

PROYECTOS III

INGENIERÍA AUTOMOTRIZ

PLANIFICACIÓN DE PROYECTOS

A 28 DE AGOSTO DE 2025

Los estudiantes de ingeniería automotriz suelen tener formación teórica en áreas como chasis, frenos, motores y dirección, pero pocas oportunidades prácticas para aplicar estos conocimientos en proyectos reales debido a los altos costos de la industria. El desarrollo de un **prototipo funcional** ofrece una plataforma accesible para experimentar, diseñar y tomar decisiones reales, fortaleciendo la preparación integral del futuro ingeniero.

1 DESARROLLO DE GO-KART

Un Go-kart es un pequeño vehículo de carreras ligero y de bajo perfil, sin carrocería ni suspensión, diseñado para la competición en circuitos cortos y compactos o para la diversión recreativa

- El kart permite a los estudiantes aplicar los conocimientos aprendidos en clases (dinámica de sistemas, estructuras, suspensión, frenos, motores, electrónica embarcada) en un **prototipo tangible**, generando experiencia práctica y competencias reales.

¿CÓMO FUNCIONA?

El prototipo funciona mediante una estructura tubular (chasis) que sostiene todas las piezas: motor, asiento, sistema de dirección, ruedas y frenos.

Un motor—ya sea de gasolina o eléctrico—genera movimiento que, a través de una cadena o engranajes, impulsa las ruedas traseras.



SIRVE PARA:

El kart sirve como un laboratorio en movimiento. Permite a los estudiantes experimentar con conceptos reales de dinámica vehicular, distribución de peso, diseño estructural y selección de componentes. También fomenta la creatividad al buscar soluciones accesibles y prácticas, y refuerza la preparación profesional para enfrentar proyectos automotrices más complejos.

SE HACE:

El desarrollo del kart se divide en fases:

- **Diseño:** creación de planos, cálculos de resistencia del chasis, centro de gravedad, geometría de la dirección y elección de motor.
- **Construcción:** armado del chasis, fabricación de soportes, instalación de frenos, motor, transmisión, dirección y ruedas.
- **Pruebas y mejora:** realizar ensayos de seguridad, eficiencia y desempeño, detectando fallas y corrigiendo el prototipo.

MATERIALES BÁSICOS:

- **Chasis:** tubos de acero (ej. AISI 1018) para resistencia y bajo costo.
- **Motor:** motores de 6 a 13 HP de combustión interna, disponibles en Amazon o Mercado Libre.
- **Suspensión:** dependiendo del diseño, puede usarse rígida (más simple) o independiente (más avanzada).
- **Sistema de frenos:** discos mecánicos o hidráulicos de bajo costo.
- **Ruedas** llantas de karting o de motocicleta pequeña.
- **Otros:** volante, pedales, cojinetes, soportes metálicos, soldadora, tornillería y herramientas manuales.

INFORMACIÓN:

Artículos técnicos de **SAE International** (<https://www.sae.org/>).

Manuales y tutoriales de karting en **IAME Karting** (<https://www.iamekarting.com/>).

LIMITANTES:

La disponibilidad de recursos y materiales adecuados (como motores, chasis, frenos y ruedas) puede restringir el diseño y la construcción del Go-Kart, limitando su desempeño, seguridad y durabilidad durante las pruebas.

2 ECU DIDÁCTICA

Una ECU (Engine Control Unit), o Unidad de Control del Motor, es la computadora de un vehículo que actúa como el "cerebro" del motor, gestionando funciones vitales como la inyección de combustible, el encendido, el control de emisiones y otros parámetros para optimizar el rendimiento, la eficiencia y el cumplimiento de las normativas medioambientales.

- Una ECU didáctica permite simular señales, sensores y actuadores en un entorno seguro, facilitando el entendimiento práctico de cómo una ECU regula motores y sistemas automotrices para el desarrollo del estudiante.

¿CÓMO FUNCIONA?

La ECU didáctica opera mediante un microcontrolador (como Arduino o Raspberry Pi) que emula el comportamiento de una ECU real. A través de entradas (sensores virtuales) y salidas (actuadores virtuales), los estudiantes pueden interactuar con el sistema, modificar parámetros y observar resultados en tiempo real. Esto les permite experimentar con diferentes escenarios sin riesgo para equipos reales.

¿PARA QUÉ SIRVE?

Este proyecto tiene como objetivo proporcionar una herramienta educativa que permita a los estudiantes de ingeniería automotriz

- Comprender los principios teóricos y prácticos de los sistemas electrónicos automotrices.
- Desarrollar habilidades en diagnóstico, programación y análisis de sistemas electrónicos.
- Simular y experimentar con fallas y soluciones en un entorno seguro.
- Complementar la enseñanza teórica con prácticas interactivas.

LIMITANTES

La ECU didáctica no puede replicar completamente la complejidad de una ECU real, ni integrar todos los sensores y actuadores de un vehículo, y su capacidad de procesamiento limita la simulación de procesos avanzados.



SE HACE:

1. Diseño del Sistema

- Objetivo: Crear un esquema funcional que represente los sistemas electrónicos de un vehículo.
- Herramientas: Software de diseño como Fritzing o Tinkercad.
- Componentes: Selección de sensores, actuadores y microcontroladores adecuados.

2. Construcción del Prototipo

- Montaje: Ensamblaje de los componentes en una placa base o protoboard.
- Cableado: Conexión de los componentes según el diseño establecido.
- Programación: Codificación del microcontrolador para simular el comportamiento de la ECU.

3. Pruebas y Validación

- Escenarios: Creación de diferentes situaciones para evaluar el funcionamiento del sistema.
- Ajustes: Modificación de parámetros y códigos según los resultados obtenidos.
- Documentación: Registro detallado de las pruebas realizadas y sus resultados.

MATERIALES BÁSICOS:

- Microcontrolador: Arduino, Raspberry Pi o similar.
- Sensores: Temperatura, presión, posición, etc.
- Actuadores: LEDs, motores, relés, etc.
- Placa base: Protoboard o PCB.
- Cables y conectores: Para realizar las conexiones necesarias.
- Software: IDE de Arduino, Tinkercad, Fritzing.

INFORMACIÓN:

- **Fundamentos de ECUs:** Estudio de la arquitectura y funciones de una ECU en vehículos.
- **Sensores y actuadores automotrices:** Tipos, funcionamiento y aplicaciones.
- **Protocolos de comunicación:** CAN bus, LIN, etc.
- **Programación de microcontroladores:** Lenguajes como C/C++ para Arduino.

3 SISTEMA DE SILENCIADOR A ESCALA

Un **sistema de escape a escala** es una réplica reducida de un sistema de escape de un vehículo, incluyendo componentes como **colector, tuberías, silenciador y salida de gases**, diseñado para fines educativos o de demostración.

El **silenciador** es un dispositivo que reduce el ruido generado por los gases de escape del motor mediante la absorción, reflexión y expansión del flujo de gases.

- Con un modelo a escala, los estudiantes pueden simular el funcionamiento sin riesgo, usando aire comprimido o ventiladores pequeños, sin emisiones peligrosas.

¿CÓMO FUNCIONA?

1. Entrada de gases: Los gases provenientes del motor fluyen por el colector hacia la tubería de escape.
2. Reducción de ruido: En el silenciador, los gases atraviesan cámaras o tubos perforados que reflejan y dispersan las ondas sonoras.
3. Salida: Los gases se liberan al ambiente con menor ruido y, en sistemas reales, menor contaminación.

En una versión a escala, se puede simular con aire comprimido o pequeños ventiladores para demostrar flujo y efecto acústico sin usar gases de combustión

¿PARA QUÉ SIRVE?

- Educativo: Permite a los estudiantes comprender la función del escape y el silenciador, cómo se reduce el ruido y cómo se guía el flujo de gases.
- Demostrativo: Facilita el análisis de geometría de tuberías, pérdidas de presión y acústica.
- Investigativo: Permite experimentar con distintos diseños de silenciadores y tuberías para evaluar eficiencia y reducción de ruido.

SE HACE:

1. Diseño del sistema

- Dibujar el esquema del escape a escala: colector, tubería y silenciador.
- Definir dimensiones proporcionales al motor o modelo que simula.

2. Construcción

- Silenciador: Tubo con cámaras internas y perforaciones que simulen absorción acústica.
- Tuberías: Se pueden usar tubos de PVC o metal delgado.
- Colector: Tubo adaptado para unir la entrada de "gases" o aire.

3. Pruebas

- Hacer fluir aire con bomba o compresor pequeño para observar la dispersión del flujo y reducción de ruido.
- Comparar diseños de silenciador: recto, con cámaras de expansión o perforado.

MATERIALES BÁSICOS:

- Tubos de PVC o aluminio (para tuberías y silenciador)
- Tubo perforado o esponja metálica (para la absorción sonora dentro del silenciador)
- Adhesivos resistentes al calor (si se usan materiales metálicos)
- Compresor o bomba de aire pequeña (para simular flujo)
- Soportes y base de madera o acrílico (para mantener el sistema fijo)

INFORMACIÓN:

- Principios de **acústica** en motores.
- Funcionamiento del **silenciador y resonador**.
- Dinámica de **flujo de gases** en tuberías.
- Materiales resistentes al calor y vibración.
- Escalado de sistemas mecánicos para modelos reducidos.

LIMITANTES:

El sistema de escape a escala no simula con exactitud el flujo de gases ni el ruido de un motor real, y los materiales y estructura del modelo no soportan condiciones extremas.

BIBLIOGRAFÍA

- Burgess, A. T. (s.f.). *Kart design and construction*. [Editorial desconocida].
- Quezada Poma, P. A. (2018). *Diseño y fabricación del chasis para un kart KF4 según la normativa CIK/FIA* (Trabajo de titulación, Universidad Politécnica Salesiana). <https://dspace.ups.edu.ec/bitstream/123456789/15766/1/UPS-CT007735.pdf>
- Pulgarin Vicuña, V. (2024). *Implementación de un banco didáctico con panel de control para la enseñanza de sistemas electrónicos automotrices*.
- IAME Karting. (s.f.). *IAME Karting - Endless Racing*. <https://www.iamekarting.com/>



INGENIERÍA

PROYECTOS III

PROBLEMA DE INVESTIGACIÓN

INGENIERÍA AUTOMOTRIZ

SIMULADOR INTERACTIVO DE DIAGNÓSTICO AUTOMOTRIZ PARA FORMACIÓN INICIAL

SIMCA

SIMULADOR DE MANTENIMIENTO Y CAPACITACIÓN AUTOMOTRIZ



ANTECEDENTES DE LA INVESTIGACIÓN

La formación en ingeniería automotriz ha cambiado profundamente con el paso del tiempo. Antes, el aprendizaje se centraba en el taller de manera empírica, donde muchos de los estudiantes desmontaban motores y seguían manuales técnicos, un método práctico pero limitado y costoso. Con la llegada de la electrónica automotriz en los años 80, surgió la necesidad de dominar nuevas herramientas como el multímetro y el escáner. Esto transformó la pedagogía, incorporando maquetas y bancos de pruebas para acercar la teoría a la práctica.

Con el paso del tiempo, la digitalización abrió la puerta a los primeros simuladores de software, inspirados en los de aviación. Aunque básicos, permitieron practicar diagnósticos eléctricos sin dañar vehículos. En la actualidad México persiste una brecha muchos técnicos automotrices empíricos dominan la práctica tradicional, pero enfrentan retos al adaptarse a la autotrónica.

Buscando una solución, los simuladores de se presentan como una alternativa entre la experiencia práctica y el mundo digital. Más que una herramienta, buscan ser un recurso visual, interactivo y accesible que acompañe principalmente a los estudiantes en sus primeros pasos y redefina la enseñanza automotriz para una nueva generación.

PALABRAS CLAVE

MANTENIMIENTO PREVENTIVO

Revisiones y cambios programados para evitar fallas (ej. aceite, filtros, frenos).

APRENDIZAJE KINESTÉSICO

Aprender practicando o manipulando.

GAMIFICACIÓN

Usar dinámicas de juego (puntos, logros, niveles) para aprender de forma más motivante.

MVP

MVP: Versión mínima de un proyecto que muestra cómo funciona la idea.

INVESTIGACIÓN PREVIA

Este proyecto nace de la necesidad de enseñar mecánica y diagnóstico automotriz de una manera distinta. Tradicionalmente, los estudiantes aprendían en el taller, desmontando piezas y usando manuales llenos de diagramas. Ese método funciona, pero tiene límites: no siempre hay autos disponibles, es caro dañar componentes y muchos conceptos teóricos se sienten lejanos a la práctica.

Hoy en día, ya existen simuladores como **Electude**, que son muy buenos y realistas. Con ellos se pueden hacer diagnósticos complejos usando herramientas virtuales, como un osciloscopio o un escáner. Pero estos programas no están pensados para principiantes, sino para alumnos que ya saben lo básico. Ese es el hueco que este proyecto quiere cubrir: una herramienta de nivel inicial que ayude a comprender lo más sencillo de manera clara, visual e interactiva.

Muchos temas en automoción son complicados de explicar con métodos tradicionales. Por ejemplo, un maestro puede dibujar en el pizarrón cómo cambia la señal de un sensor TPS, pero es difícil que el estudiante lo entienda solo con la teoría.

SIMCA

CAPCITACIÓN AUTOMOTRIZ

En cambio, si ese alumno abre un acelerador virtual en un simulador y ve cómo cambia el voltaje en una gráfica o cómo las revoluciones del motor se vuelven inestables cuando hay una falla, la comprensión es mucho más natural. Lo mismo pasa con la afinación: cambiar un filtro de aire sucio por uno nuevo y ver cómo sube la eficiencia del motor motiva más que solo leerlo en un libro. Incluso puede aprender cómo reiniciar la luz de servicio después de un cambio de aceite, algo que en la vida real es confuso al inicio.

El proyecto también toma fuerza con la **gamificación**, es decir, aplicar mecánicas de juego al aprendizaje. La idea es que el estudiante no solo lea y memorice, sino que participe, experimente y reciba retroalimentación inmediata. Por ejemplo, cambiar una bujía y escuchar cómo mejora el motor en ese momento, o pasar de un nivel básico (apagar una luz de servicio) a uno más avanzado (diagnosticar un código de error usando un multímetro y un escáner). Con este sistema progresivo, se gana confianza paso a paso, sin miedo a equivocarse ni a dañar un auto real.

En cuanto a la parte técnica, la propuesta es usar el motor de videojuegos **Godot Engine**, porque es gratuito, fácil de aprender y está pensado para proyectos tanto en 2D como en 3D. Cada parte del coche (batería, bujía, filtro de aire, sensor TPS) será un objeto con propiedades como voltaje o desgaste, y un "motor central" del programa se encargará de calcular en tiempo real el resultado de las acciones del usuario. Este diseño modular permite crecer poco a poco y mantener el proyecto estable.



GODOT

Game engine

Godot Engine es un motor de desarrollo de videojuegos gratuito y de código abierto que permite crear juegos 2D y 3D de forma multiplataforma para PC, móvil y web.

Para empezar, no es necesario que el proyecto cree un auto completo desde cero. Lo más recomendable es definir un MVP (producto mínimo viable), es decir, un prototipo pequeño que funcione bien. Puede iniciar con el sistema de arranque (batería, motor de arranque y solenoide) y hacerlo totalmente interactivo. Una vez que ese módulo esté sólido, se pueden agregar otros como el encendido o el sensor TPS.

Antes de programar, resulta importante planear y diseñar la interfaz en papel. Esto permite que el simulador sea claro, atractivo y fácil de usar, ya que, si la interfaz es confusa, el estudiante pierde interés.

El proyecto también se apoya en la idea de la gamificación de la educación. Se ha demostrado que aplicar mecánicas de juego como niveles, logros y retroalimentación rápida aumenta la motivación y ayuda a que el aprendizaje sea más duradero. En habilidades prácticas como la mecánica automotriz, la simulación funciona como un paso previo muy eficaz antes de enfrentarse a un motor real.

En pocas palabras, este proyecto no solo busca crear un software, sino abrir una nueva forma de aprender lo automotriz: más visual, más interactiva y cercana al lenguaje digital que domina la vida actual de los estudiantes. Su propósito es conectar lo práctico con lo digital, reduciendo la brecha entre la experiencia empírica y las nuevas tecnologías del diagnóstico automotriz.

SIMCA

PEDAGÓGICO



Atiende a estudiantes que aprenden mejor viendo y experimentando, convirtiendo la teoría en algo práctico y fácil de recordar a través de un "laboratorio virtual".

MERCADO



Cubre un espacio poco atendido: la formación inicial. No busca reemplazar a plataformas avanzadas sino ser un puente para principiantes y un recurso económico para universidades y centros de capacitación.

OPORTUNIDADES



En México y Latinoamérica, representa una forma de digitalizar la formación empírica, ofreciendo a mecánicos y estudiantes una herramienta simple, en español y de bajo costo que ayuda a enfrentar los retos de la electrónica automotriz.

JUSTIFICACIÓN DE SIMCA

Este proyecto se justifica porque puede impactar en tres puntos clave: mejorar cómo aprenden los estudiantes, hacer más accesible la formación técnica y alinearse con la tecnología actual de la industria automotriz.

En cuanto al aprendizaje, el simulador ayuda a dejar atrás la enseñanza pasiva, donde los alumnos solo leen manuales o escuchan clases. Con él, los estudiantes pasan de ser espectadores a participantes activos, ya que pueden experimentar, cometer errores y ver las consecuencias de inmediato. Esto acelera la forma de aprender y fomenta el pensamiento crítico y la resolución de problemas.

También tiene un fuerte valor social y económico. Estudiar mecánica automotriz de forma práctica suele ser caro: se necesitan talleres, autos, refacciones y herramientas que se desgastan o se dañan con facilidad. Un simulador en computadora reduce mucho esas barreras, ya que se puede usar incluso en equipos de gama media. En México, esto significa que muchos estudiantes y técnicos podrían capacitarse de manera más barata y accesible, mejorando sus oportunidades laborales y elevando la calidad del servicio en el país.

Por último, este proyecto está totalmente conectado con el rumbo de la industria. Hoy los autos son prácticamente computadoras con ruedas: traen electrónica, sensores, ADAS y sistemas complejos que requieren técnicos con habilidades digitales. Con este simulador, los estudiantes no solo aprenden mecánica, sino también a manejar software, interpretar datos y comprender la lógica detrás de los sistemas electrónicos. Esto los prepara para la industria automotriz del futuro, donde lo digital será lo normal.

OBJETIVO GENERAL

Desarrollar un prototipo funcional de un simulador interactivo 2D para computadora, denominado "SIMCA", que enseñe los conceptos fundamentales de **mantenimiento preventivo (afinación)** y diagnóstico básico de los sistemas de **arranque, encendido y control de aire (a través del sensor TPS)** de un motor de combustión interna, dirigido a estudiantes de nuevo ingreso de ingeniería automotriz y técnicos en formación.

OBJETIVOS ESPECIFICOS

- Investigar y definir los requisitos del simulador para asegurar que cumpla con lo necesario.
- Diseñar la arquitectura y la interfaz de manera clara y atractiva para los usuarios.
- Desarrollar el módulo del sistema de arranque, incluyendo batería, solenoide y motor de arranque.
- Crear el módulo del sistema de encendido para simular el funcionamiento básico.
- Implementar un módulo de mantenimiento preventivo que permita cambiar aceite, filtros y bujías con un clic, además de un sistema de "desgaste" que afecte el rendimiento y active una luz de advertencia.
- Construir el módulo de control de aire, con una mariposa de acelerador 2D y el sensor TPS, mostrando cómo influye en las RPM del motor.
- Integrar herramientas de diagnóstico y servicio, como un multímetro virtual y un escáner para leer códigos de falla (P0122, P030X) y resetear la luz de mantenimiento.
- Realizar pruebas de usabilidad y funcionalidad para garantizar que el simulador sea práctico y fácil de usar.

2.7. Pregunta de Investigación

¿En qué medida el uso de un simulador interactivo (SIMCA) que integra procedimientos de mantenimiento preventivo y diagnóstico de sistemas básicos mejora la comprensión conceptual y la confianza en la ejecución de tareas automotrices fundamentales en estudiantes de primer año, en comparación con métodos de enseñanza tradicionales?



INGENIERÍA

SIMCA

SIMULADOR DE MANTENIMIENTO Y CAPACITACIÓN AUTOMOTRIZ



INGENIERÍA
CAPITULO III
COTIZACIÓN

3.1 MATERIALES

Para el desarrollo del prototipo del simulador **SIMCA**, se identifican los siguientes recursos:

HARDWARE MÍNIMO REQUERIDO

- Computadora de escritorio o laptop con procesador de al menos 4 núcleos, 8 GB de RAM y tarjeta gráfica integrada o dedicada.
- Monitor de 21" o superior para pruebas de interfaz.
- Mouse y teclado.
- Almacenamiento de 500 GB (HDD o



SOFTWARE Y ENTORNOS DE DESARROLLO

- **Godot Engine 4.x** (motor gráfico de código abierto y gratuito).
- **Lenguaje GDScript** (lenguaje nativo de Godot).
- **Inkscape o GIMP** (para diseño de elementos gráficos 2D).
- **Blender** (para modelado de componentes si se requiere 3D).
- **Visual Studio Code** (editor de código complementario).

PARA LA PRESENTACIÓN FINAL

- Pantalla (para la exposición).
- Carpeta impresa con documentación del proyecto.
- Fotografías y capturas de pantalla del simulador en funcionamiento.



3.2 COTIZACIÓN

Para el desarrollo del simulador **SIMCA** se requiere principalmente de programas y complementos de apoyo, además de los servicios de impresión y encuadernado del material final. A continuación, se presentan las opciones de proveedores y costos aproximados:

1. **GODOT ENGINE 4.X**

- Motor de desarrollo principal.
- Descarga gratuita desde la página oficial: <https://godotengine.org>.
- **Costo: \$0 MXN.**

2. **VISUAL STUDIO CODE**

- Editor de código para organización de scripts.
- Descarga gratuita desde <https://code.visualstudio.com>.
- **Costo: \$0 MXN.**

3. **GIMP E INKSCAPE**

- Programas para la edición de imágenes y diseño de gráficos en 2D.
- Descarga gratuita desde <https://www.gimp.org> y <https://inkscape.org>.
- **Costo: \$0 MXN.**

4. **BLENDER (OPCIONAL)**

- Software de modelado 3D para incorporar elementos gráficos adicionales si es necesario.
- Descarga gratuita INSTITUCIONAL desde <https://www.blender.org>.
- **Costo: \$0 MXN.**

5. **BIBLIOTECA DE RECURSOS GRÁFICOS**

- **Flaticon Premium (anual):** \$700 MXN.
- **Envato Elements (mensual):** \$450 MXN.



3.3 COMPARACIÓN DE PRECIOS

| MOTOR | VENTAJAS | DESVENTAJAS | COMENTARIOS |
|---|--|--|---|
| Unity | Amplio ecosistema, Asset Store muy completo, fuerte comunidad, muchas herramientas disponibles (física avanzada, render 3D, plugins) | Curva de aprendizaje más pronunciada para principiantes; puede requerir licencias o versiones de pago avanzadas; sobrecarga de funciones que no se utilizarán; puede consumir más recursos | Se usa mucho en la industria, pero en proyectos pequeños o de simulación básica puede ser más complejo de lo necesario. |
| Unreal Engine | Potente en gráficos de alta fidelidad, ideal para simulaciones 3D avanzadas y visuales realistas | Su enfoque es más complejo, usa C++ o exige hardware más robusto, y su uso puede resultar excesivo para simulaciones 2D o lógicas básicas | No es la opción más eficiente para un proyecto de simulador automotriz básico en 2D con lógica de comportamiento, pues la potencia gráfica no será el factor crítico. |
| Motores más ligeros / frameworks 2D (ej. libGDX, Phaser, Godot 2D partiendo de motores web) | Ligereza, flexibilidad, control fino sobre la lógica del simulador | Tienen que integrar componentes por separado (motor de física, render, manejo de escenas) lo que implica más trabajo de infraestructura | Pueden servir para prototipos, pero el desarrollo será más artesanal |
| Godot Engine | Muy ligero, orientado tanto a 2D como a 3D, arquitectura de nodos/escenas intuitiva, scripts con GDScript fáciles | En algunos casos, las herramientas 3D aún no alcanzan la madurez de Unity/Unreal; menor | Para un simulador educativo 2D con lógica de componentes, sus ventajas de usabilidad, rapidez de prototipado, |

| | | | |
|--|-------------------------------------|---------------------------------------|---|
| | de aprender, gratuito y open-source | cantidad de assets comerciales listos | peso ligero y costo cero lo hacen muy adecuado. |
|--|-------------------------------------|---------------------------------------|---|

Para fundamentar la elección de **Godot Engine** en el desarrollo del simulador **SIMCA**, se comparan otros motores de desarrollo relevantes y se muestran las ventajas relativas de Godot para un proyecto de simulación en 2D.

Ventajas específicas de Godot para SIMCA

1. Ligereza y rendimiento en 2D

Godot ofrece un motor dedicado de 2D que no requiere proyectar escenas 3D para simular entornos bidimensionales. Esto redundará en una menor carga de procesamiento y mejor rendimiento, especialmente en equipos modestos.

2. Curva de aprendizaje suave con GDScript

El lenguaje GDScript tiene una sintaxis parecida a Python, lo que facilita su aprendizaje para estudiantes. Esto permite que el equipo dedique más tiempo al diseño de la simulación y menos a luchar con la complejidad del lenguaje.

3. Arquitectura basada en nodos y escenas reutilizables

La forma en que Godot organiza los elementos (nodos, escenas "anidadas") permite componer módulos de simulación (por ejemplo, motor, sistema de encendido, diagnóstico) de forma modular y escalable.

4. Gratuidad y comunidad abierta

Al ser open-source y sin licencias comerciales obligatorias, no hay costos ocultos ni restricciones para distribuir versiones de prueba o institucionales. Además, la comunidad aporta mejoras, plugins y soluciones que pueden integrarse al proyecto.

3.4 PRESUPUESTO

Con base en lo anterior, el presupuesto final se estima de la siguiente forma:

- Software base (Godot, VS Code, GIMP, Inkscape, Blender): \$0 MXN.
- Biblioteca de recursos gráficos (FlatIcon Premium anual o Envato mensual): \$450 - \$700 MXN.

Costo final estimado del proyecto: \$450 a \$700 MXN.

El rango depende de la plataforma de recursos gráficos seleccionada. En caso de trabajar únicamente con programas gratuitos, el costo total sería de \$0 MXN.



3.5 COMPRA Y VERIFICACIÓN DE MATERIALES

los programas se descargarán desde sus páginas oficiales para garantizar seguridad y autenticidad. En el caso de adquirir recursos gráficos como Flaticon o Envato, se conservarán los comprobantes digitales de compra como evidencia.

La verificación de los materiales se realizará mediante capturas de pantalla que muestren:

- La instalación correcta de Godot y su ejecución inicial.
- La utilización de GIMP e Inkscape en la creación de íconos y gráficos.
- La incorporación de recursos descargados desde Flaticon o Envato en el diseño del simulador.
- Avances del simulador en funcionamiento dentro del motor Godot.

Estas evidencias servirán para confirmar que los programas y recursos adquiridos fueron efectivamente aplicados en el desarrollo del prototipo.



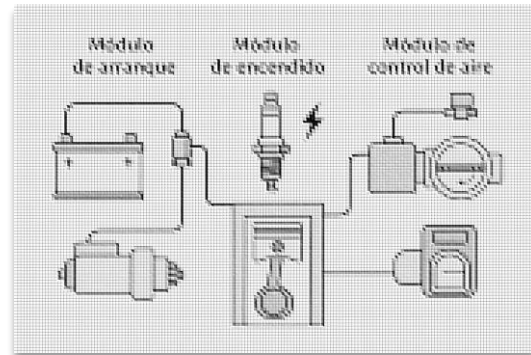
INGENIERÍA
CAPITULO IV
DISEÑO

CAPÍTULO IV. DISEÑO Y CÁLCULOS

4.1 Planeación del diseño

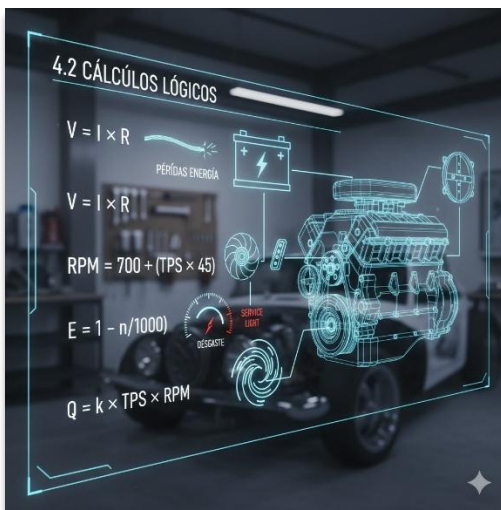
Se definió el diseño modular del simulador "SIMCA", compuesto por tres subsistemas principales:

- Módulo de arranque: incluye batería, motor de arranque y solenoide.
- Módulo de encendido: simula el sistema de chispa (bujías, bobina y señales eléctricas).
- Módulo de control de aire (TPS): representa la mariposa del acelerador y su



Cada módulo se conecta con un motor central de simulación dentro del motor de videojuegos Godot Engine, encargado de procesar los valores eléctricos y mecánicos virtuales (voltaje, RPM, temperatura, etc.). El diseño se realizó inicialmente en bocetos 2D y posteriormente en el entorno digital, cuidando una interfaz clara, didáctica y con retroalimentación visual inmediata (colores, sonidos y mensajes).

4.2 Cálculos



Estos cálculos permiten que el simulador reaccione de manera coherente ante fallas o

Aunque se trata de un simulador digital, se contemplan cálculos lógicos para representar fenómenos reales:

- Voltaje del sistema eléctrico: $V = I \times R$ (para representar pérdidas de energía y fallos eléctricos simulados).
- Relación de RPM - ángulo del acelerador: $RPM = 700 + (TPS \times 45)$, donde el valor del TPS varía entre 0 y 100 %.
- Desgaste de componentes: $E = 1 - (n/1000)$, donde n es el número de ciclos de uso virtual. Cuando $E < 0.7$, el sistema activa una "luz de servicio".
- Flujo de aire virtual: $Q = k \times TPS \times RPM$.

4.3 Bocetaje a mano

Se elaboraron bocetos a mano en perspectiva frontal, lateral y superior, que incluyen:

- Distribución de módulos: panel principal, motor virtual, tablero de diagnóstico y herramientas (multímetro y escáner).
- Dimensiones base: proporción 16:9 para pantalla; botones y menús escalados al 10 % del ancho de la interfaz.
- Detalles: iconos de batería, filtros, bujías y mariposa de aceleración con sus respectivos indicadores.



(Aquí se incluirán las imágenes o dibujos escaneados del bocetaje con medidas y etiquetas de cada componente).

4.4 Lista de piezas principales y secundarias



Principales (virtuales y físicas):

- Computadora con sistema operativo Windows/Linux.
 - Software Godot Engine (motor de desarrollo).
 - Imágenes y sprites de componentes automotrices (batería, bujía, filtro, TPS, escáner, etc.).
 - Scripts en lenguaje GDScript para la simulación.
- Secundarias (complementarias):
- Biblioteca de sonidos (ruido de motor, clics, avisos).
 - Interfaz de usuario (botones, menús, fondo).
 - Manual digital de uso.
 - Base de datos de fallas y códigos OBD-II (P0122, P030X).

4.5 Ensamblaje

El ensamblaje se realizó de manera digital:

1. Creación de la estructura base del simulador en Godot Engine (escena principal).
2. Integración de módulos funcionales:
 - Arranque → Animación de batería y sonido del motor.
 - Encendido → Generación de chispa virtual.
 - Control de aire → Variación de RPM con el TPS.
3. Conexión de los sistemas mediante código que gestiona el flujo de datos entre los módulos.
4. Pruebas de interacción y depuración para garantizar una experiencia fluida y educativa.
5. Validación del funcionamiento mediante escenarios de diagnóstico y mantenimiento preventivo.





INGENIERÍA

PROYECTOS III

CAPITULO V

INGENIERÍA AUTOMOTRIZ

ELABORACIÓN

Los Gráficos (Sprites): Son el corazón visual del simulador. La pieza principal fue la foto del motor, el "chasis". A partir de ahí, usando programas como Photoshop, Gemini AI, Upscaly (Imagen 5.1.2) entre otras más, se empezó a crear el resto de los gráficos. Es como si de una sola plancha de metal se sacaran todas las partes de la carrocería (Imagen 5.1.1).



IMAGEN 5.1.1 MOTOR SPRITES



IMAGEN 5.1.2 PROGRAMAS DE APOYO

Los Botones y Menús (UI): No todo es el motor. Se necesitaban botones ("Girar Llave", "Usar Escáner"), ventanas para el diagnóstico y textos que guiaran al jugador. Se diseñó todo esto buscando que fuera fácil de entender a la primera, sin necesidad de un manual. Muchos de estos siguen en proceso de desarrollo, pero se siguen trabajando.

El Sonido: Un motor que no suena, no convence. Se buscó en librerías *online* como **Freesound.org** para encontrar sonidos realistas: el clic de una herramienta, el sonido de un motor que no quiere arrancar, el pitido del escáner... Esto le da toda la atmósfera al simulador.

[LINK PARA FREESOUND](#)

MEDICIONES

EL TAMAÑO DEL TALLER (RESOLUCIÓN): se decidió trabajar en 1920x1080 píxeles (Full HD). Esto asegura que el simulador se vea bien en casi cualquier pantalla moderna sin que las cosas se vean estiradas o cortadas (Imagen 5.1.3).

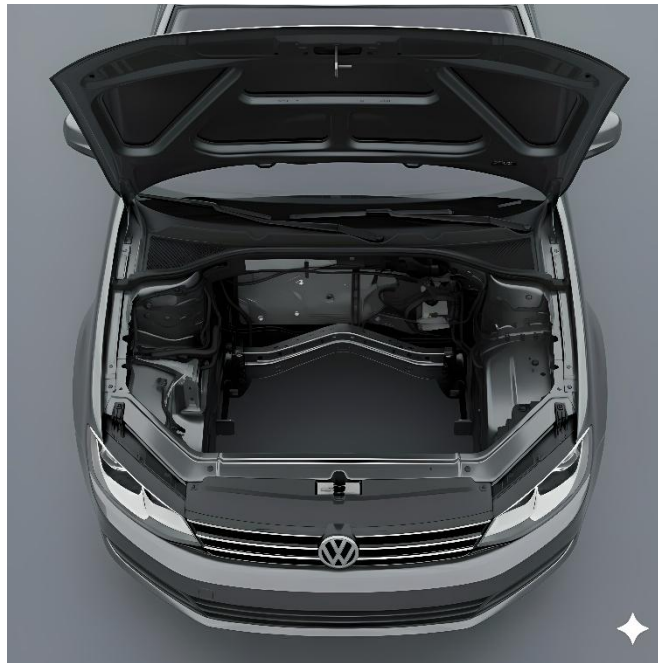


IMAGEN 5.1.3 TALLER SCRIPT

QUE LAS PIEZAS ENCAJEN: La regla fue que todas las imágenes del motor (con batería, sin batería, etc.) tuvieran exactamente el mismo tamaño. Si no, al quitar una pieza, todo el motor se movería un poco, rompiendo la lógica del proyecto..

ZONAS DE CLIC (COLLIDERS): Esta parte se encuentra en proceso, pero es básicamente **que a** cada pieza **se le aplica** una herramienta llamada Box Collider 2D, como si **se dibujara** un contorno invisible alrededor de la batería para decirle al programa: "Si el *mouse* hace clic dentro de esta área, **se está** tocando la batería".

DIAGRAMAS DE FLUJO

MAPA GENERAL DEL PROYECTO

MAPA GENERAL DEL PROYECTO

DIAGRAMA DE FLUJO

