**Tarea 2: Desarrollar el modelo de comportamiento del sistema**

**2.1 Objetivos**

**2.1.1 Objetivo General**

Modelar el comportamiento dinámico del sistema a través de la definición de operaciones, estados y transiciones de los objetos del sistema, para la profundización en el conocimiento de los requisitos funcionales y la detección de posibles conflictos en su especificación.

**2.1.2 Objetivos Específicos**

Analizar los requisitos funcionales definidos, para detectar ambigüedades, contradicciones o vacíos que puedan afectar la lógica del sistema

Especificar las operaciones del sistema necesarias para representar el comportamiento esperado frente a las acciones de los usuarios y eventos del entorno.

Identificar los tipos de objetos con comportamiento significativo y definir sus estados y transiciones, reflejando su ciclo de vida dentro del sistema.

Representar gráficamente el comportamiento del sistema utilizando diagramas de estados y traza de eventos, facilitando la comprensión del flujo de ejecución y las interacciones.

**2.2 Descripción**

En esta tarea nos enfocamos en modelar el comportamiento dinámico del sistema, es decir, cómo reacciona el sistema frente a las acciones de los usuarios y los eventos que ocurren durante su funcionamiento. A partir de los requisitos funcionales obtenidos en la elicitación y expresados en forma de descripciones tradicionales y casos de uso, se definen las operaciones que el sistema debe realizar y se identifican los tipos de objetos con comportamiento relevante, detallando sus estados y las transiciones entre ellos.

El objetivo principal es asegurar que el sistema responda correctamente a situaciones como el registro de vehículos, la validación de acceso mediante lectura de placas, la generación de alertas ante accesos no autorizados, y la administración del parqueadero en tiempo real. Además, este análisis nos permite detectar posibles conflictos, ambigüedades o vacíos en la definición de requisitos, lo que ayuda a mejorar la precisión del diseño antes de programar.

Durante esta etapa, se definieron varias operaciones clave del sistema, como:

* Registrar un vehículo con su respectiva placa.
* Validar el acceso de un vehículo en tiempo real mediante lectura de placa.
* Generar un registro de acceso (permitido o denegado).
* Emitir alertas automáticas si se detecta un intento de ingreso no autorizado.
* Consultar el estado del parqueadero (espacios disponibles).
* Administrar los usuarios, vehículos y accesos desde una interfaz.

También se identificaron objetos con comportamiento dinámico significativo, como:

* Vehículo: cambia de estado cuando se autoriza o desautoriza su acceso.
* Acceso: pasa por estados como “pendiente”, “permitido” o “denegado”.
* Parqueadero: actualiza en tiempo real su disponibilidad.
* Alerta: se activa o se cierra según se resuelva el evento.

Estos comportamientos se representarán mediante diagramas de estados y diagramas de traza de eventos, lo que facilita visualizar el flujo de ejecución del sistema y las interacciones entre los usuarios y el sistema. Este modelado no solo permite validar la lógica del sistema, sino que también sirve de base para las etapas posteriores de diseño, codificación y pruebas

**2.4 Productos entregables**

**Modelo de comportamiento como parte del DAS**

El modelo de comportamiento representa cómo actúan los objetos del sistema frente a las operaciones y eventos que ocurren durante su ejecución. Para el Sistema Inteligente de Control de Acceso Vehicular, se identificaron los siguientes elementos dinámicos clave:

**Operaciones del sistema:**

Registrar un vehículo.

Validar acceso vehicular en tiempo real.

Generar eventos de acceso (permitido o denegado).

Emitir alertas ante accesos no autorizados.

Consultar estado de disponibilidad del parqueadero.

Administrar usuarios, vehículos y accesos.

**Objetos con comportamiento significativo:**

Vehículo: puede cambiar de estado entre autorizado y no autorizado.

Acceso: transita entre “pendiente”, “permitido” o “denegado”.

Parqueadero: actualiza su disponibilidad tras cada acceso.

Alerta: se activa ante un evento irregular y se cierra al resolverse.

**Diagramas desarrollados:**

**Diagrama de traza de eventos:** muestra la secuencia de interacción entre usuarios, sistema, base de datos y parqueadero durante la validación de acceso vehicular.

**Diagrama de estados:** representa el ciclo de vida del objeto Vehículo y sus transiciones según eventos del sistema.

**Plantilla de operación ValidarAccesoVehicular:** detalla las condiciones esperadas, posibles errores y resultados de la operación de validación de ingreso/salida vehicular.

Estos elementos permiten comprender el flujo de ejecución del sistema, facilitar la detección de inconsistencias lógicas, y servir como base para el diseño y la codificación.

**Conflictos detectados durante el análisis**

Durante el modelado del comportamiento se identificaron los siguientes conflictos que podrían afectar la implementación o funcionamiento del sistema:

**Ambigüedad en el método de validación** 

No se define con claridad qué método tiene prioridad (QR o lectura de placas) ni cómo se gestiona la falla de uno de ellos.

**Falta de manejo de excepciones en horarios restringidos** 

Aunque se menciona que el acceso debe ser denegado en horarios no habilitados, no se especifica si existe alguna excepción o criterio de flexibilidad.

**Eventos no cubiertos por los requisitos actuales** 

No se detallan los flujos alternativos ante caídas de red, fallos de los lectores o errores del sistema. Esto dificulta anticipar comportamientos del sistema bajo condiciones no ideales.

**Estado del vehículo poco definido** 

La transición entre estados del objeto Vehículo no contempla escenarios como vehículos con permisos temporales o condicionados.

**Inconsistencias en la sincronización de datos** 

En caso de validación offline o errores en tiempo real, no se especifica cómo se sincronizan luego los eventos o cambios en la disponibilidad del parqueadero.

Estos conflictos serán revisados y resueltos en la etapa de diseño detallado, para garantizar un comportamiento predecible y robusto del sistema en distintos escenarios.

**2.5 Técnicas**

Durante esta fase del análisis se emplearon tres técnicas fundamentales para modelar el comportamiento del sistema: la especificación de operaciones, el diagrama de traza de eventos y el diagrama de estados. A continuación, se presenta cada técnica con su respectiva aplicación al proyecto.

***2.5.1  Plantilla para operaciones del sistema***

Se utilizó la técnica de especificación formal de operaciones propuesta en la metodología (sección 4.4 del PDF), que permite detallar el comportamiento esperado del sistema ante una operación específica. Esta técnica emplea precondiciones, postcondiciones y manejo de excepciones para definir con claridad el efecto de una operación.

Operación especificada: ValidarAccesoVehicular

acción ValidarAccesoVehicular( placa: String, tipo: String )

post:

( placa está registrada

  and vehículo está autorizado

  and tipo = "entrada" or "salida"

  and se registra el evento de acceso

  and se actualiza el contador de espacios

  and respuesta = Set{ "Acceso concedido" } )

or

( placa no registrada

  and respuesta->includes( "Error 01: Vehículo no registrado" ) )

or

( vehículo no autorizado

  and respuesta->includes( "Error 02: Acceso denegado" ) )

or

( lector de QR/placa falla

  and respuesta->includes( "Error 03: Falla en dispositivo de entrada" ) )

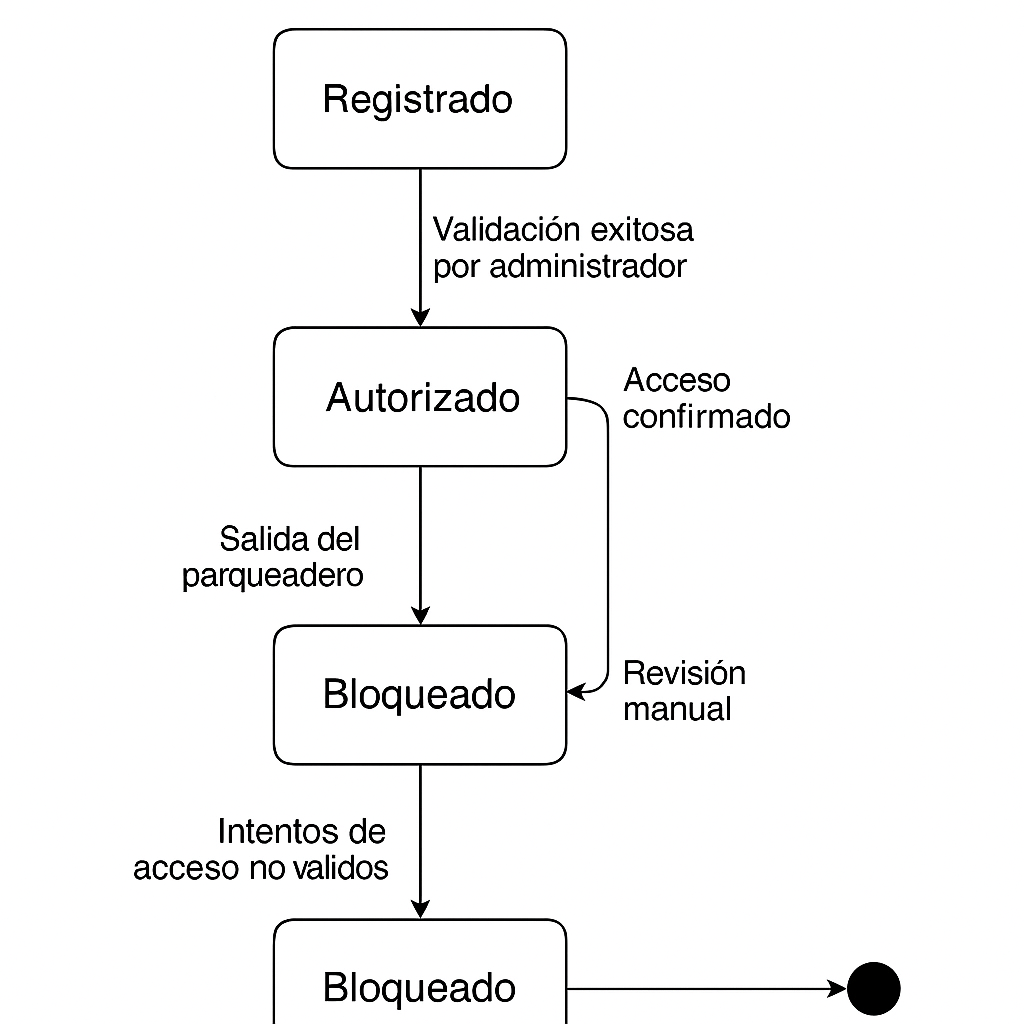
***2.5.2  Diagrama de traza de eventos***

Este diagrama (también llamado diagrama de secuencia UML) muestra cómo interactúan los distintos elementos del sistema cuando se ejecuta la operación de validar acceso. Fue generado con base en el flujo típico del caso de uso.

Diagrama: Validación de acceso vehicular

Participantes:   
Usuario, Sistema, BD (Base de Datos), Parqueadero

Flujo de eventos:



***2.5.3  Diagrama de estados***

Este diagrama modela los diferentes estados posibles de un objeto clave en el sistema, en este caso un Vehículo, y muestra cómo cambia de estado según eventos o condiciones.

Objeto modelado: Vehículo

Estados y transiciones:

