARIOS where
CODIGOUSUARIO='"+usuariodp.getCodigoUsuario(
)+"'");

          while(resultset.next()){


datos[0]=resultset.getString("CODIGOUSUARIO");
               datos[1]=resultset.getString("USER");

datos[2]=resultset.getString("PASSWORD");
               datos[3]=resultset.getString("TIPO");
          }
       }catch(SQLException x){
          System.out.println(x.getMessage());
       }
       return datos;
    }

   public boolean verificarMD(){
       boolean band=false;

       try {
          resultset=sentencia.executeQuery( "SELECT
* FROM USUARIOS where CODIGOUSUARIO =
'"+usuariodp.getCodigoUsuario()+"'");
          while(resultset.next())
          {
             band= true;
          }
       }catch(SQLException x){
          System.out.println(x.getMessage());
```

```
        }
        return band;
    }

    public UsuarioDP[] verServiciosMD() {
        int cont=0;
        int tot=0;
        UsuarioDP[] servicios=new UsuarioDP[0];

        try {

                resultset =
sentencia.executeQuery("SELECT * from
Usuarios");

resultset=sentencia.executeQuery("SELECT
count(CODIGOUSUARIO) m from Usuarios");

                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
                servicios = new UsuarioDP[tot];
                resultset =
sentencia.executeQuery("select * from usuarios");
                while(resultset.next()){
                    servicios[cont]=new
UsuarioDP(resultset.getString("CODIGOUSUARIO")
,resultset.getString("USER"),resultset.getString("PA
SSWORD"),resultset.getString("TIPO"));
                    cont++;
                }
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
        return servicios;
    }
}
```

Empaquetado: MD **Componente:** VehiculoMD.java

```
package md;
import java.sql.*;
import dp.*;
import java.util.*;

public class VehiculoMD {
   //VehiculoDP vehiculodp;
   private Connection conexion;
   private Statement sentencia;
   private ResultSet resultset;
   private String cadena;
   private VehiculoDP vehiculodp;


    public VehiculoMD(VehiculoDP parpvehiculoDP) {
        vehiculodp=parpvehiculoDP;
        conexion=null;
        try {
            Class.forName
("oracle.jdbc.driver.OracleDriver");
        }catch(ClassNotFoundException
e){System.out.println(e.getMessage());}
        try {
            Locale.setDefault(Locale.ENGLISH);
            conexion =
DriverManager.getConnection("jdbc:oracle:thin:@loc
alhost:1521:XE","TESIS","abc123");
            sentencia = conexion.createStatement();

        }catch(SQLException x){
            System.out.println(x.getMessage());}

    }

    public HashMap getHashParqueaderoMD() {
        HashMap map=new HashMap();
        try {
            resultset=sentencia.executeQuery("select
CodigoParqueadero,Descripcion from
Parqueadrero");
            while(resultset.next())
            {

map.put(resultset.getString(1),resultset.getString(2))
;
            }
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
        return map;
    }

    public HashMap getHashVehiculoMD() {
        HashMap map=new HashMap();
        try {
            resultset=sentencia.executeQuery("select
CodigoVehiculo,CodigoParqueadero from
Vehiculo");
            while(resultset.next())
            {

map.put(resultset.getString(1),resultset.getString(2))
;
            }
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
        return map;
    }

    public void ingresarMD() {
        try{
            sentencia.executeUpdate("INSERT INTO
Vehiculos(CodigoVehiculo,CodigoParqueadero,Res
ponsable,Marca,FechaFab,Placa,Color,Modelo,Tipo
Parq, Periodo) VALUES
('"+vehiculodp.getCodigoVehiculo()+"','"+vehiculodp.
getCodigoParqueaderoVehiculo()+"','"+vehiculodp.g
etResponsableVehiculo()+"','"+vehiculodp.getMarca
Vehiculo()+"','"+vehiculodp.getFechaFabVehiculo()+
"','"+vehiculodp.getPlacaVehiculo()+"','"+vehiculodp.
getColorVehiculo()+"','"+vehiculodp.getModeloVehic
ulo()+"','"+vehiculodp.getTipoParqVehiculo()+"','"+ve
hiculodp.getPeriodoVehiculo()+"')");
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());}
    }

  public void eliminarMD(){

        try {
            sentencia.executeUpdate("delete from
Vehiculos where CodigoVehiculo =
'"+vehiculodp.getCodigoVehiculo()+"'");
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
    }
  public String[] verDatosMD() {
        String[] datos = new String[9];

        try {

            resultset =
sentencia.executeQuery("SELECT * FROM
VEHICULOS where
CODIGOVEHICULO='"+vehiculodp.getCodigoVehic
ulo()+"'");
```

```
            while(resultset.next()){

datos[0]=resultset.getString("CODIGOPARQUEADERO");

datos[1]=resultset.getString("RESPONSABLE");
                datos[2]=resultset.getString("MARCA");

datos[3]=resultset.getString("FECHAFAB");
                datos[4]=resultset.getString("PLACA");
                datos[5]=resultset.getString("COLOR");
                datos[6]=resultset.getString("MODELO");

datos[7]=resultset.getString("TIPOPARQ");

datos[8]=resultset.getString("PERIODO");

            }

        }catch(SQLException x){
            System.out.println(x.getMessage());
        }

        return datos;
    }

    public void modificarMD() {

        try {
            sentencia.executeUpdate("update vehiculos set
CODIGOPARQUEADERO='"+vehiculodp.getCodigoParqueaderoVehiculo()+"',RESPONSABLE='"+vehiculodp.getResponsableVehiculo()+"',MARCA='"+vehiculodp.getMarcaVehiculo()+"',FECHAFAB='"+vehiculodp.getFechaFabVehiculo()+"',PLACA='"+vehiculodp.getPlacaVehiculo()+"',COLOR='"+vehiculodp.getColorVehiculo()+"',MODELO='"+vehiculodp.getModeloVehiculo()+"',TIPOPARQ='"+vehiculodp.getTipoParqVehiculo()+"',PERIODO='"+vehiculodp.getPeriodoVehiculo()+"' where
CODIGOVEHICULO='"+vehiculodp.getCodigoVehiculo()+"'");
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
    }

     public boolean verificarMD(){
        boolean band=false;

        try {
            resultset=sentencia.executeQuery("select * from VEHICULOS where CODIGOVEHICULO =
'"+vehiculodp.getCodigoVehiculo()+"'");
            while(resultset.next())
            {
                band= true;
            }
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
        return band;
    }
     public VehiculoDP[] verServiciosMD(String verservicios,int band) {
        int cont=0;
        int tot=0;
        VehiculoDP[] servicios=new VehiculoDP[0];

        try {

            resultset = sentencia.executeQuery("select * from VEHICULOS");
            if(band==0)
            {
                resultset=sentencia.executeQuery("select count(CODIGOVEHICULO) m from VEHICULOS");

                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
                servicios = new VehiculoDP[tot];
                resultset =
sentencia.executeQuery("select * from VEHICULOS");
                while(resultset.next()){
                    servicios[cont]=new
VehiculoDP(resultset.getString("CODIGOVEHICULO"),resultset.getString("CODIGOPARQUEADERO"),resultset.getString("RESPONSABLE"),resultset.getString("MARCA"),resultset.getString("FECHAFAB"),resultset.getString("PLACA"),resultset.getString("COLOR"),resultset.getString("MODELO"),resultset.getString("TIPOPARQ"),resultset.getString("PERIODO"));
                    cont++;
                }
            }
            if(band==1)
            {
                resultset=sentencia.executeQuery("select count(CODIGOVEHICULO) m from VEHICULOS where CODIGOVEHICULO='"+verservicios+"'");

                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
                servicios = new VehiculoDP[tot];
                resultset =
sentencia.executeQuery("select * from VEHICULOS where CODIGOVEHICULO='"+verservicios+"'");

                while(resultset.next()){
```

```
                servicios[cont]=new
VehiculoDP(resultset.getString("CODIGOVEHICUL
O"),resultset.getString("CODIGOPARQUEADERO"),
resultset.getString("RESPONSABLE"),resultset.getS
tring("MARCA"),resultset.getString("FECHAFAB"),re
sultset.getString("PLACA"),resultset.getString("COL
OR"),resultset.getString("MODELO"),resultset.getStr
ing("TIPOPARQ"),resultset.getString("PERIODO"));
                cont++;
              }

          }
          if(band==2)
          {
              resultset=sentencia.executeQuery("select
count(CODIGOVEHICULO) m from VEHICULOS
where
CODIGOPARQUEADERO='"+verservicios+"'");

              while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
              }
              servicios = new VehiculoDP[tot];
              resultset =
sentencia.executeQuery("select * from VEHICULOS
where
CODIGOPARQUEADERO='"+verservicios+"'");

                while(resultset.next()){
                    servicios[cont]=new
VehiculoDP(resultset.getString("CODIGOVEHICUL
O"),resultset.getString("CODIGOPARQUEADERO"),
resultset.getString("RESPONSABLE"),resultset.getS
tring("MARCA"),resultset.getString("FECHAFAB"),re
sultset.getString("PLACA"),resultset.getString("COL
OR"),resultset.getString("MODELO"),resultset.getStr
ing("TIPOPARQ"),resultset.getString("PERIODO"));
                cont++;
              }
            }
            sentencia.close();
            conexion.close();
        }catch(SQLException x){
            System.out.println(x.getMessage());
        }
        return servicios;
    }

}
```

Empaquetado: MD **Componente:** VerificarMD.java

```
package md;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Locale;

/**
 *
 * @author Juanka
 */
public class VerificarMD {
    Connection conexion;
    Statement sentencia;
    ResultSet resultset;
    public VerificarMD()
    { try
        {
            Class.forName
("oracle.jdbc.driver.OracleDriver");
        }
        catch(ClassNotFoundException e)
        {
            System.out.println(e.getMessage());
        }
        try
        {
            Locale.setDefault(Locale.ENGLISH);
            conexion =
DriverManager.getConnection("jdbc:oracle:thin:@loc
alhost:1521:XE","TESIS","abc123");
            sentencia =
conexion.createStatement(resultset.TYPE_SCROLL
_INSENSITIVE, resultset.CONCUR_READ_ONLY);
            //sentencia = conexion.createStatement();
        }
        catch(SQLException x)
        {
            System.out.println(x.getMessage());
        }
    }

    public int verificarMes(int mes)
    {
        int num=0;
        int cont=0;
        try{
                resultset =
sentencia.executeQuery("select * from REGISTRO
where MES='"+mes+"'");
                    while(resultset.next()){
                        num++;
                        cont++;
                    }
                sentencia.close();
                conexion.close();
            }catch(SQLException x){}
     return num;
    }

   public int completoUso()
   {
       int tot=0;
       try{
          resultset=sentencia.executeQuery("select
count(CODIGOREGISTRO) m from REGISTRO");
                while(resultset.next()){

tot=Integer.parseInt(resultset.getString("m"));
                }
          }catch(SQLException x){}
       return tot;
   }

}
```

Empaquetado: Tesis **Componente:** Fondo.java

```
package tesis;

import java.awt.*;
import javax.swing.ImageIcon;
/**
 *
 * @author Juanka
 */
public class FondoJava extends javax.swing.JPanel
{
    ImageIcon imagenFondo;
    public FondoJava(){
    this.setSize(325,225);
    }

public FondoJava(int num){
    if(num==1)
    {
     imagenFondo = new
ImageIcon(getClass().getResource("/Gimagenes/Lo
go Intro.jpg"));
     this.setSize(325,225);
    }
    if(num==2)
    {
     imagenFondo = new
ImageIcon(getClass().getResource("/Gimagenes/Lo
go Menu Principal.jpg"));
     this.setSize(800,700);
    }
    if(num==3)
    {
     imagenFondo = new
ImageIcon(getClass().getResource("/Gimagenes/Lo
go Pantallas.png"));
     this.setSize(600,330);
    }
    if(num==4)
    {
     imagenFondo = new
ImageIcon(getClass().getResource("/Gimagenes/Lo
go Pantallas1.jpg"));
     this.setSize(550,420);
    }
    if(num==5)
    {
     imagenFondo = new
ImageIcon(getClass().getResource("/Gimagenes/Lo
go Pantallas1.jpg"));
     this.setSize(876,600);
    }
    if(num==6)
    {
     imagenFondo = new
ImageIcon(getClass().getResource("/Gimagenes/Lo
go Pantallas1.jpg"));
     this.setSize(1050,600);
    }
    if(num==7)
    {
     imagenFondo = new
ImageIcon(getClass().getResource("/Gimagenes/Lo
go Asignacion.jpg"));
     this.setSize(410,240);
    }
    if(num==8)
    {
     imagenFondo = new
ImageIcon(getClass().getResource("/Gimagenes/Lo
go Pantallas2.jpg"));
     this.setSize(420,480);
    }
}

@Override
    public void paint(Graphics g){
        Dimension tamanio = getSize();

g.drawImage(imagenFondo.getImage(),0,0,tamanio.
width, tamanio.height, null);
        setOpaque(false);
        super.paintComponent(g);
}

}
```

MANUAL DE USUARIO

Validación de Ingreso



Esta pantalla es la que permite hacer el ingreso de usuarios al sistema.

Existen 2 tipos de usuarios Administradores y Guardias, el administrador posee acceso total a todas las administraciones, procesos y consultas. Mientras el guardia solo puede realizar Consultas

1. En los campos de Usuario y Password digite su usuario y contraseña respectivamente
2. Presione Ingresar
3. El caso de querer salir del sistema presione Salir.

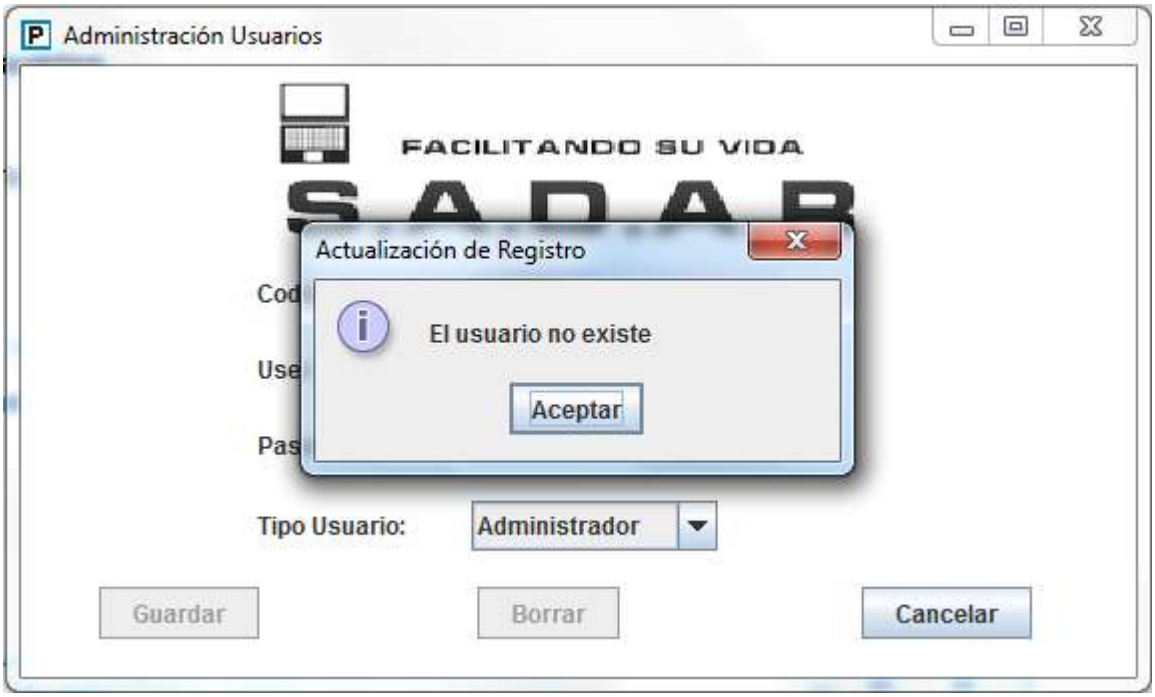
Usuario Tipo: Administrador

Administración Usuarios

Esta ventana permite el ingreso de nuevos usuarios en el sistema.

Ingreso Usuarios:

1. Digite el código del usuario en el campo Código Usuario. En caso de no existir dicho código se presentara un mensaje de información de que dicho código de usuario no existe en la base de datos.



2. Presione Aceptar y todos los demás campos se habilitarán excepto el campo de código usuario.
3. Llene todos los demás campos y presione guardar.
4. Un cuadro informativo notificara si desea ingresar un nuevo registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.



En caso de afirmar el ingreso del nuevo usuario se mostrará una pantalla informativa



Modificación Usuarios:

1. Digite el código del usuario en el campo Código Usuario. En caso de existir dicho código se presentara un mensaje de información de que dicho código de usuario existe en la base de datos y si desea modificarlo.



2. Presione “SI” en el caso de desear modificar dicho registro, en caso contrario presione “NO” para no modificar dicho registro y presione “Cancelar” en el caso de desear hacer alguna modificación al código previamente ingresado
3. Cambie todos los demás campos y presione guardar.
4. Un cuadro informativo notificara si desea modificar este registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.
5. En caso de afirmar la modificación del usuario se mostrará una pantalla informativa.



Usuario Tipo: Administrador

Administración Parqueaderos

Esta ventana permite el ingreso de nuevos parqueaderos al sistema.



Ingreso Parqueadero:

1. Digite el código del parqueadero en el campo Código. En caso de no existir dicho código se presentara un mensaje de información de que si desea ingresar este nuevo parqueadero.



2. Si desea ingresar un nuevo registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.
3. Llene todos los demás campos y presione guardar.
4. Un cuadro informativo notificara si desea ingresar un nuevo registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.



En caso de afirmar el ingreso del nuevo usuario se mostrará una pantalla informativa de que el parqueadero fue ingresado con éxito.

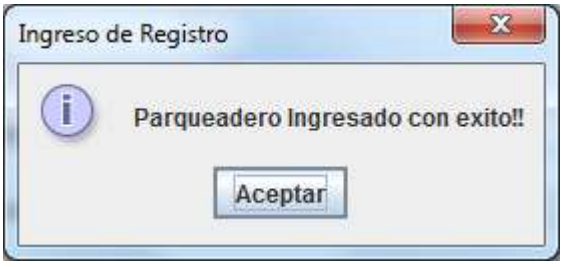


Modificación Parqueadero:

1. Digite el código del usuario en el campo Código Usuario. En caso de existir dicho código se presentara un mensaje de información de que dicho código de usuario existe en la base de datos y si desea modificarlo.



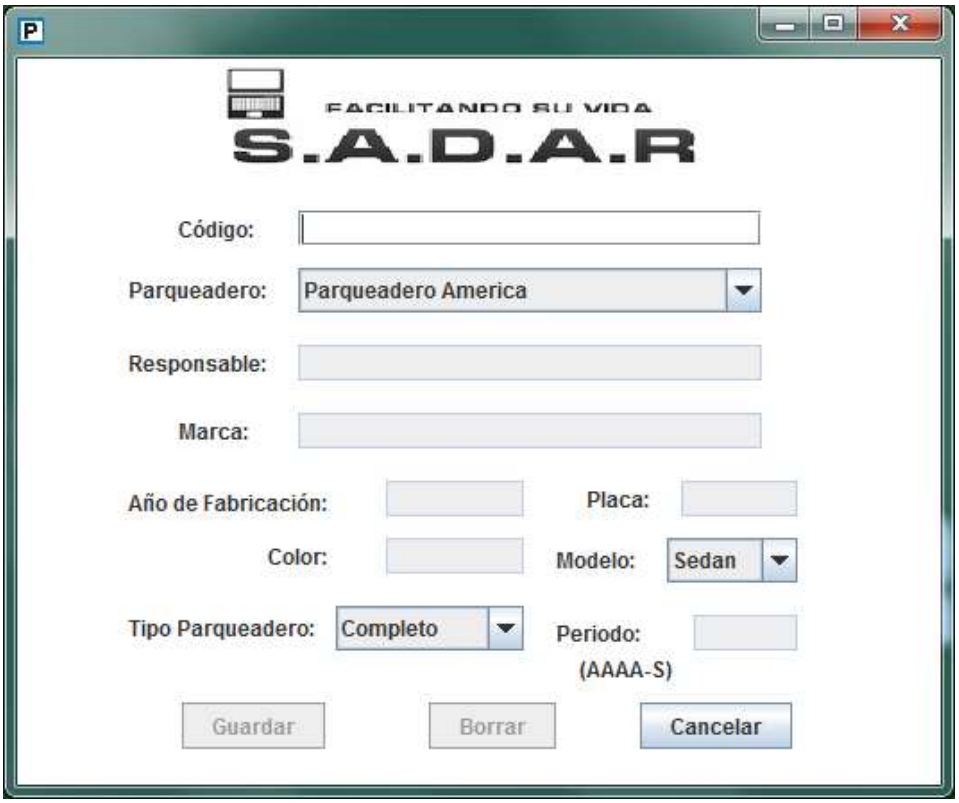
2. Presione “SI” en el caso de desear modificar dicho registro, en caso contrario presione “NO” para no modificar dicho registro y presione “Cancelar” en el caso de desear hacer alguna modificación al código previamente ingresado
3. Cambie todos los demás campos y presione guardar.
4. Un cuadro informativo notificara si desea modificar el registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.
5. En caso de afirmar la modificación del parqueadero se mostrará una pantalla informativa



Usuario Tipo: Administrador

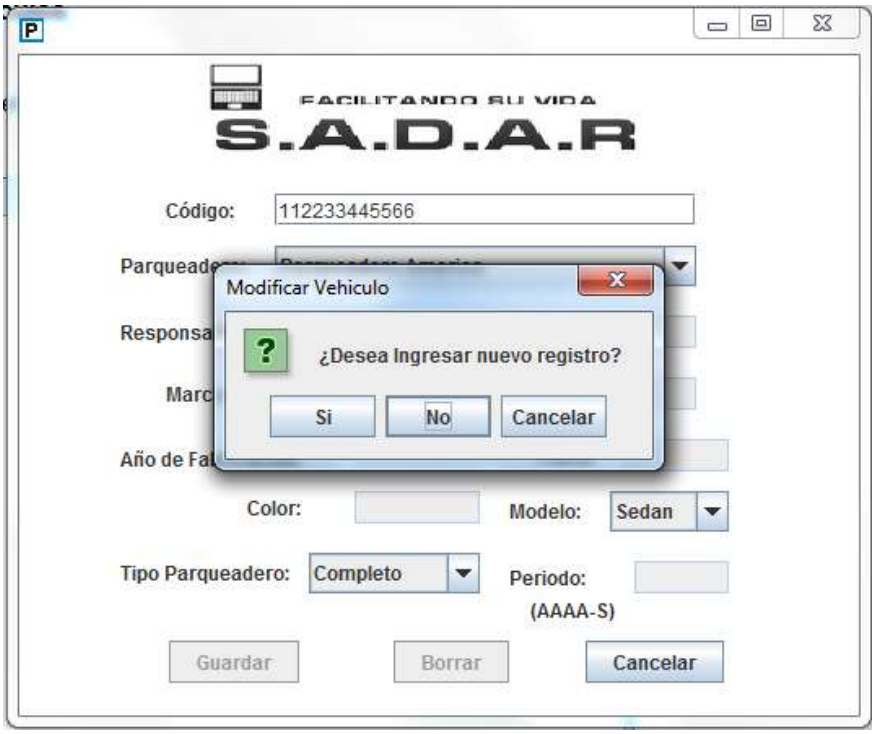
Administración Vehículos

Esta ventana permite el ingreso de nuevos vehículos al sistema.

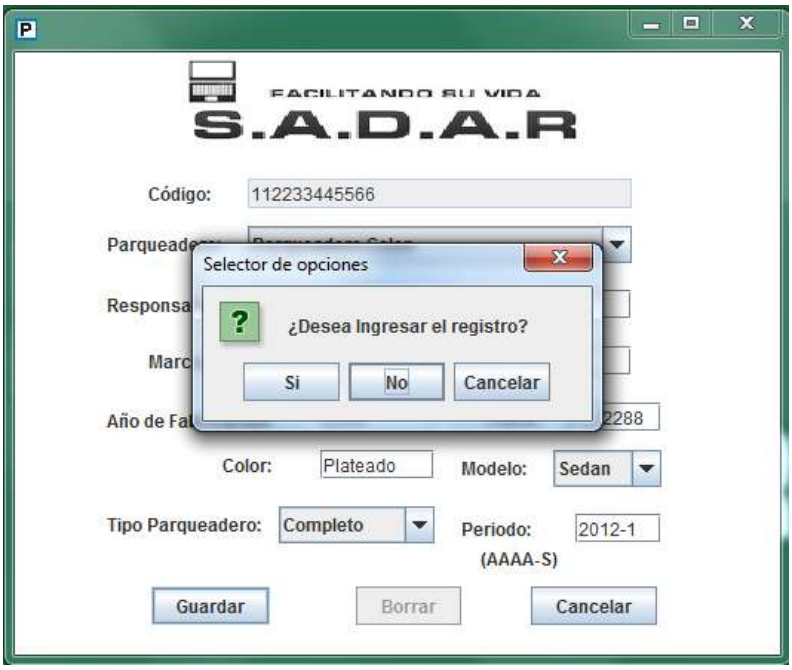


Ingreso Vehículo:

1. Digite el código del vehículo en el campo Código. En caso de no existir dicho código se presentara un mensaje de información de que si desea ingresar este nuevo parqueadero.



2. Si desea ingresar un nuevo registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.
3. Llene todos los demás campos y presione guardar.
4. Un cuadro informativo notificara si desea ingresar un nuevo registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.



En caso de afirmar el ingreso del nuevo vehículo se mostrará una pantalla informativa de que el espacio fue ingresado con éxito.



Modificación Vehículo:

1. Digite el código del usuario en el campo Código Usuario. En caso de existir dicho código se presentara un mensaje de información de que dicho código de usuario existe en la base de datos y si desea modificarlo.



2. Presione “SI” en el caso de desear modificar dicho registro, en caso contrario presione “NO” para no modificar dicho registro y presione “Cancelar” en el caso de desear hacer alguna modificación al código previamente ingresado
3. Cambie todos los demás campos y presione guardar.
4. Un cuadro informativo notificara si desea modificar el registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.
5. En caso de afirmar la modificación del vehículo se mostrará una pantalla informativa



Usuario Tipo: Administrador

Administración Espacios

Esta ventana permite el ingreso de nuevos espacios en los parqueaderos.



Ingreso Espacio:

1. Digite el código del espacio en el campo Código Espacio. En caso de no existir dicho código se presentara un mensaje de información de que si desea ingresar este nuevo parqueadero.



2. Si desea ingresar un nuevo registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.
3. Llene todos los demás campos y presione guardar.
4. Un cuadro informativo notificara si desea ingresar un nuevo registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.



En caso de afirmar el ingreso del nuevo vehículo se mostrará una pantalla informativa de que el parqueadero fue ingresado con éxito.



Modificación Parqueadero:

6. Digite el código del usuario en el campo Código Usuario. En caso de existir dicho código se presentara un mensaje de información de que dicho código de usuario existe en la base de datos y si desea modificarlo.



7. Presione “SI” en el caso de desear modificar dicho registro, en caso contrario presione “NO” para no modificar dicho registro y presione “Cancelar” en el caso de desear hacer alguna modificación al código previamente ingresado
8. Cambie todos los demás campos y presione guardar.
9. Un cuadro informativo notificara si desea modificar el registro confirme con “SI”, si no se desea guardar este registro presione “NO” y presione cancelar si desea hacer alguna modificación a todo lo que escribió.
10. En caso de afirmar la modificación del vehículo se mostrará una pantalla informativa



Usuario Tipo: Administrador y Guardia

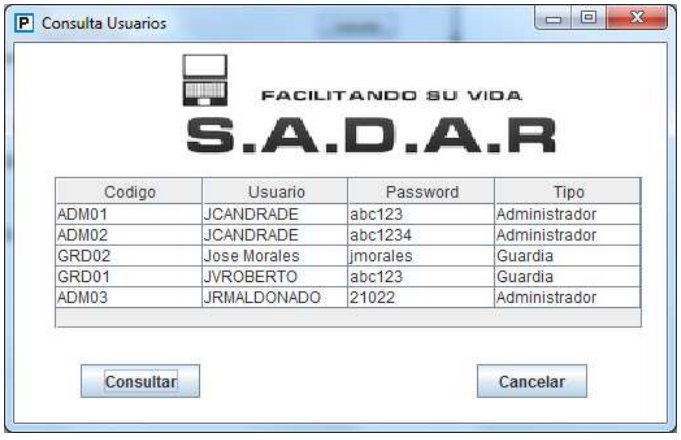
Consulta Usuarios

Esta ventana permite la consulta completa de todos los usuarios en la base de datos así como sus demás atributos.



En este módulo la consulta es completamente general a diferencia de los demás que se puede hacer por un parámetro pre establecido.

Presione Consultar para visualizar todos los usuarios.



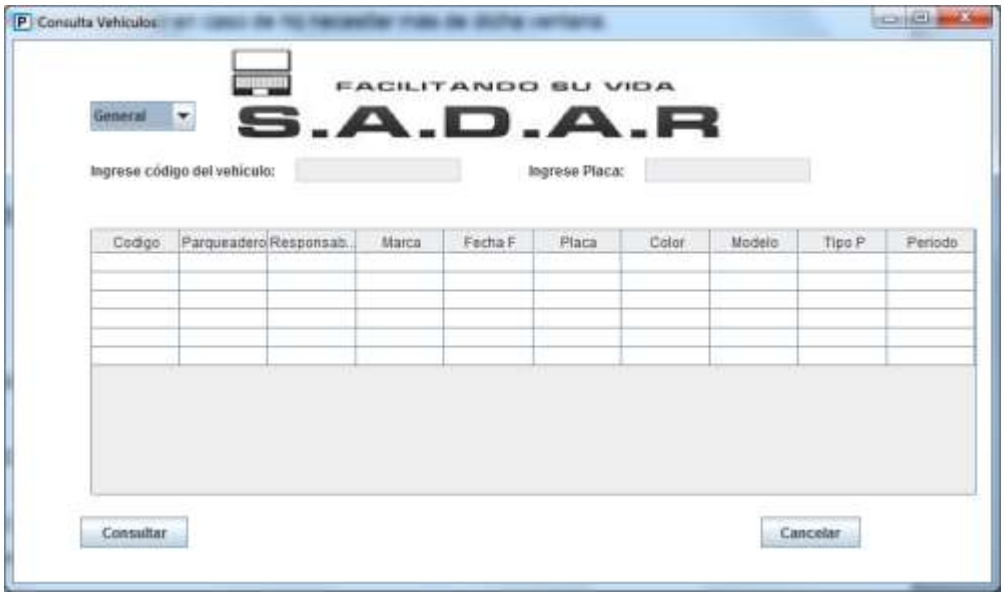
Presione cancelar en caso de no necesitar más de dicha ventana.

.

Usuario Tipo: Administrador y Guardia

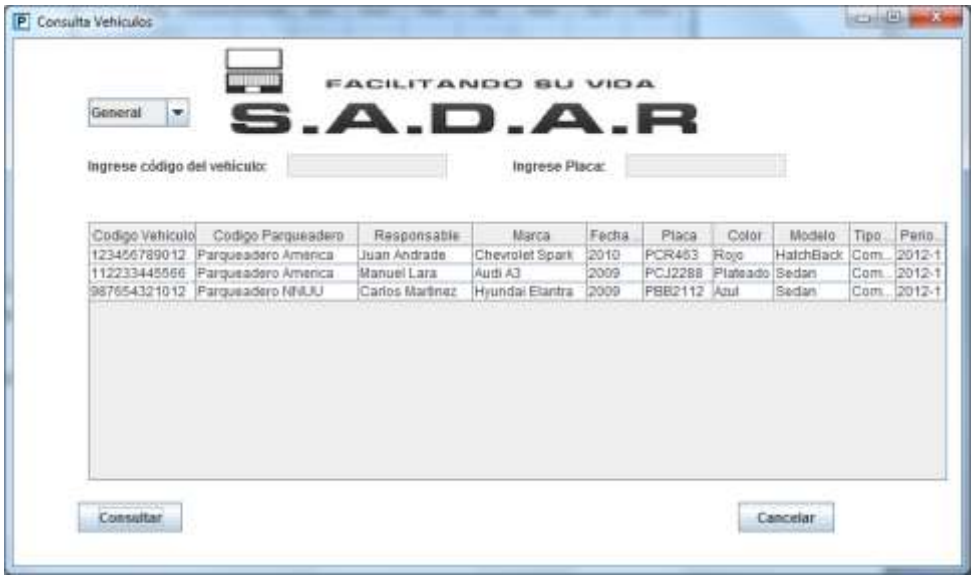
Consulta Vehículos

Esta ventana permite la consulta completa ya sea general o por parámetros de todos los registros de vehículos en la base de datos y de todos sus atributos.



Consulta General:

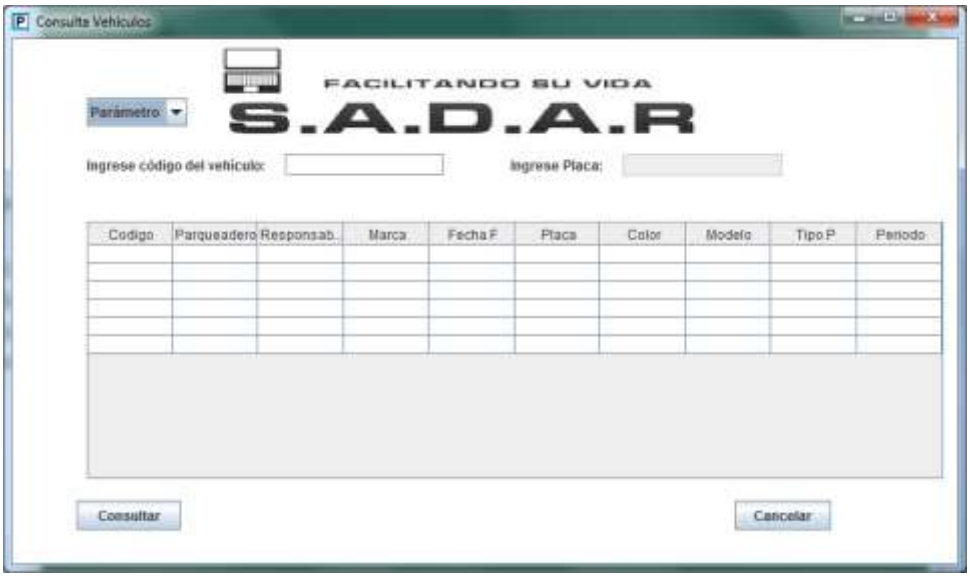
1. Seleccione Combo Box opción General.
2. Presione Consultar para visualizar todos los usuarios.



3. Presione cancelar en caso de no necesitar más de dicha ventana.

Consulta por Parámetro:

1. Seleccione Combo Box opción Parámetro.
2. Seleccione tipo parámetro de búsqueda.
3. Escriba dicho parámetro.



4. Presione Consultar
5. En caso de existir coincidencias con su parámetro de búsqueda sistema notifica que encontró dicho registro, caso contrario notifica no haber encontrado dicho registro que compagine con ese parámetro.



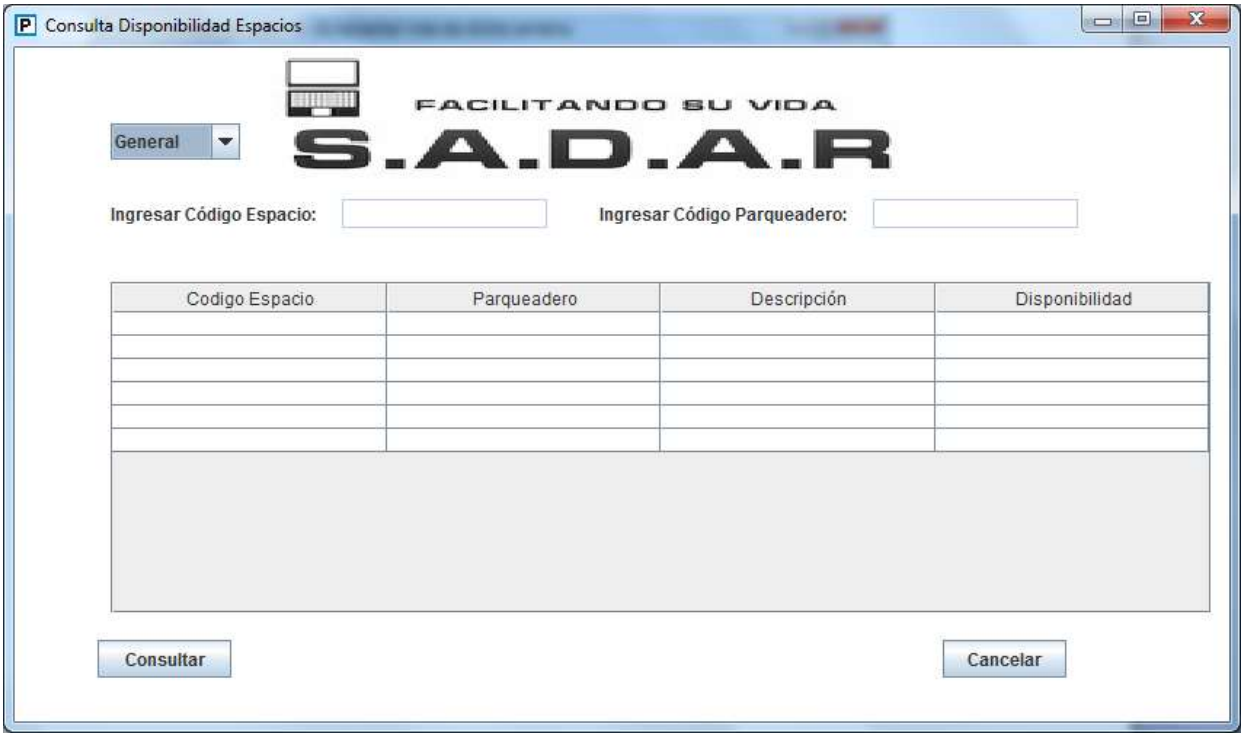
6. Presione Aceptar y desplegará la información



Usuario Tipo: Administrador y Guardia

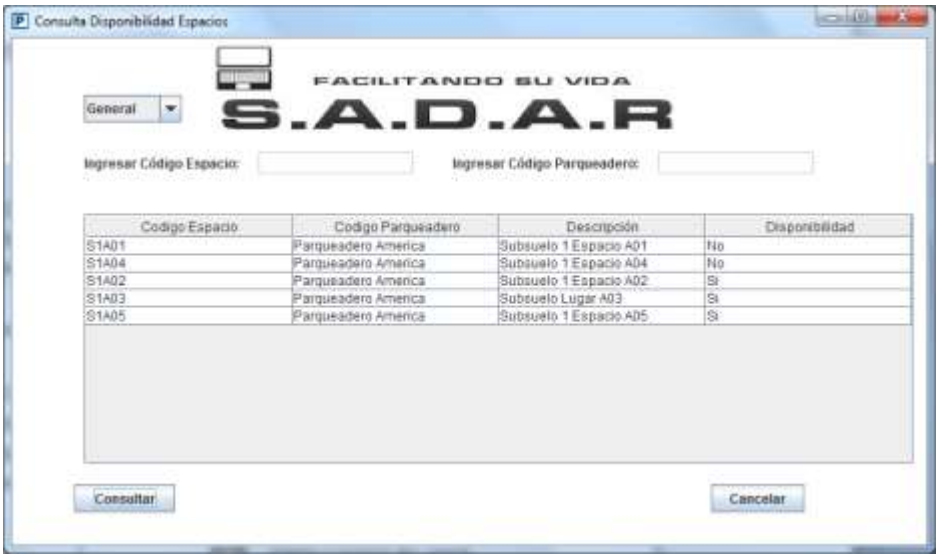
Consulta Espacios

Esta ventana permite la consulta completa ya sea general o por parámetros de todos los registros de espacios y parqueaderos en la base de datos y de todos sus atributos.



Consulta General:

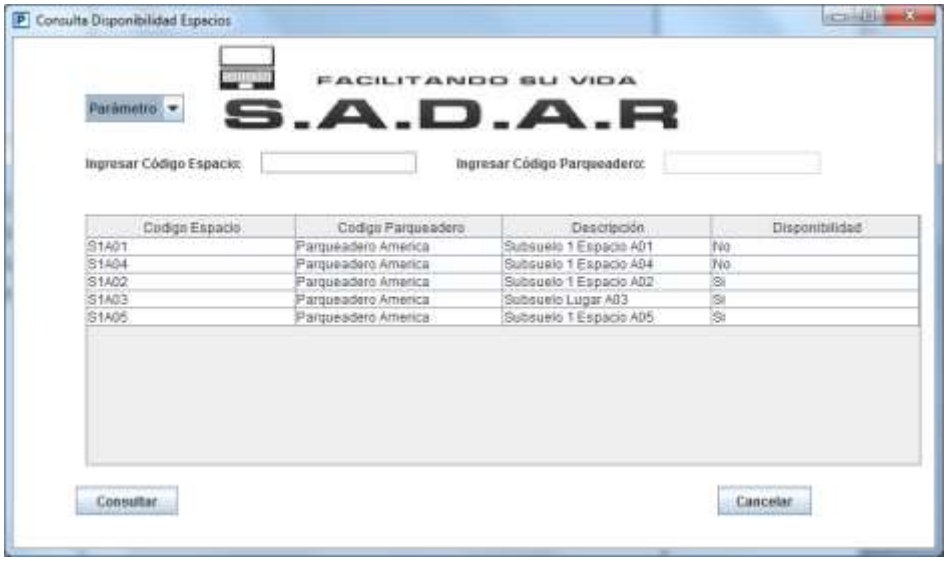
1. Seleccione Combo Box opción General.
2. Presione Consultar para visualizar todos los usuarios.



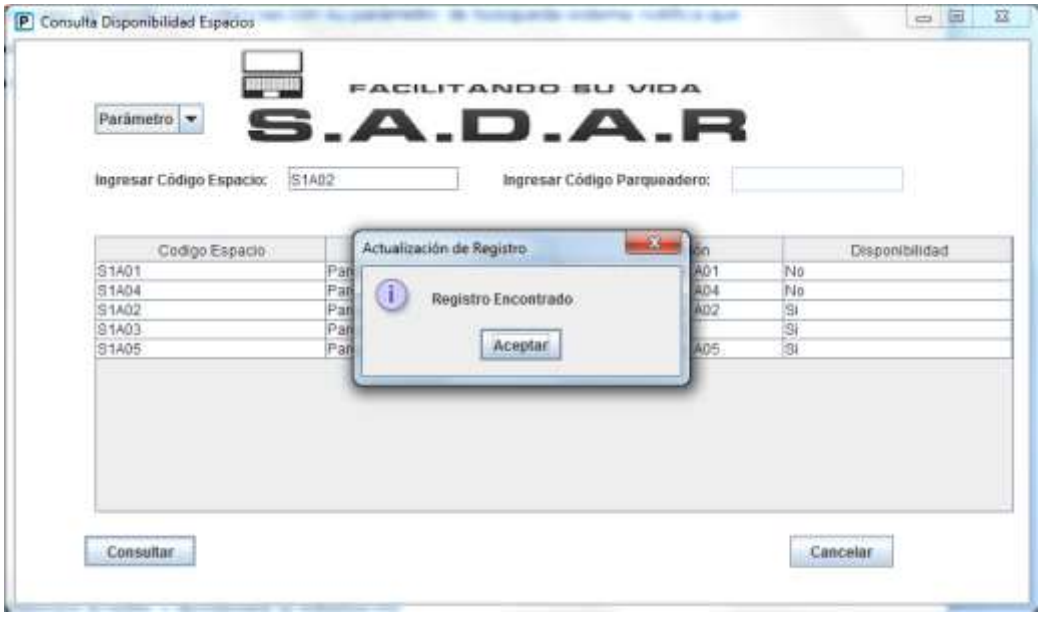
3. Presione cancelar en caso de no necesitar más de dicha ventana.

Consulta por Parámetro:

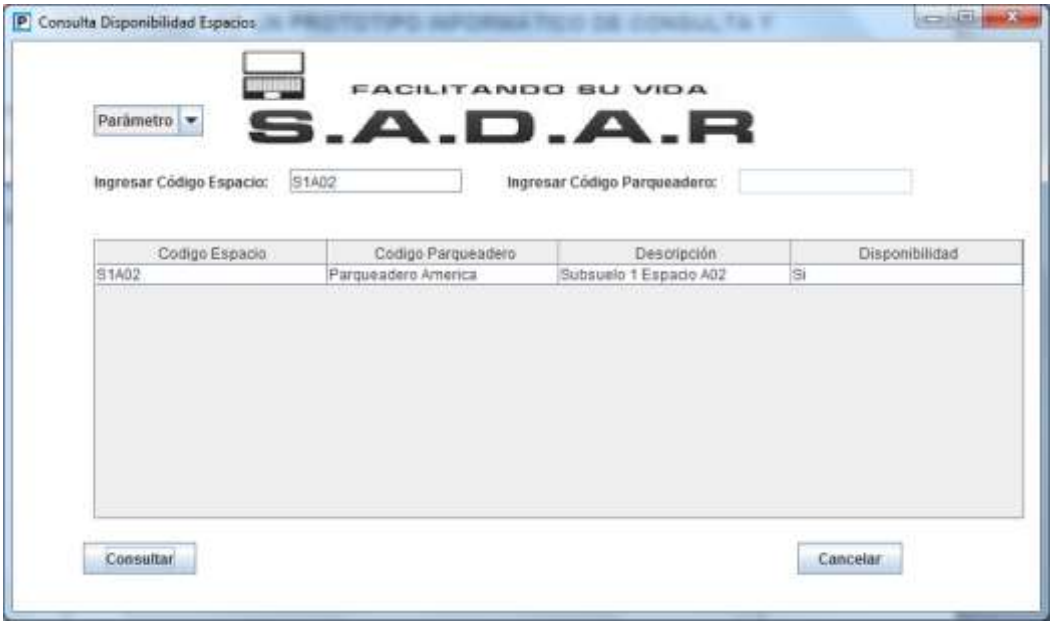
1. Seleccione Combo Box opción Parámetro.
2. Seleccione tipo parámetro de búsqueda.
3. Escriba dicho parámetro.



4. Presione Consultar
5. En caso de existir coincidencias con su parámetro de búsqueda sistema notifica que encontró dicho registro, caso contrario notifica no haber encontrado dicho registro que compagine con ese parámetro.



6. Presione Aceptar y desplegará la información.



Usuario Tipo: Administrador y Guardia

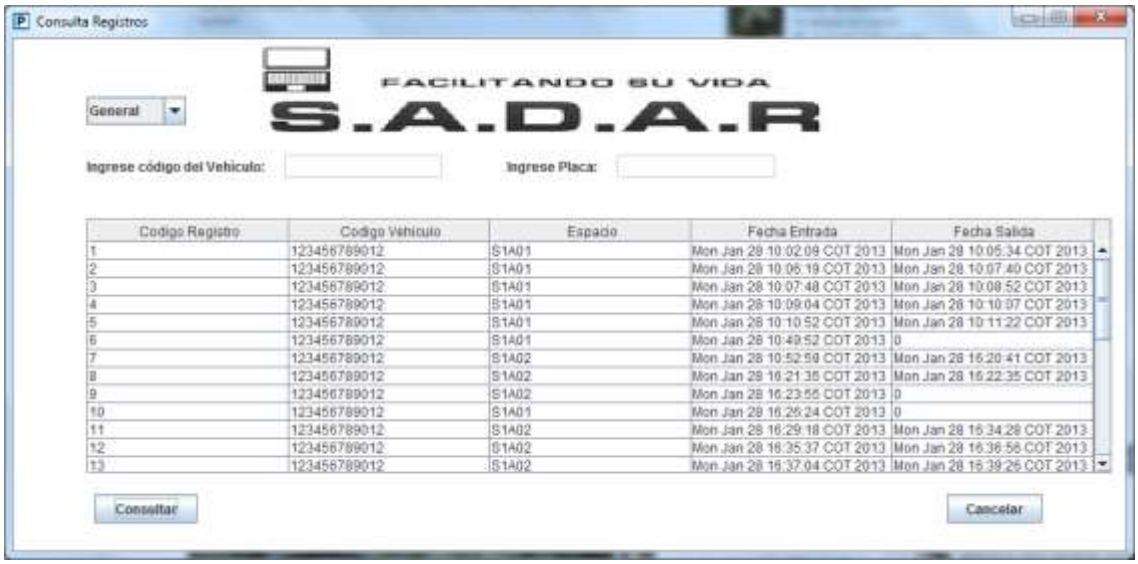
Consulta Registros

Esta ventana permite la consulta completa ya sea general o por parámetros de todos los registros de usos del parqueadero en la base de datos y de todos sus demás atributos.



Consulta General:

4. Seleccione Combo Box opción General.
5. Presione Consultar para visualizar todos los usuarios.

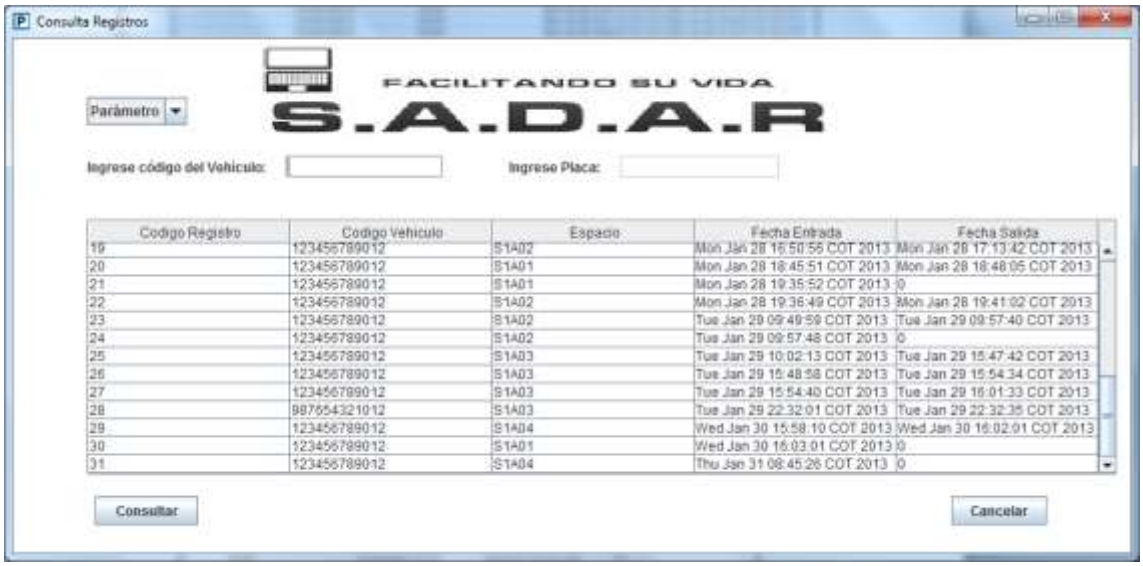


Codigo Registro	Codigo Vehiculo	Espacio	Fecha Entrada	Fecha Salida
1	123456789012	S1A01	Mon Jan 28 10:02:09 COT 2013	Mon Jan 28 10:05:34 COT 2013
2	123456789012	S1A01	Mon Jan 28 10:06:19 COT 2013	Mon Jan 28 10:07:40 COT 2013
3	123456789012	S1A01	Mon Jan 28 10:07:48 COT 2013	Mon Jan 28 10:08:52 COT 2013
4	123456789012	S1A01	Mon Jan 28 10:09:04 COT 2013	Mon Jan 28 10:10:07 COT 2013
5	123456789012	S1A01	Mon Jan 28 10:10:52 COT 2013	Mon Jan 28 10:11:22 COT 2013
6	123456789012	S1A01	Mon Jan 28 10:49:52 COT 2013	0
7	123456789012	S1A02	Mon Jan 28 10:52:59 COT 2013	Mon Jan 28 16:20:41 COT 2013
8	123456789012	S1A02	Mon Jan 28 16:21:35 COT 2013	Mon Jan 28 16:22:35 COT 2013
9	123456789012	S1A02	Mon Jan 28 16:23:55 COT 2013	0
10	123456789012	S1A01	Mon Jan 28 16:26:24 COT 2013	0
11	123456789012	S1A02	Mon Jan 28 16:29:18 COT 2013	Mon Jan 28 16:34:28 COT 2013
12	123456789012	S1A02	Mon Jan 28 16:35:37 COT 2013	Mon Jan 28 16:36:56 COT 2013
13	123456789012	S1A02	Mon Jan 28 16:37:04 COT 2013	Mon Jan 28 16:39:26 COT 2013

6. Presione cancelar en caso de no necesitar más de dicha ventana.

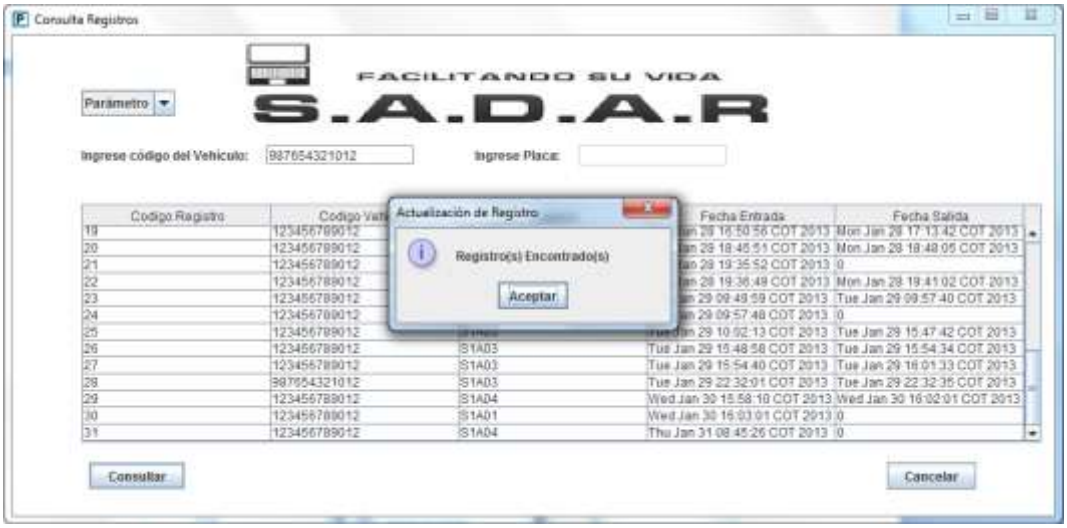
Consulta por Parámetro:

7. Seleccione Combo Box opción Parámetro.
8. Seleccione tipo parámetro de búsqueda.
9. Escriba dicho parámetro.



Codigo Registro	Codigo Vehiculo	Espacio	Fecha Entrada	Fecha Salida
19	123456789012	S1A02	Mon Jan 28 16:50:56 COT 2013	Mon Jan 28 17:13:42 COT 2013
20	123456789012	S1A01	Mon Jan 28 18:45:51 COT 2013	Mon Jan 28 18:48:05 COT 2013
21	123456789012	S1A01	Mon Jan 28 19:35:52 COT 2013	0
22	123456789012	S1A02	Mon Jan 28 19:36:49 COT 2013	Mon Jan 28 19:41:02 COT 2013
23	123456789012	S1A02	Tue Jan 29 09:49:59 COT 2013	Tue Jan 29 09:57:40 COT 2013
24	123456789012	S1A02	Tue Jan 29 09:57:48 COT 2013	0
25	123456789012	S1A03	Tue Jan 29 10:02:13 COT 2013	Tue Jan 29 15:47:42 COT 2013
26	123456789012	S1A03	Tue Jan 29 15:48:58 COT 2013	Tue Jan 29 15:54:34 COT 2013
27	123456789012	S1A03	Tue Jan 29 15:54:40 COT 2013	Tue Jan 29 16:01:33 COT 2013
28	987654321012	S1A03	Tue Jan 29 22:32:01 COT 2013	Tue Jan 29 22:32:35 COT 2013
29	123456789012	S1A04	Wed Jan 30 15:58:10 COT 2013	Wed Jan 30 16:02:01 COT 2013
30	123456789012	S1A01	Wed Jan 30 16:03:01 COT 2013	0
31	123456789012	S1A04	Thu Jan 31 08:45:26 COT 2013	0

10. Presione Consultar
11. En caso de existir coincidencias con su parámetro de búsqueda sistema notifica que encontró dicho registro, caso contrario notifica no haber encontrado dicho registro que compagine con ese parámetro.



12. Presione Aceptar y desplegará la información.



Usuario Tipo: Administrador

Proceso Asignar

Esta ventana permite ingresar y reservar un espacio dentro del parqueadero además de imprimir un ticket de reserva del espacio.



Al presentar su código de vehículo el sistema evaluará los espacios libres y se le accionará uno e imprimirá su ticket de asignación.



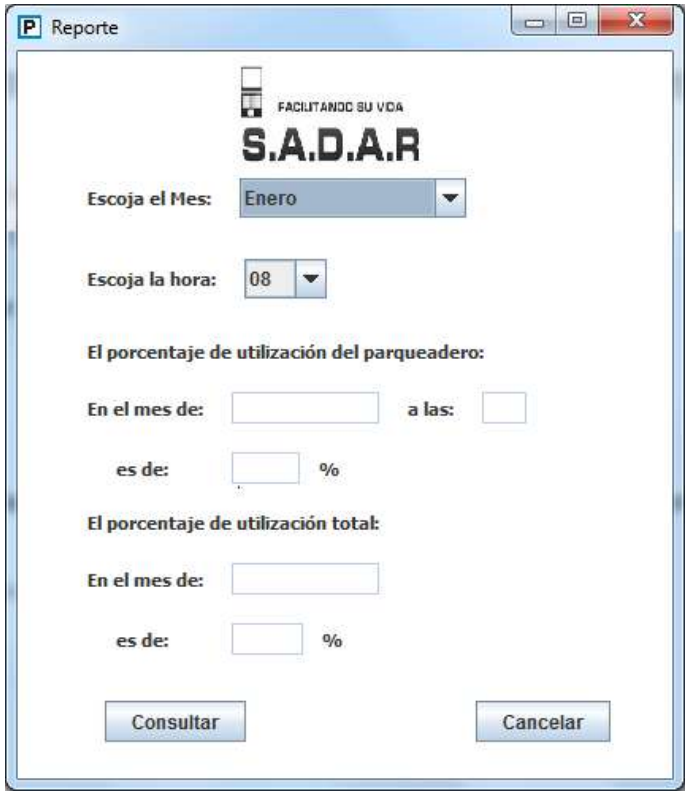
Ticket Impreso



Usuario Tipo: Administrador

Proceso Reporte

Esta ventana permite consultar un reporte de porcentaje de uso del parqueadero en cuanto a un delimitado mes y hora.
También calcula el porcentaje de uso total del parqueadero en dicho mes.



Presione Consultar y se realizará el cálculo de los promedios de uso del parqueadero según los parámetros que haya escogido.

