Instituto Politécnico Nacional

Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas

Sistema de gestión de inventario y personal para maquiladora

Integrantes del equipo:
Álvarez Santiago Daniela
Rojas Solís Juan Carlos
Salgado Casiano Milton Aldair
Servin Zamora Alejandro

Profesora: Obdulia Pichardo Lagunas

06 de junio de 2019

Índice

1. Descripción del problema **3**

2. Objetivo general **4**

3. Objetivos particulares **5**

4. Análisis de requerimientos **7**
4.1. Requerimientos funcionales . . . 7
4.2. Requerimientos no funcionales . . . 9
4.3. Registro de requerimientos en formatos . . . 10

5. Estado del arte **15**
5.1. Software 1: Daifuku . . . 15
5.2. Software 2: NetSuite . . . 15
5.3. Software 3: Katana MRP . . . 16
5.4. Software 4: Zoho . . . 16

6. Marco teórico **17**
6.1. Definición lenguaje de programación . . . 17
6.2. Paradigmas de programación . . . 17
6.2.1. Paradigma estructurado . . . 17
6.2.2. Paradigma orientado a objetos . . . 17
6.3. Antecedentes . . . 18

7. Solución propuesta **19**
7.1. Diagrama de casos de uso . . . 19
7.2. Diagrama de clases . . . 21
7.3. Diagrama de estado . . . 22
7.4. Diagrama de secuencia . . . 25
7.5. Código . . . 28
7.5.1. Agregar almacenista . . . 28
7.5.2. Agregar materia prima . . . 30
7.5.3. Borrar almacenista . . . 30
7.5.4. Borrar materia prima . . . 31

8. Conclusiones **32**

Referencias **33**

1. Descripción del problema

Una empresa que se dedica a la manufactura de ropa solicitó que se le realizará una actualización al sistema que vienen manejando para la gestión del inventario y el personal. La empresa argumenta que debido al aumento de clientes (pasó de unas decenas de clientes, a cientos de clientes alrededor del mundo) el sistema se ha hecho cada vez más lento y menos eficiente. También comentan que, si el sistema que vienen manejando es demasiado obsoleto como para actualizarse, cabe la posibilidad de que se haga uno nuevo.

2. Objetivo general

Diseñar e implementar un sistema que coadyuve en la correcta gestión de un almacén de materia prima, para la elaboración de ropa. Este mismo podrá agregar, eliminar, buscar y editar algún registro del almacén.

También podra mostrar todos los registros y podrá calcular el costo del material comprado. Asímismo podrá agregar, eliminar, buscar y editar algún registro de los almacenistas, mostrar todos los registros e imprimir un reporte de éstos últimos.

3. Objetivos particulares

- Se colocará un mensaje de bienvenida al momento de iniciar el sistema, dicho mensaje contendrá lo siguiente:
  - El nombre del proyecto ("Sistema de gestión de inventario y personal para maquiladora").
  - El día, mes, año y hora.
- Se implementará un login, pidiendo un usuario y una contraseña:
  - El usuario y la contraseña solo se podrán modificar directamente en el código fuente, el ejecutable no podrá cambiar dichos parámetros.
  - En caso de que el operador del sistema se equivoque escribiendo el usuario y/o la contraseña 3 veces, el sistema desplegará un mensaje diciendo: "Error, 3 intentos de usuario y/o contraseña incorrectos" y automáticamente cerrara el programa.
- Se incluirá un módulo que será el menú general. Dicho módulo contendrá:
  1. Menú de almacenista
  2. Menú de materia prima
  3. Salir
- El menú de almacenista contendrá 7 opciones:
  1. Agregar registro almacenista:
     - El ID lo generará el programa de forma automática y secuencial, por lo que no se podrá repetir.
     - Luego pedirá el nombre del almacenista.
  2. Borrar registro almacenista:
     - Para borrar un registro de almacenista pedirá el número de ID.
  3. Buscar registro almacenista:
     - Para buscar un registro de almacenista pedirá el número de ID. Si el registro existe desplegará el ID y el nombre del almacenista.
  4. Editar registro almacenista:
     - Para editar un registro de almacenista pedirá el número de ID. Si el registro existe se podrá editar el nombre del almacenista.
  5. Mostrar registro almacenista:
     - El sistema desplegará el ID y el nombre del almacenista de todos los registros.
  6. Imprimir reporte:
     - Para mostrar el reporte pedirá el número de ID. Si el registro existe desplegará el ID y el nombre del almacenista (**NOTA: LO VA A IMPRIMIR EN PANTALLA, NO EN FÍSICO, además de que esto solo muestra un reporte por almacenista**)
  7. Salir del programa:
- En caso de que el operador escoja cualquier otro número que no sea del 1 al 7, el sistema desplegará el siguiente mensaje: "Opción no válida, elija nuevamente".

- El menú de materia prima contendrá 7 opciones:

  1. Agregar materia prima:
     - El ID lo generará el programa de forma automática y secuencial, por lo que no se podrá repetir.
     - Luego pedirá el nombre de la materia prima (por ejemplo, algodón, nilón, cuero, licra, poliéster, lana, etc.)
     - Luego pedirá el nombre del proveedor de la materia prima.
     - Después pedirá el precio unitario.
     - Después las unidades compradas.
  2. Borrar registro de materia prima:
     - Para borrar un registro del inventario pedirá el número de ID.
  3. Buscar registro:
     - Para buscar un registro del inventario pedirá el número de ID. Si el registro existe desplegará el ID, el nombre de la materia prima, el nombre del proveedor, el precio unitario y las unidades compradas.
  4. Editar registros:
     - Para editar un registro del inventario pedirá el número de ID. Si el registro existe se podrá editar el nombre de la materia prima, el nombre del proveedor, el precio unitario y las unidades compradas.
  5. Mostrar registros:
     - El sistema desplegará el ID, el nombre de la materia prima, el nombre del proveedor, el precio unitario y las unidades compradas de todos los registros de materia prima.
  6. Calcular costo:
     - Para calcular el costo de un registro del inventario pedirá el número de ID. Si el registro existe desplegará el costo calculado (NOTA: esto solo calculará el costo de un registro a la vez). La operación que realiza para calcular el costo es multiplicar el precio unitario por las unidades compradas.
  7. Salir del programa:

- En caso de que el operador escoja cualquier otro número que no sea del 1 al 7, el sistema desplegará el siguiente mensaje: “Opción no válida, elija nuevamente”.

4. Análisis de requerimientos

Los análisis de requerimientos son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Reflejan las necesidades del cliente de un sistema que resolverá un problema. En este capítulo se mostrarán las necesidades de nuestro cliente que se dividen en funcionales y no funcionales.

Funcionales: son funciones externas del programa como la información, interfaz, navegación y personalización del mismo.

No funcionales: son los requerimientos que tienen que ver con la usabilidad, eficiencia, fiabilidad, implementación y portabilidad del sistema.

4.1. Requerimientos funcionales

- El programa desplegará un mensaje de bienvenida que contendrá el nombre del sistema y la fecha (día, mes, año y hora).
- Solo los usuarios que ingresen el nombre de usuario y la contraseña correctamente podrán ingresar al sistema.
- El nombre de usuario y la contraseña solo podrán ser modificados directamente en el código fuente.
- El nombre de usuario es: “admin” y la contraseña es: “MyNewP4ss!”.
- En caso de que el nombre de usuario o la contraseña sean introducidos de manera incorrecta un máximo de 3 veces, el sistema desplegará un mensaje de alerta y se cerrará el programa.
- El sistema contendrá 3 menús: el menú general, el menú para todas las operaciones relacionadas con almacenista y el menú para todas las operaciones relacionadas con materia prima.
- Para el menú almacenista:
  - Contendrá 7 opciones:
    1. Agregar registro almacenista
       - El ID lo generará el programa de forma automática y secuencial, por lo que no se podrá repetir.
       - El ID sera una variable de tipo entero (int).
       - El campo nombre del almacenista podrá aceptar caracteres alfanuméricos.
    2. Eliminar registro almacenista
       - Esta opción pedirá el número de ID para eliminar el registro de almacenista.
    3. Buscar registro almacenista
       - Esta opción pedirá el número de ID para buscar el registro de almacenista.
    4. Editar registro almacenista
       - Esta opción pedirá el número de ID para editar el registro de almacenista.
    5. Mostrar registros almacenistas
       - Esta opción mostrará todos los registros guardados de almacenistas.
    6. Imprimir reporte (**NOTA: LO VA A IMPRIMIR EN PANTALLA, NO EN FÍSICO**)
       - Esta opción pedirá el número de ID para imprimir el reporte de un solo almacenista.
    7. Salir

- Para el menú materia prima:
  - Contendrá 7 opciones:
    1. Agregar registro materia prima
       - El ID lo generará el programa de forma automática y secuencial, por lo que no se podrá repetir.
       - El ID será una variable de tipo entero (int).
       - El campo nombre de la materia prima podrá aceptar caracteres alfanuméricos.
       - El campo nombre del proveedor podrá aceptar caracteres alfanuméricos.
       - El campo precio unitario solo podrá aceptar números con o sin decimales.
       - El campo cantidad total comprada solo podrá aceptar números con o sin decimales.
    2. Eliminar registro materia prima
       - Esta opción pedirá el número de ID para eliminar el registro de materia prima.
    3. Buscar registro materia prima
       - Esta opción pedirá el número de ID para buscar el registro de materia prima.
    4. Editar registro materia prima
       - Esta opción pedirá el número de ID para editar el registro de materia prima.
    5. Mostrar registros materia prima
       - Esta opción mostrará todos los registros guardados de materia prima.
    6. Costo total por producto
       - Esta opción pedirá el número de ID para calcular el costo total por producto. La operación que realiza es la siguiente: multiplica el precio unitario por la cantidad total comprada y el resultado es el costo total por producto.
    7. Salir
- El sistema solo podrá ser utilizado en sistemas operativos Windows.

4.2. Requerimientos no funcionales

- La rapidez de las operaciones del sistema dependerá enteramente del hardware en el que se ejecute.
- El sistema será desarrollado únicamente para sistemas operativos Windows.
- El sistema solo manejará el idioma español.
- El proceso de desarrollo se gestionará por medio de la herramienta proporcionada por el sitio web Github.
- El sistema no revelará el nombre de usuario y/o contraseña para ingresar al mismo.
- El sistema no estará bajo ninguna licencia.
- El sistema será escrito en el lenguaje de programación C++.

4.3. Registro de requerimientos en formatos

Registro de requerimientos				
ID del requerimiento:	FD01			
Nombre del requerimiento:	Bienvenida			
Identificación del requerimiento:	Mensaje de bienvenida			
Características:	El programa desplegará un mensaje de bienvenida que contendrá el nombre del sistema y la fecha (día, mes, año y hora)			
Descripción del requerimiento:	Este requerimiento será para desplegar el mensaje de bienvenida al momento de iniciar el sistema			
Requerimiento NO funcional:	NFD01 ; NFD02 ; NFD03 ; NFD04 ; NFD05 ; NFD06 ; NFD07			
Prioridad del requerimiento				
Alta	**Media Alta**	**Media**	**Media Baja**	**Baja**
			X	

Cuadro 1: Requerimiento 1

Registro de requerimientos				
ID del requerimiento:	FD02			
Nombre del requerimiento:	Ingreso al sistema			
Identificación del requerimiento:	Usuarios autorizados			
Características:	El usuario y contraseña deberán coincidir con el proporcionado por el desarrollador			
Descripción del requerimiento:	Solo los usuarios que ingresen el nombre de usuario y la contraseña correctamente podrán ingresar al sistema			
Requerimiento NO funcional:	NFD01 ; NFD02 ; NFD03 ; NFD04 ; NFD05 ; NFD06 ; NFD07			
Prioridad del requerimiento				
Alta	**Media Alta**	**Media**	**Media Baja**	**Baja**
X				

Cuadro 2: Requerimiento 2

Registro de requerimientos				
ID del requerimiento:	FD03			
Nombre del requerimiento:	Configuración usuario y contraseña			
Identificación del requerimiento:	Configuración seguridad login			
Características:	Solo se podrá cambiar el usuario y contraseña a través del código fuente			
Descripción del requerimiento:	El nombre de usuario y la contraseña solo podrán ser modificados directamente en el código fuente			
Requerimiento NO funcional:	NFD01 ; NFD02 ; NFD03 ; NFD04 ; NFD05 ; NFD06 ; NFD07			
Prioridad del requerimiento				
Alta	**Media Alta**	**Media**	**Media Baja**	**Baja**
X				

Cuadro 3: Requerimiento 3

Registro de requerimientos				
ID del requerimiento:	FD04			
Nombre del requerimiento:	Usuario y contraseña			
Identificación del requerimiento:	Credenciales			
Características:	Nombre de usuario y contraseña proporcionados por el desarrollador			
Descripción del requerimiento:	El nombre de usuario es: “admin” y la contraseña es: “MyNewP4ss!”			
Requerimiento NO funcional:	NFD01 ; NFD02 ; NFD03 ; NFD04 ; NFD05 ; NFD06 ; NFD07			
Prioridad del requerimiento				
Alta	**Media Alta**	**Media**	**Media Baja**	**Baja**
X				

Cuadro 4: Requerimiento 4

Registro de requerimientos				
ID del requerimiento:	FD05			
Nombre del requerimiento:	3 intentos			
Identificación del requerimiento:	Error al tercer intento login			
Características:	Se tendrá un máximo de 3 intentos para ingresar al sistema			
Descripción del requerimiento:	En caso de que el nombre de usuario o la contraseña sean introducidos de manera incorrecta un máximo de 3 veces, el sistema desplegará un mensaje de alerta y se cerrará el programa			
Requerimiento NO funcional:	NFD01 ; NFD02 ; NFD03 ; NFD04 ; NFD05 ; NFD06 ; NFD07			
Prioridad del requerimiento				
Alta	**Media Alta**	**Media**	**Media Baja**	**Baja**
X				

Cuadro 5: Requerimiento 5

Registro de requerimientos				
ID del requerimiento:	FD06			
Nombre del requerimiento:	Menús			
Identificación del requerimiento:	3 menús			
Características:	Se tendrán 3 menús			
Descripción del requerimiento:	El sistema contendrá 3 menus: el menú general, el menú para todas las operaciones relacionadas con almacenista y el menú para todas las operaciones relacionadas con materia prima			
Requerimiento NO funcional:	NFD01 ; NFD02 ; NFD03 ; NFD04 ; NFD05 ; NFD06 ; NFD07			
Prioridad del requerimiento				
Alta	**Media Alta**	**Media**	**Media Baja**	**Baja**
		X		

Cuadro 6: Requerimiento 6

Registro de requerimientos				
ID del requerimiento:	FD07			
Nombre del requerimiento:	Plataforma SO			
Identificación del requerimiento:	Plataforma			
Características:	Solo se podrá ejecutar en sistemas operativos Windows			
Descripción del requerimiento:	El sistema solo podrá ser utilizado en sistemas operativos Windows			
Requerimiento NO funcional:	NFD01 ; NFD02 ; NFD03 ; NFD04 ; NFD05 ; NFD06 ; NFD07			
Prioridad del requerimiento				
Alta	**Media Alta**	**Media**	**Media Baja**	**Baja**
				X

Cuadro 7: Requerimiento 7

- **NFD01 -** La rapidez de las operaciones del sistema dependerá enteramente del hardware en el que se ejecute.
- **NFD02 -** El sistema será desarrollado únicamente para sistemas operativos Windows.
- **NFD03 -** El sistema solo manejará el idioma español.
- **NFD04 -** El proceso de desarrollo se gestionará por medio de la herramienta proporcionada por el sitio web Github.
- **NFD05 -** El sistema no revelará el nombre de usuario y/o contraseña para ingresar al mismo.
- **NFD06 -** El sistema no estará bajo ninguna licencia.
- **NFD07 -** El sistema será escrito en el lenguaje de programación C++.

5. Estado del arte

El estado del arte nos permite el estudio de conocimiento acumulado sobre un tema específico y de esta manera determinar los avances más importantes en dicho tema, este capítulo tiene como objetivo mostrar la implementación de softwares parecidos al que desarrollamos y sus resultados.

5.1. Software 1: Daifuku

Desde 1966 Daifuku desarrolló el primer sistema de almacenamiento. Los objetivos para desarrollar este sistema incluían alcanzar reducciones en la carga de trabajo y ahorros en costos mediante:

1. El uso efectivo de la tierra
2. Las mejoras en la eficacia del almacenaje
3. Ahorro en costos de personal y de mano de obra en el trabajo en almacén y
4. Mejoras en los niveles de gestión [1]

5.2. Software 2: NetSuite

NetSuite Inc. es una empresa estadounidense de computación en la nube fundada en 1998 con sede en San Mateo, California, que proporciona software y servicios para administrar las finanzas comerciales, las operaciones y las relaciones con los clientes. Su software y servicios están diseñados para pequeñas y medianas empresas.

NetSuite ofrece cuatro soluciones principales de software y servicios opcionales de implementación y soporte:

- **Planificación de recursos empresariales (ERP):** NetSuite ERP admite operaciones de back office que incluyen recursos financieros, recursos humanos, compras, pedidos, inventario, envío y facturación.
- **NetSuite OneWorld:** ofrece los servicios anteriores más capacidades adicionales para compañías multinacionales, como la capacidad de administrar múltiples subsidiarias, monedas, estándares de contabilidad y requisitos fiscales.
- **Gestión de las relaciones con los clientes (CRM):** NetSuite CRM es compatible con las operaciones de marketing, ventas y servicios.
- **Comercio electrónico (e-commerce):** SuiteCommerce de NetSuite está pensado como una plataforma para ventas en línea e integración con las herramientas tradicionales de teléfono y punto de venta (POS). La plataforma de SuiteCommerce está actualmente en uso por más de 1600 sitios web en línea.
- **Automatización de servicios profesionales (PSA):** NetSuite PSA se basa en la adquisición de OpenAir y administra las operaciones de negocios basados en servicios y orientados a proyectos.
- **Servicios Netsuite:** SuitePackages ofrece servicios básicos de implementación para clientes. Packaged Services ofrece personalización e integración de las soluciones Netsuite; el soporte de suite ofrece ayuda y soporte en línea en caso de problemas.[2]

5.3. Software 3: Katana MRP

Katana MRP es un software de producción e inventario para fabricantes. Katana está repleta de características que ayudan a dirigir un taller inteligente.

Características:

- **Planeación de producción**
  - Prioridades de arrastrar y soltar para trabajos de fabricación.
  - Rastrear la disponibilidad de materiales requeridos.
  - Identificar los riesgos de demora relacionados con los tiempos de entrega del material.
  - Fechas de finalización esperadas exactas.
  - Obtenga información general del estado de producción en tiempo real desde el nivel del piso.
- **Control de inventario en tiempo real y optimización**
  - Control de inventario de productos terminados y materias primas.
  - Transacciones de inventario automatizadas.
  - Control en mano, cantidad de stocks comprometidos y esperados en tiempo real.
  - Mantener niveles de inventario óptimos usando puntos de pedido.
  - Tomar decisiones precisas de fabricación y compra.
  - Gestiona fácilmente las variantes de productos y materiales.
- **Cumplimiento de órdenes de venta**
  - Rastrear la disponibilidad de productos requeridos.
  - Prioridades de arrastrar y soltar de pedidos de clientes.
  - Hacer a pedido o cumplir con el stock de producto disponible.
  - Identifique los riesgos de demora en la entrega y vuelva a priorizar para cumplir con los plazos.
  - Sincronizar pedidos de ventas de múltiples canales en un único panel de control.
- **Integrado con servicios en línea**
  - Sincronizar ventas con plataformas de comercio electrónico.
  - Empuje las órdenes de venta a la contabilidad para la facturación.
  - Mantener las compras sincronizadas con la contabilidad.[3]

5.4. Software 4: Zoho

Software completo y adecuado para negocios e industrias de todos los tamaños. Equipado con AI.
Características:

- **Multicanal:** conéctese en tiempo real con los clientes y clientes potenciales en todos los canales. Desde correo electrónico hasta teléfono, pasando por chat en vivo e incluso en medios sociales. SalesSignals notifica cuando un cliente potencial prioritario navega en su sitio, lee su campaña de correo electrónico o interactúa con su marca en medios sociales.
- **Zia (inteligencia artificial):** gracias a que Zia aprende sobre su empresa y su equipo, puede realizar predicciones sobre tendencias, anomalías, conversiones y tratos que se cerrarán. Zia también automatiza de forma automática las tareas más recurrentes para agilizar su ciclo de ventas.
- **Automatización de ventas:** las capacidades de automatización avanzadas de Zoho CRM ayudan a que su equipo de ventas ahorre tiempo, ya que completa las tareas de rutina automáticamente, de este modo se pueden enfocar en cerrar una mayor cantidad de tratos en menos tiempo.
- **Móvil:** manténgase informado sobre el negocio en cualquier lugar y a toda hora gracias a la edición para dispositivos móviles de Zoho CRM. Infórmese mejor, manténgase conectado con sus clientes y colegas y reaccione a los rápidos cambios que ocurren, dondequiera que se encuentre.[4]

6. Marco teórico

6.1. Definición lenguaje de programación

Vamos a comenzar definiendo que es un lenguaje de programación. En informática, se conoce como lenguaje de programación a un programa destinado a la construcción de otros. Su nombre se debe a que comprende un lenguaje formal que está diseñado para organizar algoritmos y procesos lógicos que serán luego llevados a cabo por un ordenador o sistema informático, permitiendo controlar así su comportamiento físico, lógico y su comunicación con el usuario humano.[5]

6.2. Paradigmas de programación

Ahora, comúnmente los lenguajes de programación se clasifican por paradigmas de programación. Podemos decir que los paradigmas son propuestas que podemos adoptar o seguir para solucionar un problema. Existen distintos tipos de paradigmas, pero solamente nos enfocaremos en dos: paradigma estructurado y paradigma orientado a objetos (retomaremos estos conceptos más adelante).

6.2.1. Paradigma estructurado

El paradigma de la programación estructurada es probablemente el más usado para iniciar en la programación. Tiene reglas muy sencillas:

- El inicio del archivo es el inicio de la programación y el final del archivo es el final de la programación; en otras palabras es secuencial.
- Se pueden utilizar instrucciones de control como las condicionales (if)
- Se pueden utilizar instrucciones de iteración (Bucles)

6.2.2. Paradigma orientado a objetos

El paradigma de la programación orientada a objetos es de los más usados para desarrollo de software en la industria, esto se debe a que los objetos y las clases son la base de muchos proyectos.

Algunas de las cosas que debemos saber de este paradigmas son:

- Los objetos se crean y se destruyen
- Las clases son lo que utilizamos para definir los objetos
- Las clases pueden contener atributos y métodos.[6]

6.3. Antecedentes

Ahora bien, hablaremos un poco acerca de la historia del lenguaje C. Es uno de los lenguajes de programación más populares en el mundo ya que tiene funciones avanzadas y notorias. Creado en el año 1969 en la empresa de AT&T, aunque solo era una idea para poder resolver algunos inconvenientes con los lenguajes de la época y fue hasta 1972 donde se sentarón las bases del lenguaje C.[7]

Posteriormente ANSI desarrollo C++ que fue hecho con base en el lenguaje C, y la principal razón por la cual se desarrollo fue para extender el ya existente lenguaje C, con funciones que permitirían la manipulación de objetos, por lo que se puede decir que este lenguaje es un híbrido, ya que se puede tomar como paradigma de programación estructurada o también puede ser orientada a objetos.[8]

Para el paradigma de programación estructurada podemos entender que esta enfocado a mejorar la claridad, calidad y tiempo de desarrollo de un proyecto. De esta manera para poder crear un programa parte de un teorema desarrollado por Edsger Dikjstra, que en el demuestra que todo sistema puede escribirse utilizando únicamente tres estructuras de control:

1. **Secuencia:** el bloque secuencial de instrucciones, ejecutadas sucesivamente, una detrás de otra.
2. **Selección:** la instrucción condicional con doble alternativa, de la forma "**if** condición **then** instrucción 1, **else** instrucción 2".
3. **Iteración:** el bucle condicional "**while** condición **do** instrucción", este ejecuta repetidamente la acción mientras la condición se cumpla.

Aquí también entra un nuevo enfoque:
Segmentación: es la división de un problema global en varios mas pequeños y así poder resolver mas fácilmente el problema global (divide y vencerás).[9]

Mientras que en la programación estructurada dividía los problemas para obtener unos más pequeños o que sean mas generales, en el paradigma orientado a objetos tomaba otro enfoque en el cual consistía en usar objetos y sus interacciones que pueden llegar a tener con otros objetos.

Esta basado en varias técnicas como:

- **Herencia:** es cuando una clase que contiene objetos crea una subclase, en el cual pueden compartir ya sean varias características de la clase madre o tenga todas las características de la principal y aparte ésta subclase tenga mas métodos o atributos autónomos de la clase madre.
- **Abstracción:** son las características que puede tener un objeto y que ayuda a diferenciarse de otros.
- **Polimorfismo:** es la capacidad que tienen los objetos de una clase de responder al mismo evento en función de los parámetros utilizados durante su invocación. Podemos entender por esto que es el tipo de datos que manejara el objeto para poder trabajar y que al final de su ejecución devolverá un valor del mismo tipo que el objeto.
- **Encapsulamiento:** es un mecanismo el cual consiste en no permitir el acceso a datos por cualquier medio distinto a los especificados. Por lo tanto, el encapsulamiento garantiza la integridad de los datos que contiene un objeto.[10]

Entonces podemos decir que el paradigma de programación orientada a objetos se diferencía de la estructurada con el manejo de objetos, en el cual depende de creación de clases que son las encargadas de generar los objetos y a su vez estos utilizan ciertos atributos y métodos de la clase para poder realizar una ejecución de un bloque de código que puede o no regresar valores del mismo tipo como resultado de su ejecución.

Para poder ejecutar o ver los resultados del lenguaje C++, este utiliza un programa especial que procesa las instrucciones escritas en un lenguaje de alto nivel y las convierte a lenguaje máquina(binario). El programa permite la comprobación de errores y a su vez crea un archivo ejecutable, que es el que se encarga de mostrar el resultado de lo que se hizo en el lenguaje de programación.[11]

En la actualidad se han desarrollado diversos tipos de programas con este lenguaje, que al trabajar con el te ayuda a entenderlo más fácilmente y así poder crear una solución a un problema. También con ayuda de ciertos diagramas que se elaboran previamente al código fuente podemos tener una perspectiva de como funcionará el programa.

7. Solución propuesta

En esta sección, se va a mostrar la solución propuesta para el objetivo planteado. Se mostrarán los diagramas que se realizaron para la creación del "Sistema de gestión de inventario y personal para maquiladora".

7.1. Diagrama de casos de uso

Como se puede apreciar en las figuras 1 y 2, se propuso que un solo actor iba a manejar el sistema (en este caso, sería el administrador del almacen).

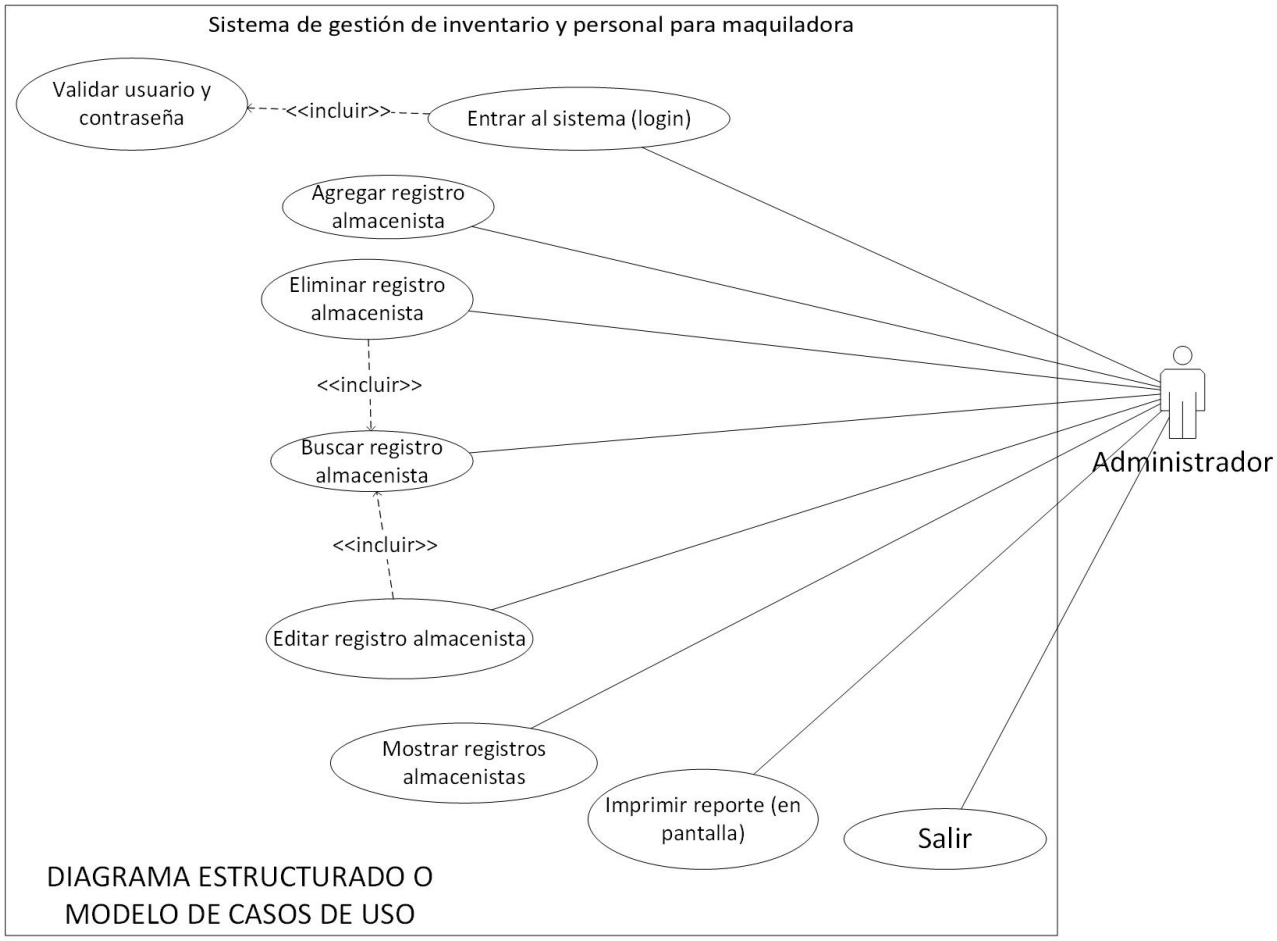

Figura 1: Diagrama de casos de uso para el "Sistema de gestión de inventario y personal para maquiladora".

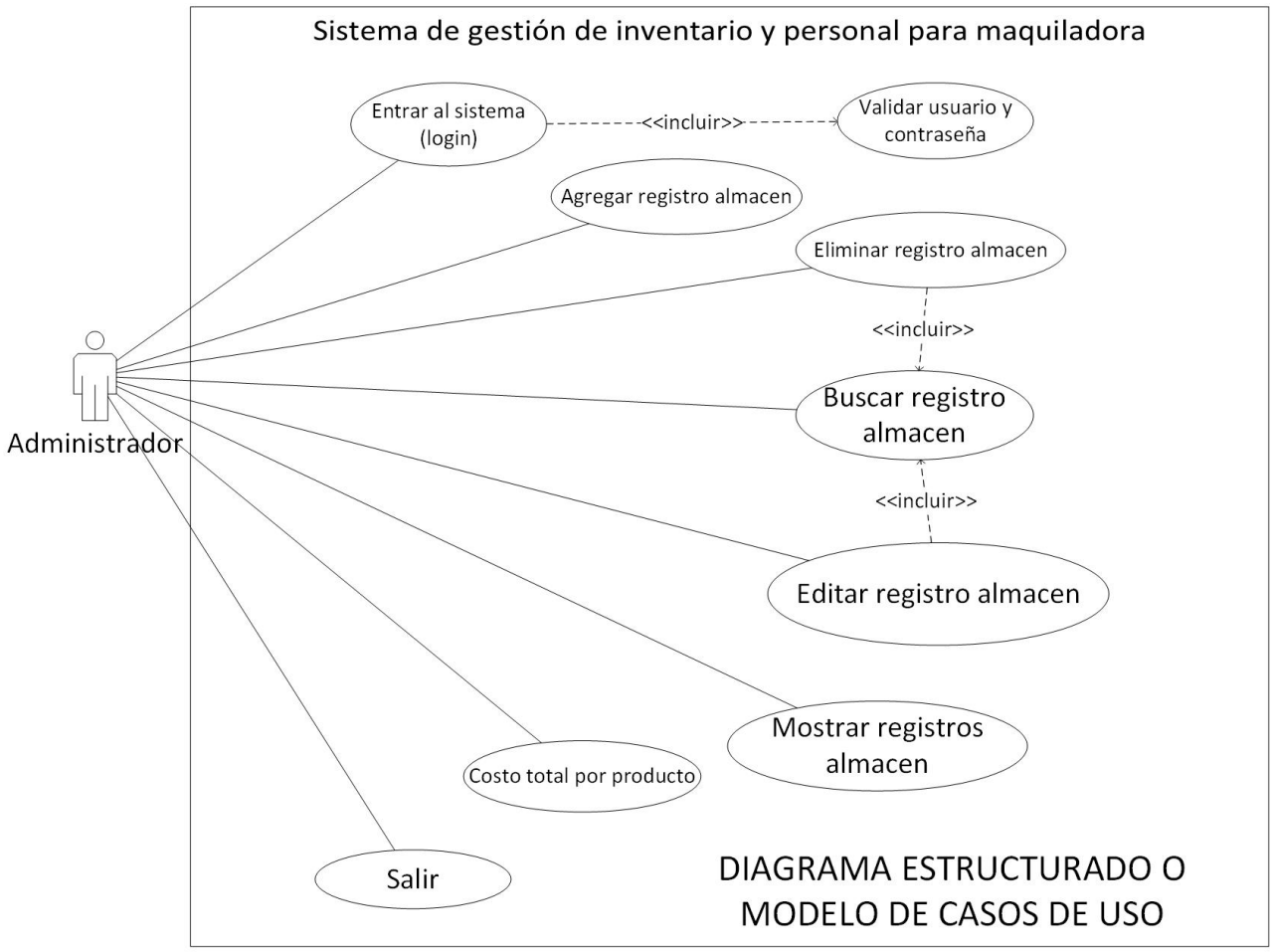

Figura 2: Diagrama de casos de uso para el "Sistema de gestión de inventario y personal para maquiladora".

7.2. Diagrama de clases

En la figura 3, se propusieron 5 clases. La clase C_sistema es la superclase, mientras que las clases C_materia_prima y C_almacenista heredan sus métodos (cabe mencionar que son métodos virtuales). La clase C_ login esta en una relación de composición con la clase C_menú y con multiplicidad 1 a 1.

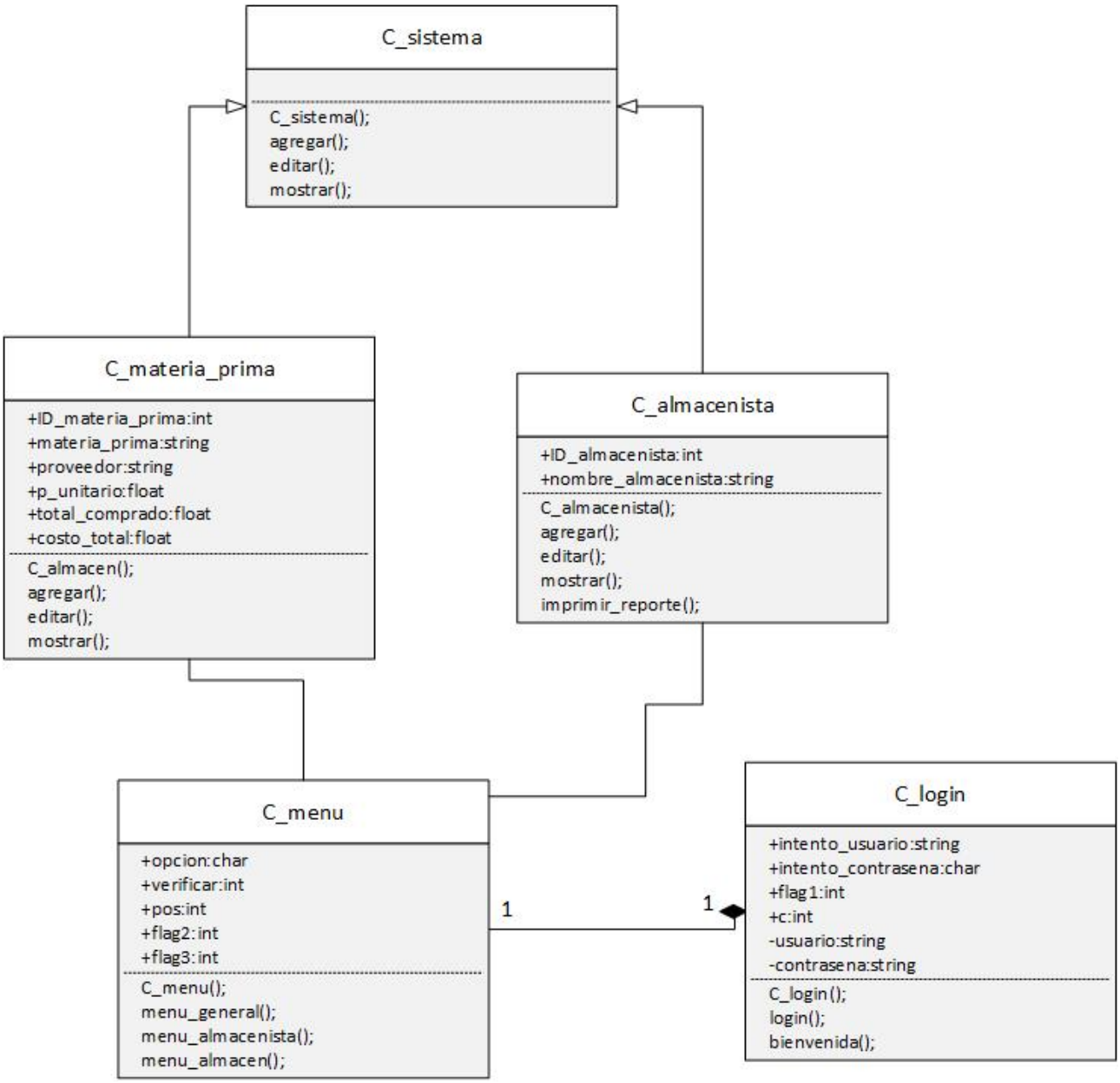

Figura 3: Diagrama de clases para el "Sistema de gestión de inventario y personal para maquiladora".

7.3. Diagrama de estado

Debido a que el diagrama de estados iba a ser demasiado grande y complicado de interpretar, se optó por dividir los procesos. Como se puede apreciar en la figura 4, se realizó el diagrama de estados de la opción agregar.

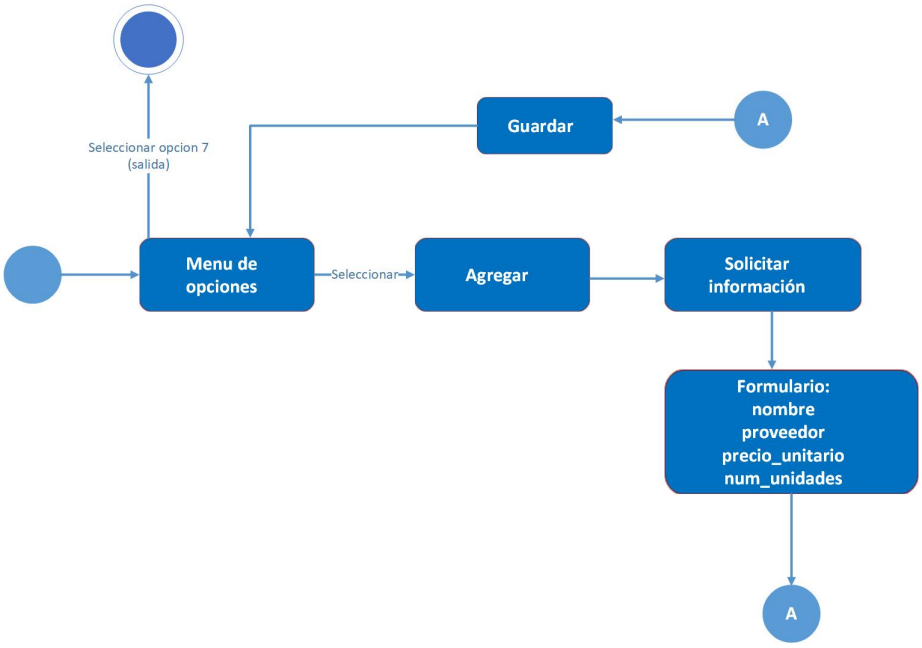

Figura 4: Diagrama de estados de la opción agregar para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 5, se realizó el diagrama de estados de la opción borrar.

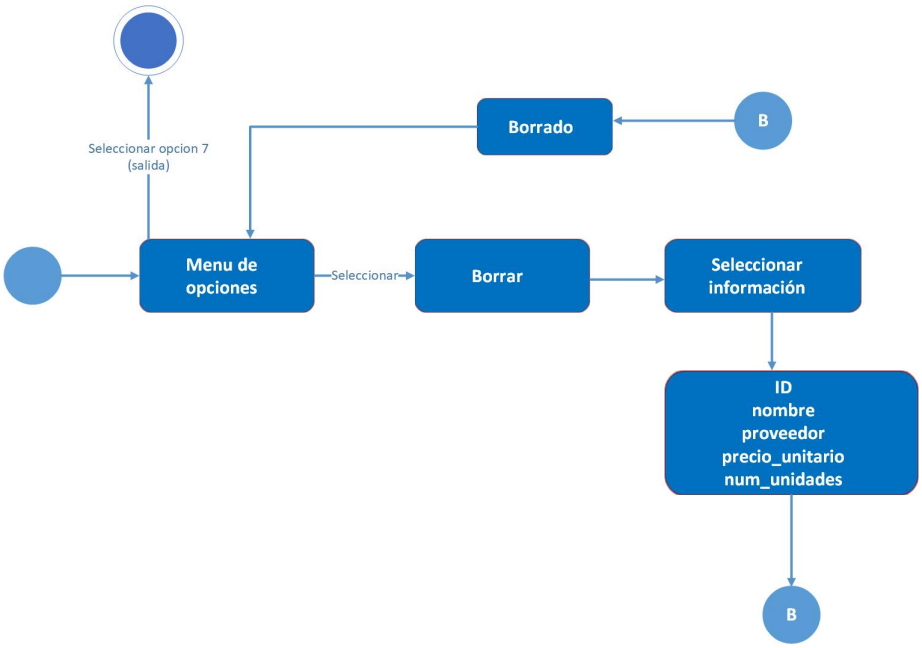

Figura 5: Diagrama de estados de la opción borrar para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 6, se realizó el diagrama de estados de la opción editar.

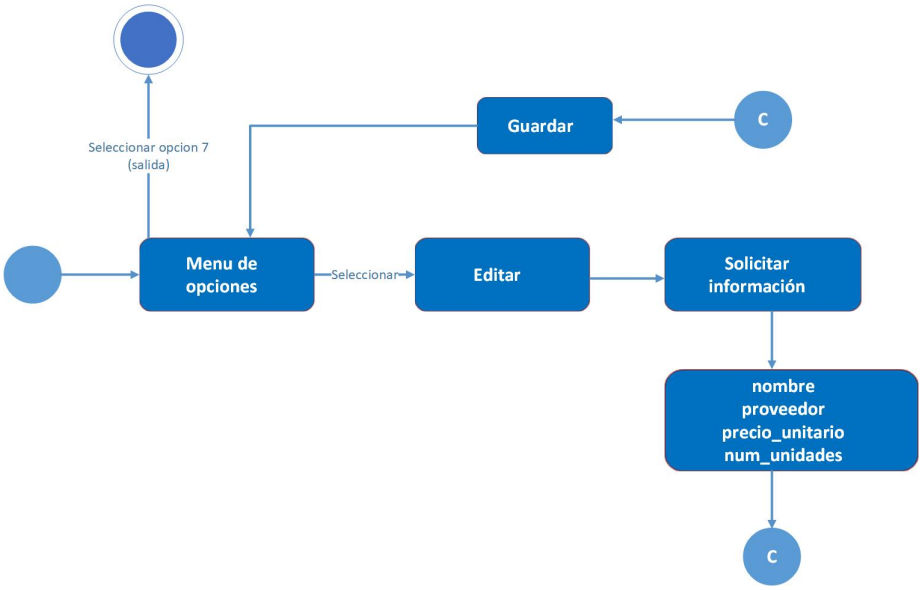

Figura 6: Diagrama de estados de la opción editar para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 7, se realizó el diagrama de estados de la opción mostrar.

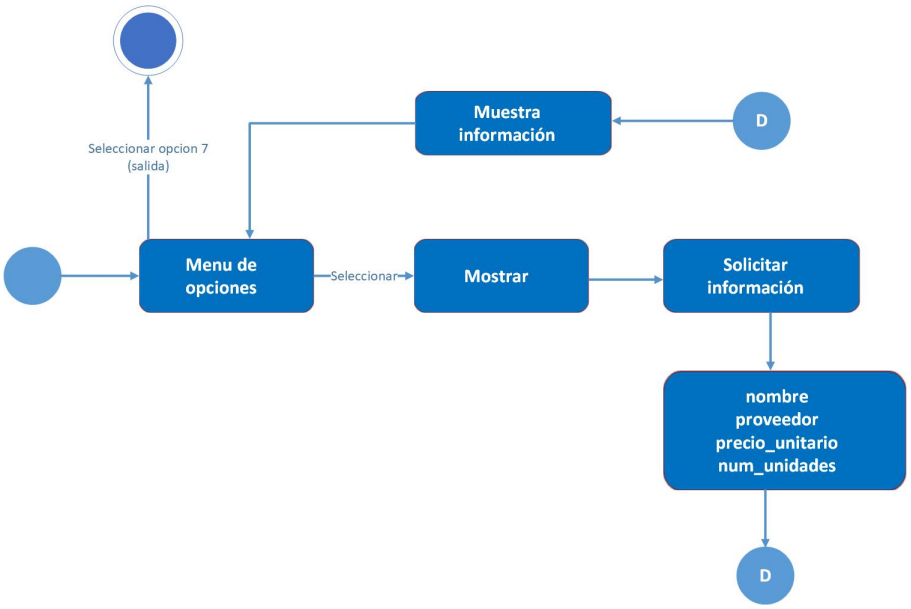

Figura 7: Diagrama de estados de la opción mostrar para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 8, se realizó el diagrama de estados de la opción costo total.

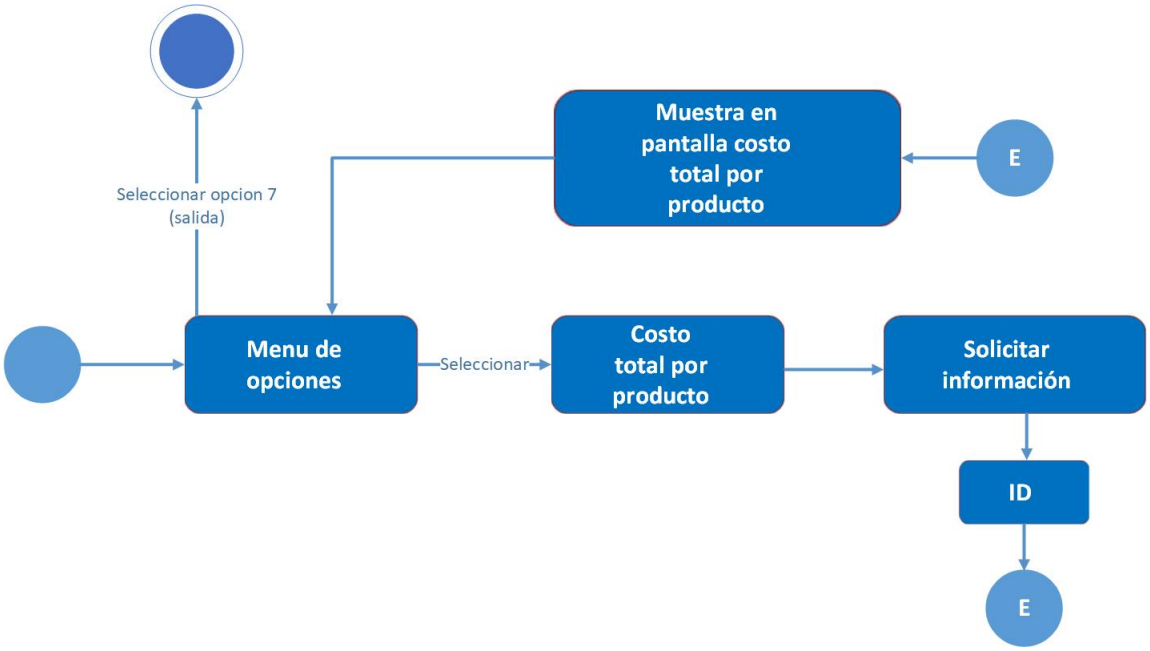

Figura 8: Diagrama de estados de la opción costo total para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 9, se realizó el diagrama de estados de la opción imprimir reporte.

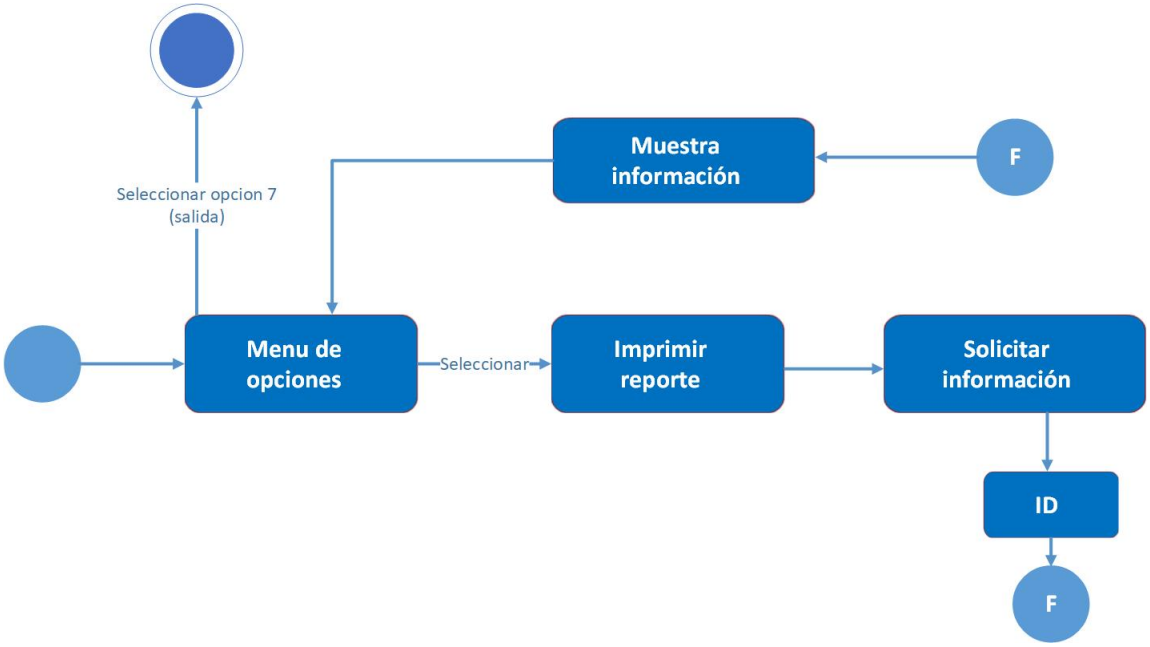

Figura 9: Diagrama de estados de la opción imprimir reporte para el "Sistema de gestión de inventario y personal para maquiladora".

7.4. Diagrama de secuencia

Se presentarán los diagramas de secuencia que se elaboraron. Como se puede apreciar en la figura 10, se realizó el diagrama de secuencia de la opción agregar materia prima.

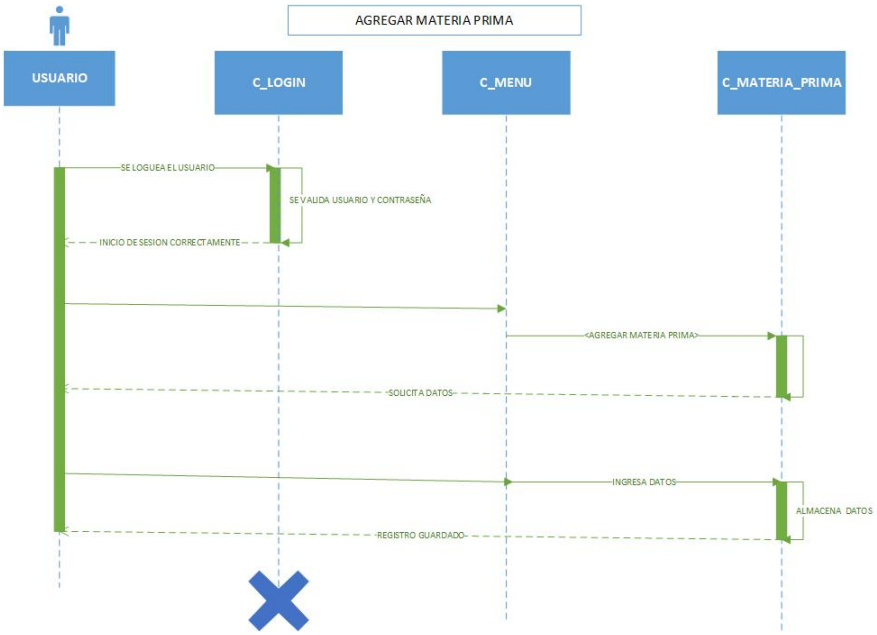

Figura 10: Diagrama de secuencia de la opción agregar materia prima para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 11, se realizó el diagrama de secuencia de la opción agregar almacenista.

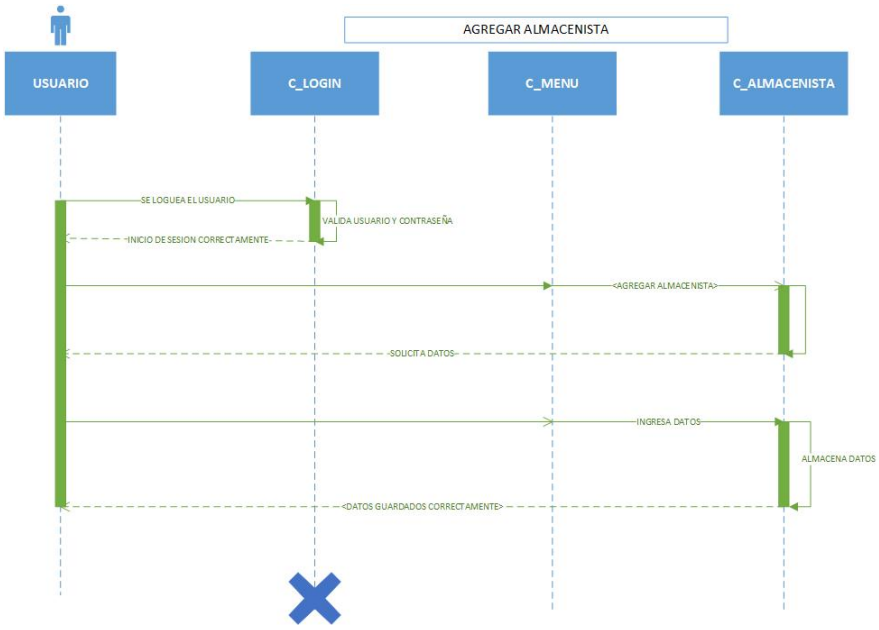

Figura 11: Diagrama de secuencia de la opción agregar almacenista para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 12, se realizó el diagrama de secuencia de la opción editar materia prima.

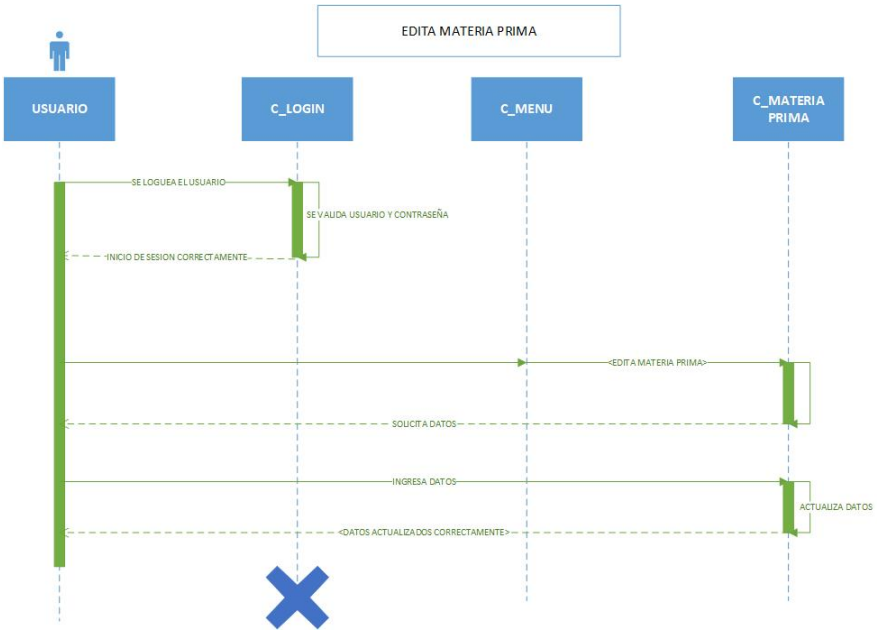

Figura 12: Diagrama de secuencia de la opción editar materia prima para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 13, se realizó el diagrama de secuencia de la opción editar almacenista.

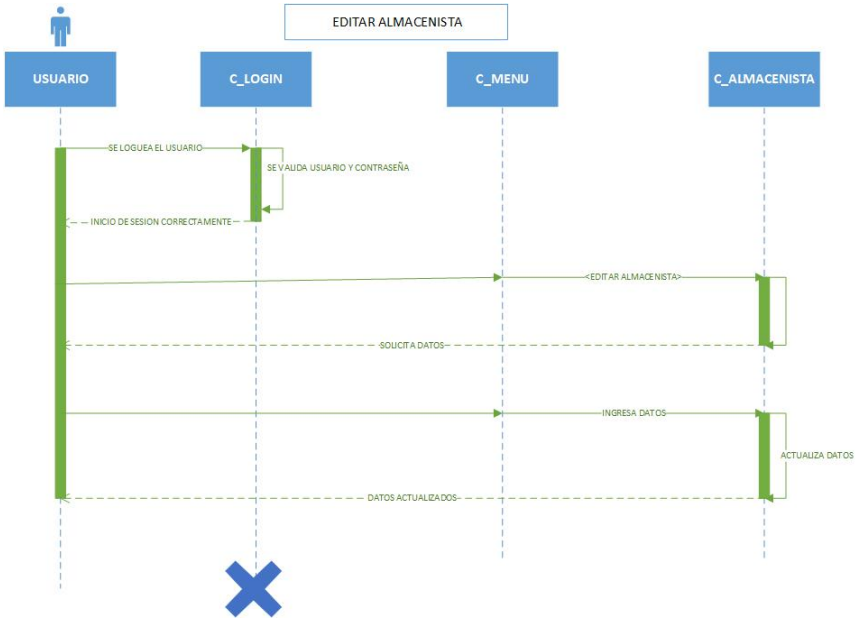

Figura 13: Diagrama de secuencia de la opción editar almacenista para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 14, se realizó el diagrama de secuencia de la opción calcular costo materia prima.

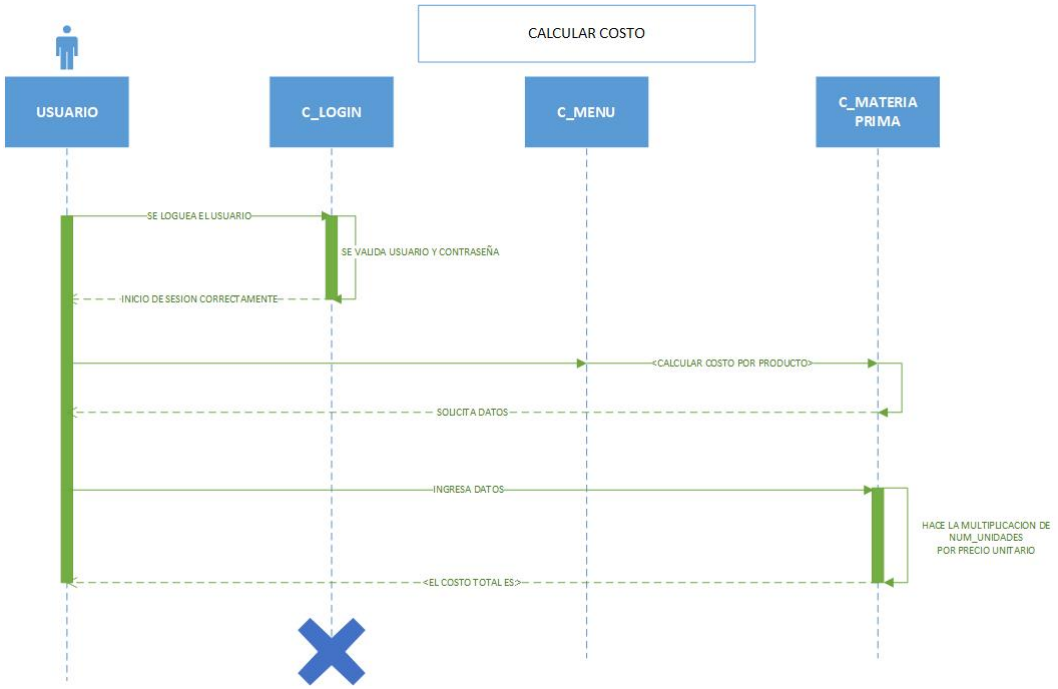

Figura 14: Diagrama de secuencia de la opción calcular costo materia prima para el "Sistema de gestión de inventario y personal para maquiladora".

En la figura 15, se realizó el diagrama de secuencia de la opción imprimir reporte almacenista.

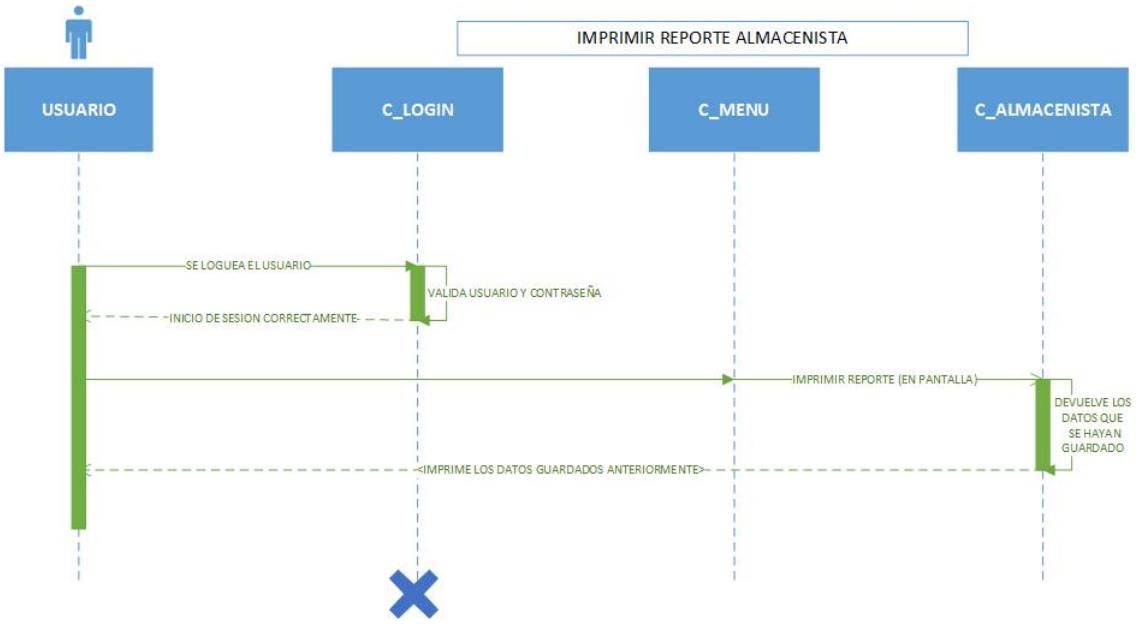

Figura 15: Diagrama de secuencia de la opción imprimir reporte almacenista para el "Sistema de gestión de inventario y personal para maquiladora".

7.5. Código

En esta sección, se mostrarán algunos fragmentos del código que se elaboró para el "Sistema de gestión de inventario y personal para maquiladora".

7.5.1. Agregar almacenista

Para este fragmento del código, dentro de la clase C_sistema se declararon métodos virtuales (como se explicó anteriormente en el diagrama de clases). Posteriormente se creo la subclase C_almacenista y dentro de ésta se creo un método agregar. El código de esta subclase es el siguiente:

```
#ifndef _ALMACENISTA_H
#define _ALMACENISTA_H
#include "SISTEMA.h"

using namespace std;

//C_ALMACENISTA HEREDA ATRIBUTOS Y METODOS DE C_SISTEMA
class C_almacenista : public C_sistema
{
        public://ATRIBUTOS
            int ID_almacenista;
            string nombre_almacenista;

        public://METODOS
            C_almacenista();
            void agregar();
            void editar();
            void mostrar();
            void imprimir_reporte();
};

//CONSTRUCTOR
C_almacenista::C_almacenista()
{
    //SE INICIALIZA EL ID DE ALMACENISTAS EN 1
    ID_almacenista=1;
};
//METODO AGREGAR
void C_almacenista::agregar()
{
    ID_almacenista;
    //SE PIDE EL  NOMBRE DEL ALMACENISTA
    cout<<"Ingresa el nombre del almacenista"<<endl;
    getline(cin, nombre_almacenista);
}
```

Después se creo una clase llamada C_menú, que será la encargada de administrar las demás clases.

```
#ifndef _MENU_H
#define _MENU_H
#include "ALMACENISTA.h"
#include "MATERIA_PRIMA.h"
#include<vector>

using namespace std;

class C_menu
{
        public://ATRIBUTOS
            char opcion;
            vector<C_almacenista> dato2;
            C_almacenista objeto2;
            vector<C_materia_prima> dato3;
            C_materia_prima objeto3;
            int verificar,pos,flag2,flag3;

        public://METODOS
            C_menu();
            void menu_general();
            void menu_almacenista();
            void menu_materia_prima();
};
//CONSTRUCTOR DE MENU
C_menu::C_menu()
{
    pos=-1;
};
```

Se crearon vectores de tipo C_almacenista y C_materia_prima. Tambien se crearon objetos para las subclases C_almacenista y C_materia_prima. Ahora, para finalmente agregar un registro se hizo lo siguiente:

```
//BORRA PANTALLA Y DESPLIEGA EL METODO AGREGAR DE LA CLASE ALMACENISTA
system("cls");
objeto2.agregar();
dato2.push_back(objeto2);//SE GUARDA EL NOMBRE DEL ALMACENISTA
objeto2.ID_almacenista++;//SE DA EL INCREMENTO DEL ID PARA EL SIGUIENTE REGISTRO
cout<<"--------------------------------"<<endl;
cout<<"REGISTRO AGREGADO EXITOSAMENTE"<<endl;
cout<<"--------------------------------"<<endl;
```

El código mostrado esta dentro de un switch case, siendo la opción 1 de agregar registro almacenista. Básicamente lo que hace es, llama al método agregar a traves del objeto2, luego todos los datos son metidos dentro de un arreglo dinámico (la función push_back lo que hace es guarda los datos de atras hacia adelante). Luego se aumenta el ID para que la siguiente vez que se guarde un dato, el ID sea el inmediato superior o lo que es lo mismo, el que le sigue.

7.5.2. Agregar materia prima

Lo mismo aplica para este método. Solo se pondrá el código del método agregar de la clase C_materia_prima.

```
//SE SOLICITAN LOS DATOS DE LA MATERIA PRIMA
ID_materia_prima;

cout<<"Ingresa el nombre de la materia prima"<<endl;
getline(cin, materia_prima);

cout<<"Ingresa el nombre del proveedor"<<endl;
getline(cin, proveedor);

cout<<"Ingrese el precio unitario"<<endl;
cin>>p_unitario;
cin.ignore();

cout<<"Ingrese la cantidad total comprada"<<endl;
cin>>total_comprado;
cin.ignore();
```

7.5.3. Borrar almacenista

Para borrar un registro de almacenista, no se creo un método borrar como tal ya que la clase C_menú lo puede hacer. El código que se ocupa es el siguiente:

```
//BORRA LA PANTALLA Y DESPLIEGA EL METODO ELIMINAR ALMACENISTA
system("cls");
cout<<"Ingrese el ID"<<endl;
cin>>verificar;//INGRESA EL ID DEL ALMACENISTA QUE DESEA ELIMINAR
cin.ignore();
flag2=0;
for(int i=0; i<dato2.size();i++)
{
    if(dato2[i].ID_almacenista==verificar)
    {
            pos=i;
            dato2.erase(dato2.begin()+pos);//BORRA EL REGISTRO CON EL ID INGRESADO
            flag2=1;//NOS DICE SI ENTRO A ELIMINAR EL REGITRO SOLICITADO
    }
}
//VALIDA SI EL REGISTRO FUE ELIMINADO O NO
if(flag2==1)
{
    cout<<"--------------------------------"<<endl;
    cout<<"REGISTRO ELIMINADO EXITOSAMENTE"<<endl;
    cout<<"--------------------------------"<<endl;
}

else
{
    cout<<"--------------------------------"<<endl;
    cout<<"REGISTRO NO ENCONTRADO"<<endl;
    cout<<"--------------------------------"<<endl;
}
```

Este código lo que hace es, pide el ID del registro que se quiere borrar; posteriormente busca coincidencias, si encuentra invoca a la funcion erase y elimina la información almacenada en el nodo. En dado caso de que no encuentre coincidencias, mostrará el mensaje de "REGISTRO NO ENCONTRADO".

7.5.4. Borrar materia prima

Y lo mismo aplica para borrar registros de materia prima. Se presenta el código.

```
//LIMPIA PANTALLA Y DESPLIEGA EL METODO ELIMINAR DE LA CLASE MATERIA PRIMA
system("cls");
cout<<"Ingrese el ID"<<endl;
cin>>verificar;
cin.ignore();
flag3=0;
for(int i=0; i<dato3.size();i++)
{
    //BUSCA EL REGISTRO
    if(dato3[i].ID_materia_prima==verificar)
    {
        pos=i;
        dato3.erase(dato3.begin()+pos);//BORRA EL REGISTRO SOLICITADO
        flag3=1;//NOS DICE SI FUE ENCONTRADO  Y ELIMINADO EL REGISTRO
    }
}
//LE AVISA AL USUARIO SI FUE EXITOSO O NO ELIMINAR EL REGISTRO
if(flag3==1)
{
    cout<<"--------------------------------"<<endl;
    cout<<"REGISTRO ELIMINADO EXITOSAMENTE"<<endl;
    cout<<"--------------------------------"<<endl;
}

else
{
    cout<<"--------------------------------"<<endl;
    cout<<"REGISTRO NO ENCONTRADO"<<endl;
    cout<<"--------------------------------"<<endl;
}
```

8. Conclusiones

El sistema de gestión de inventario y personal para maquiladora fue concluido con las mejoras solicitadas por el cliente (vistas en el apartado de objetivos particulares). Cuenta con un login que ayuda a la seguridad del sistema; cuenta con un solo administrador para todo el sistema; el programa muestra un menú en el que el usuario podrá elegir que acción realizar y de esta manera agilizar la agregación, eliminación, visualización y edición tanto de materias primas como de almacenistas.

Consideramos que a futuro pueden ser implementadas mejoras como incluir más de un administrador, una mejor interfaz ó poder registrar directamente en el programa usuarios y contraseñas.

Referencias

[1] Daifuku. Historia de los almacenes automatizados en daifuku. [En línea]. Disponible: https://www.daifuku.com/mx/solution/technology/automatedwarehouse/ (Accedido: 30-may-2019).

[2] Wikipedia. Netsuite. [En línea]. Disponible: https://en.wikipedia.org/wiki/NetSuite (Accedido: 30-may-2019).

[3] Katana. Características de katana. [En línea]. Disponible: https://katanamrp.com/ (Accedido: 30-may-2019).

[4] Zoho. Características de zoho. [En línea]. Disponible: https://www.zoho.com/ (Accedido: 30-may-2019).

[5] M. E. Raffino. ¿qué es un lenguaje de programación? [En línea]. Disponible: https://concepto.de/lenguaje-de-programacion/#ixzz5mAyIPIld (Accedido: 30-may-2019).

[6] J. Luján. Paradigmas de programación. [En línea]. Disponible: https://ed.team/blog/paradigmas-de-programacion (Accedido: 30-may-2019).

[7] E. de Expertos Universidad Internacional de Valencia. Lenguaje de programacion c, origen y trascendencia. [En línea]. Disponible: https://www.universidadviu.com/lenguaje-programacion-c-origen-trascendencia/ (Accedido: 30-may-2019).

[8] D. Manzano. Historia del lenguaje de programación c++. [En línea]. Disponible: https://es.slideshare.net/DarioManzano/historia-del-lenguaje-de-programacion-c (Accedido: 30-may-2019).

[9] I. Medina. Programación estructurada. [En línea]. Disponible: https://medium.com/laboratoria-how-to/programaci%C3%B3n-estructurada-7fe400bae43d (Accedido: 30-may-2019).

[10] Álvaro Fito & Diego Yarza. Poo: Herencia, abstraccion y polimorfismo. [En línea]. Disponible: https://es.slideshare.net/equipo2/poo-herencia-abstraccion-y-polimorfismo (Accedido: 30-may-2019).

[11] Cavsi. ¿qué es un compilador? [En línea]. Disponible: http://www.cavsi.com/preguntasrespuestas/que-es-un-compilador/ (Accedido: 30-may-2019).