Pasar la nota de José y Miguel 18.0/10.0

Universidad de las Américas
Facultad de Ingeniería de Sistemas

PROGRAMA DE TITULACION INGENIERIA DE SISTEMAS

PROYECTO DE TITULACION

Sistema Computarizado de Apoyo al Control de Producción en Granjas Avícolas

ING. BERNARDINO CHANCUSIG.
Tutor

Edison Patricio Viñachi Bermeo

Quito 2001

Agradecimientos

A Dios, quien con su infinito amor me guió y alentó para no desmayar y así alcanzar un sueño.

También a todas aquellas personas que colaboraron para que el presente Trabajo de Titulación se lleve acabo con profesionalismo y responsabilidad.

Al ingeniero Bernardino Chanchusig, quien en su calidad de Tutor, entregó con generosidad su elevada capacidad para orientarme con acierto.

A la ingeniera Cecilia Hinojosa en su calidad de Examinadora, entregó su tiempo y conocimientos para que el presente trabajo tenga el profesionalismo y calidad deseados.

A todos los distinguidos catedráticos de esta prestigiosa Universidad, que estuvieron a lo largo de dos años apoyándonos e impartiendo su invalorable enseñanza. En particular a los ingenieros Juan Carlos Trujillo, y Nelson Subía, ya que sin su gestión no hubiera sido posible que esta carrera alcance los fines esperados.

A los directivos de GRUPO ORO, por darme las facilidades para realizar la investigación con la infraestructura en ella implantada.

A los veterinarios y jefe de producción, quienes de forma desinteresada brindaron su tiempo para incursionar en el mundo de la producción avícola.

Y de manera especial a mis familiares, por su confianza, su preocupación y su cariño incondicional.

Resumen Ejecutivo

Capitulo I

En este capitulo se abordara muy brevemente los inicios de la orientación a objetos hasta la presente fecha, él porque del desarrollo de la orientación a objetos y los conceptos de Objeto, Clase, Atributo, Operación, Método, Mensaje, Encapsulamiento, Herencia, Abstracción, Polimorfismo, Persistencia los mismos que son básicos entenderlos.

Capitulo II

Se aborda la metodología OMT, de describe muy brevemente las fases que lo componen, se describe los modelos de objetos, el dinámico, y el funcional.

En el Modelo de objetos se describe los conceptos y la nomenclatura que se utiliza para representar, de diagramas de objetos, los atributos, las operaciones y métodos, las asociaciones, la multiplicidad, los roles, la clasificación, la generalización y la herencia.

En el modelo dinámico se aborda los conceptos de sucesos, estados, escenarios, diagramas de estado, operaciones, condiciones y anidamiento de diagramas.

En el modelo funcional se define lo que un diagrama de flujo de datos, procesos, flujo de datos, actores, almacenamientos de datos y las especificaciones de operaciones.

En la parte final se detalla las distintas relaciones que existen entre los distintos modelos objetos vs. dinámico, y objetos vs. funcional.

En la segunda parte de este capitulo se aborda los procesos, modelos y las herramientas que se utilizan en la ingeniería de software para llegar a obtener un software de alta calidad, se describe los modelos más utilizar para desarrollar software, entre estos el secuencial, construcción de prototipos, el DRA, Incremental, espiral, ensamblaje de componentes, formal y los de cuarta generación. Se describe la herramienta de desarrollo utilizada (Ratinal Rose 98).

En la parte final de este capitulo se describe lo que es y en consiste una arquitectura cliente servidor. Como esta constituida, las ventajas y desventajas que tiene y se detalla las distintas pruebas que se realizar en esta arquitectura entre ellas en el cliente, en el servidor, de la base de datos, en las comunicaciones, y en el sistema.

Capitulo II

En este capitulo se detalla cual es la misión, visión y la situación actual de la organización, se describe los requerimientos del software, cual es la descripción general con la utilización del diagrama de casos de usos. Se detalla los requerimientos Específicos que tendrá el software y cual son los requerimientos de software.

INDICE

CAPITULO I: Marco Teórico

1. Introducción 1
1.1. Historia de la orientación a objetos 1
1.2. El porque de la orientación a objetos 1
1.3. Conceptos básicos de orientación a objetos 2
Objeto 2
Clase 3
Atributos 3
Operaciones 3
Método 3
Mensajes 3
Encapsulamiento 4
Herencia 4
Abstracción 4
Polimorfismo 4
Persistencia 4

CAPITULO II: Metodología

2. Metodología OMT(Técnica de Modelaje de Objetos) 5

2..1. Fases de OMT 5
2.1.1. Análisis 5
2.1.2. Diseño del Sistema 5
2.1.3. Diseño de Objetos 6
2.1.4. Implementación 6

2.2. Modelado de Objeto 6
2.2.1. Diagrama de objetos 6
Diagrama de clases 6
Diagrama de instancias 6

2.2.2. Atributos 6

2.2.3. Operaciones y métodos 7

2.2.4. Enlaces y asociaciones 8
2.2.4.1. Multiplicidad 8
2.2.4.2. Atributos de los enlaces 9
2.2.4.3. Roles 9
2.2.4.4. Clasificación 9
2.2.4.5. Cualificación 10
2.2.4.6. Agregación 10

2.2.5. Generalización y Herencia 11

2.3. Modelado Dinámico 12
2.3.1. Sucesos (eventos) 12

2.3.2. Estados 13

2.3.3. Escenarios y secuencia de sucesos 13

2.3.4. Diagrama de estados 14

2.3.5. Condiciones 14

2.3.6. Operaciones 15
Actividades 15
Acciones 15

2.3.7. Anidamiento de diagramas 16
Acciones de entrada y salida 17
Acciones internas 17
Transiciones automáticas 17
Sincronización de actividades concurrentes 17

2.4. Modelado Funcional 18
2.4.1. Diagrama de flujo de datos DFD´s 18

2.4.2. Procesos 18

2.4.3. Flujo de datos 19

2.4.4. Actores 19

2.4.5. Almacenamiento de datos 19

2.4.6. Especificaciones de operaciones 20

2.5. Relación entre los diagramas 21
2.5.1. Modelo de Objetos y Dinámico 21

2.5.2. Modelo de Objetos y Funcional 21

2.6. Resumen 22

2.7. Procesos y Herramientas 23
Modelo lineal 24
Modelo de construcción de prototipo 24
Modelo DRA 25
Modelo Procesos evolutivos 25
Incremental 25
Espiral 25
Ensamblaje de componentes 25
Modelo formal 25
Modelo de cuarta generación 26

2.7.1. Modelo Utilizado 26

2.7.2. Herramientas 26

2.7.2.1 Rational Rose 98 26

2.8. Software Cliente Servidor 28
2.8.1. Estructura de los sistemas cliente servidor 28
Servidor de archivos 29
Servidor de base de datos 29
Servidor de transacciones 29
Servidor grupo de trabajo 29

2.8.2. Ventajas y desventajas 30
Ventajas 30
Desventajas 30

2.8.3. Estrategias de pruebas en sistemas cliente servidor 30
Comprobación de funciones de aplicaciones 30
Comprobación del servidor 30
Comprobación de la base de datos 31
Comprobación de transacciones 31
Comprobación de comunicaciones en la red 31

2.8.3.1. Pruebas del cliente 31
Pruebas de interfaces gráfica de usuario (IGU) 31
Pruebas de documentación y ayuda 31
Pruebas de menús emergentes y operación con el ratón 31
Entrada de datos 31

2.8.3.2. Pruebas del servidor 32

2.8.3.3. Pruebas de la base de datos 32
Control de acceso 32
Comprobar Log´s y bitácora de la base de datos 32
Prueba de integridad y consistencia 32

2.8.3.4. Pruebas de comunicación 33
Hardware de red 33
Software de red 33
Prueba del ODBC 33

2.8.3.5. Pruebas del sistema 34
Pruebas de recuperación 34
Pruebas de seguridad 34
Pruebas de rendimiento 34

CAPITULO III: DESARROLLO DEL SISTEMA

3. Introducción 35

3.1. La empresa 35
3.1.1. Misión y Visión 35

3.1.2. Organigrama 35

3.1.3. Situación Actual 36

3.2. Especificación de Requerimientos 37
3.2.1.Propósito 37

3.2.2. Ambito 37

3.2.3. Definición y abreviaturas 37

3.2.4. Descripción General 38
3.2.4.1. Perspectiva del producto 38
3.2.4.1.1 Diagrama de casos y usos 38
3.2.4.2. Funciones del producto 39
3.2.4.3. Características de los usuarios 39
3.2.4.4. Restricciones generales 39
3.2.4.5. Asunciones y dependencias 39

3.2.5. Requerimientos Específicos 40
3.2.5.1. Requerimientos Funcionales 40

Requerimiento Funcional 1 40
Requerimiento Funcional 2 40
Requerimiento Funcional 3 41
Requerimiento Funcional 4 41
Requerimiento Funcional 5 41
Requerimiento Funcional 6 42

3.2.5.2. Seguridad 43

3.2.5.3. Requerimiento de interface externa 43

3.2.5.4. Requerimiento de software 43

Conclusiones y Recomendaciones 44

Bibliografía 45

ANEXOS

Anexo A: Documentos Básico
Anexo B: Diagrama Modelo de Objetos
Anexo C: Diccionario de Datos
Anexo D: Diagrama de Sucesos
Anexo E: Diagrama de Estados
Anexo F: Diagrama de Flujo de Datos
Anexo G: Pruebas
Anexo I: Manual de Instalación
Anexo H: Manual de Usuario

CAPITULO I

1. Introducción

En este capitulo se realizará una breve reseña del nacimiento y evolución de la metodología orientada a objetos. La segunda parte hablamos del porque de la metodología orientada a objetos cuales son sus razones para su desarrollo y gran acogida que ha tenido dentro del desarrollo de sistemas.

En su parte final se describe que es un objeto, una clase, una operación, un método, una herencia, una abstracción, un encapsulamiento, un atributo, un polimorfismo, conceptos básicos y fundamentales para el modelado de sistemas, estos conceptos se aplican al análisis, diseño e implementación de un sistema en la metodología orientada a objetos.

1.1. Historia de la Orientación a Objetos

La Orientación a Objetos (O.O.) nace en Noruega en 1967 con Krinsten Nygaard y Ole-Johan Dahl en el lenguaje llamado Simula 67, en el centro de cálculo noruego. Simula 67 introdujo por primera vez los conceptos de clases, corrutinas y subclases (conceptos similares a los lenguajes Orientados a Objetos de hoy en día).

En los 70's científicos del centro de investigación en Palo Alto Xerox (Xerox park) inventaron el lenguaje Small talk, que fue el primer lenguaje Orientado a Objetos puro es decir utiliza clases y objetos

En los años 80's Bjarne Stroustrup de AT&T Labs., amplió el lenguaje C para crear C++ que soporta la programación Orientada a Objetos. En esta misma década se desarrollaron otros lenguajes Orientados a Objetos como Objective C, Common Lisp Object System (CIOS), object Pascal, Ada y otros.

En el inicio de los 90's se consolida la Orientación a Objetos como una de las mejores maneras para resolver problemas. Aumenta la necesidad de generar prototipos más rápidamente (concepto RAD Rapid Aplication Developments). Sin esperar a que los requerimientos iniciales estén totalmente precisos.

En 1996 surge JAVA (extensión de C++). Su filosofía es aprovechar el software existente. Facilitar la adaptación del mismo a otros usos diferentes a los originales sin necesidad de modificar el código ya existente.

Del 98 a la fecha se desarrolla la arquitectura de objetos distribuidos RMI, Corba, COM, DCOM.

Actualmente la orientación a objetos parece ser el mejor paradigma, no obstante, no es una solución a todos los problemas. Trata de eliminar la crisis del software. Entre los creadores de metodologías orientadas a objetos se encuentran: G. Booch, Rambaught, Ivar Jacobson y Peter Cheng.

1.2. El porqué de la Orientación a Objetos

La orientación a objetos es el resultado de la crisis de la metodología o técnica estructurada, esta tendencia se acentúa y se hace notar de manera considerable en los proyectos de desarrollo de sistemas grandes, largos y complejos, donde se ha podido observar que los métodos estructurados han dado todo lo que tenían.

La necesidad de crear sistemas de una manera mucho más rápida y a un menor costo posible permite el nacimiento de esta nueva tecnología. Las técnicas orientadas a objetos simplifican el diseño de sistemas a través de la aplicación de una serie de conceptos que de apariencia sencilla, clara y simple, pero que representan un gran avance en la concepción e implementación de los mismos.
Pero tampoco debe esperarse que la orientación a objetos vaya a solucionar todos los problemas que actualmente existen para la creación, desarrollo y mantenimiento de sistemas que requieren las organizaciones. La orientación a objetos debe ser combinada, según el caso, con otras tecnologías o enfoques existentes.

Las técnicas orientadas a objetos simplifican la complejidad en el desarrollo de sistemas a través de la aplicación de una serie de conceptos de apariencia sencilla, clara y simple, pero que representan un gran avance en la concepción e implementación de los mismos.

Según Brooks "la complejidad del software es una propiedad esencial no accidental"[1], esta dificultad se deriva de cuatro elementos:

- Complejidad del dominio del problema.
- Dificultad de gestionar el proceso de desarrollo.
- Flexibilidad.
- Problemas de caracterizar el comportamiento del sistema.

Para resolver la complejidad se tienen tres formas:

- Descomposición.- divide el problema en partes cada vez más pequeñas.
- Abstracción.- toma las características esenciales de un objeto de acuerdo al observador.
- Jerarquía.- establece los patrones entre los objetos los mismos que son heredados por otros.

1.3. Conceptos Básicos de Orientación a Objetos

Existen algunos términos que son necesarios entender sobre la tecnología orientada a objetos.

Objeto
Clase
Atributo
Operación
Método
Mensaje
Encapsulamiento
Herencia
Abstracción
Polimorfismo
Persistencia

Objeto.- Es una entidad tangible que exhibe algún comportamiento bien definido.
En términos generales:
Una cosa tangible o intangible que puede comprenderse intelectualmente, o algo hacia lo que se dirige un pensamiento o acción.

Además un objeto modela alguna parte de la realidad, algo que está en el tiempo y en el espacio.

Otra característica de los objetos es que cumple un papel bien definido dentro del entorno de un problema, es decir que tiene fronteras definidas.

[1] Brooks F. No silver bullet-essence and accidents of software engineering IEEE.(1987). P 10-19

Entonces un objeto tiene estado, exhibe algún comportamiento bien definido y tiene una identidad única.

Para Rumbaugh es "sencillamente algo que tiene sentido en el contexto de la aplicación. Se definirá un objeto como un concepto, abstracción o cosa con límites bien definidos y con significado a efecto del problema que se tenga entre manos"[2].

Ejemplo: persona, precio, impuestos, una factura, deposito, crédito.

No todo lo que hay es un objeto, así atributos como el tiempo o el color no son objetos, pueden ser propiedades de los objetos.

Clase.- Es muy frecuente decir *clase* en lugar de *clase de objetos*. Los objetos de una clase tienen los mismos patrones de comportamiento y atributos. "La clase de objetos describe un grupo de objetos con propiedades (atributos) similares, con relaciones comunes con otros y con una semántica común"[3].

Los objetos de una clase comparten un propósito común, tienen los mismos patrones de comportamiento y tienen los mismos atributos. Una clase describe un conjunto infinito o finito de objetos individuales, entonces un objeto es una instancia de su clase en la cual se comparte atributos y operaciones con las demás instancias de la clase.

Se forman con la agrupación de objetos semejantes en forma y función, es decir la estructura y comportamiento de objetos similares definen una clase.

Según Booch "es un conjunto de objetos que comparte una estructura común y un comportamiento común"[4].

Ejemplo: compañía, departamento, animal, empleado.

Atributo.- Es el valor de un dato o datos que está almacenado en los objetos de una clase, que es único dentro su clase. Cada atributo tiene un valor para cada instancia del objeto, son valores puros de datos y no objetos, estos valores puros no posean identidad.

En ocasiones se puede omitir los atributos lo cual depende del grado de detalle que se desea tener en un modelo del objeto.

Ejemplo: nombre, edad, ciudad, color, dirección, teléfono, país.

Operación.- Son algoritmos que nos permiten procesan los datos que encapsula un objeto. Cada operación encapsulada por un objeto proporciona una representación de los comportamientos del objeto.

Método.- Es la implementación de una operación para una clase, además especifica como se va a controlar los datos de un objeto, pudiendo ser operaciones que pueden realizarse con los objetos de una clase en particular, o se podría decir que es la forma de controlar los datos de un objeto.

Ejemplo: El código para realizar una operación, calcular el total de una factura

---

[2] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p45
[3] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p46
[4] Booch G. *Análisis y Diseño Orientado a Objetos*, Addson-Wesley/Diaz de Santos,1996,p120

Mensaje.- Según Pressman "Son el medio a través del cual los objetos interactúan. Un mensaje estimula la ocurrencia de cierto comportamiento en el objeto receptor"[5].

Un mensaje es una solicitud para llevar a cabo una operación y se produzca un resultado, la solicitud invoca una operación entre uno o más objetos.

Encapsulado.- Según Martin "El empaquetado de datos y métodos se llama encapsulado. El objeto esconde sus datos de los demás objetos y permite el acceso a los datos mediante sus propios métodos para evitar corrupción de los datos del objeto"[6]

Se esconde los detalles de su implementación al usuario de un objeto, estos solo saben de las operaciones que se disponen y que puede solicitar del objeto, pero no saben como se lleva a cabo.

Herencia.- Es organizar un conjunto de clases de manera jerárquica para compartir atributos, operaciones definidas en las clases superiores. Una clase puede ser divida en subclases, entonces ésta heredará los atributos, operaciones de su clase superior, además puede tener sus propios atributos y operaciones que le permiten diferenciarse de las demás.

Con la herencia se tiene los siguientes beneficios: la reutilización, reduce la necesidad de volver a crear funciones comunes, reduce el código, reduce la superficialidad, incrementa el grado de compresión y congruencia.

Abstracción.- Según Rumbaugh "Consiste en centrarse en los aspectos esenciales inherentes de una entidad, e ignorar sus propiedades accidentales. En el desarrollo de sistemas esto significa centrarse en lo que es y lo que hace un objeto antes de decidir cómo debería ser implementado"[7]

Polimorfismo.- Es la implementación de una operación de distintas maneras, pero que cumplen una misma función; una operación puede tomar distintas formas.

Ejemplo: clase Archivo (ASCII, binario, imagen) operación Imprimir.

En conclusión se puede decir que el polimorfismo permite tener operaciones que tiene el mismo nombre dentro un sistema o dentro de una misma o diferente clase.

Persistencia.- Según Booch "Es la propiedad de un objeto por la que su existencia trasciende en el tiempo (es decir el objeto continua existiendo después de que su creador deja de existir) y/o espacio (es decir la posición de objeto varía con respecto al espacio de direcciones en el que fue creado)"[8]

---

[5] Pressman R, Ingenieria de Software un enfoque practico 4ta. Ed. (1999),p371, 372
[6] Martin J,Odell J, *Análisis y Diseño Orientado a Objetos*, ,1995,p17
[7] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p27
[8] Booch G. *Análisis y Diseño Orientado a Objetos*, Addson-Wesley/Diaz de Santos,1996,p86

CAPITULO II

2. Metodología OMT

La Técnica de Modelado de Objetos (OMT) utiliza el principio del modelado de objetos para el desarrollo de software, se construye diseños independientes, mejorando la compresión de los requerimientos, logrando diseños más depurados y generando sistemas fáciles de mantenimiento.

Es posible aplicar los conceptos de orientación a objetos a lo largo de todo el ciclo de vida de desarrollo del sistema, se traspasa las mismas clases de una etapa a otra agregando ciertos detalles propios de cada etapa.

En cada etapa genera modelos cuyos objetivos son:

- Probar una entidad física antes de construir.
- Mejorar la comunicación con el cliente.
- Visualizar.
- Reducir la complejidad.

Según Rumbaugh "no existe un único modelo correcto de una situación, solo existen modelos adecuados e inadecuados"[1].

OMT emplea tres modelos:

Modelo de objetos.-	describe la estructura estática de los objetos, mediante grafos que representan las clases y su relación entre ellas.
Modelo dinámico.-	describe los cambios con relación al tiempo, especifica e implementa los controles.
Modelo funcional.-	describe las transformaciones de los datos, forma en que se derivan los valores después de un cálculo.

El más importante es el modelo de objetos que describe como se está cambiando o transformándose antes de describir cuándo o como cambia.

2.1. Fases de OMT

2.1.1. Análisis

Se inicia con la definición del problema, el analista debe construir un conjunto de modelos que representan la situación real, con los cuales se extrae las propiedades más importantes. La modelación se la realiza en tres modelos; un modelo de los objetos donde se muestra la estructura estática, un modelo dinámico para entender su comportamiento e interacción, y un modelo funcional para especificar los requerimientos funcionales del sistema. En esta fase lo más importante es identificar sus partes esenciales y las relaciones entre ellas.

2.1.2. Diseño del Sistema

El siguiente paso es la etapa en la cual se toman decisiones de alto nivel con respecto a la arquitectura general del sistema. El sistema se organiza en subsistemas basados en la estructura del análisis así como en la arquitectura propuesta. Se deciden las características

[1] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p39

de rendimiento que hay que optimizar y de esta forma se ataca los problemas, y la forma en que interactuarán los modelos entre sí.

2.1.3. Diseño de objetos

Se construye un modelo de los objetos partiendo del que se construyó en el análisis, y no se parte de cero, al cual se incluye detalles de implementación. Se inserta detalles al modelo de acuerdo con la estrategia establecida durante el diseño del sistema. En esta fase se hace énfasis especial en las estructuras de datos y algoritmos apropiados para implementar cada una de las clases.

2.1.4. Implementación

Las clases y las relaciones desarrolladas durante el diseño de objetos se transforman en lenguaje de programación, sistema de administración de bases de datos y hardware. La programación debería ser una tarea menos importante y más mecánica que etapas de análisis y diseño, puesto que las decisiones fundamentales ya fueron tomadas en fases anteriores. Al realizar la implementación se debe tener muy en cuenta los criterios de la ingeniería de software.

2.2. Modelado de Objetos

Es el encargado de mostrar los objetos, las relaciones entre ellos y los atributos, proporciona una representación gráfica del sistema a través de diagramas de objetos.

2.2.1. Diagramas de Objetos

Los diagramas son concisos, fáciles de entender y es la notación gráfica de clases y sus relaciones entre sí. Hay dos diagramas de objetos:

- Diagrama de clases
- Diagrama de instancias

Diagrama de Clases

Describe muchas instancias de datos posibles o describen clases de objetos(caso general). Su notación es un cuadrado con el nombre de la clase en negritas.

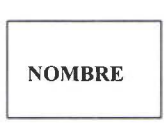
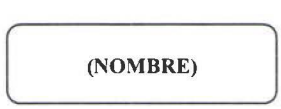


Diagrama de Instancia

Describe la forma en que un cierto conjunto de objetos se relaciona entre sí. Se usan para clarificarnos los diagramas de clases complejos. Su notación es mediante un cuadro con las esquinas redondeadas, su nombre entre paréntesis y en la parte superior del cuadro y en negritas.



2.2.2. Atributos

Es el valor de un dato que está en los objetos de una clase, el nombre del atributo es único dentro su clase.

La notación se detalla en la segunda parte del cuadro de clases separados del nombre de la clase por una línea horizontal, el nombre del atributo puede ir acompañado de detalles opcionales tales como tipo y valor por omisión, el primero ira acompañado por dos puntos, el segundo por el signo igual.

Los cuadros de objetos no tienen la línea de división.

Clase con sus atributos

Persona
Nombre: cadena Edad: entero Sexo:cadena=F

Objetos con sus valores

(Persona)
Juan Garcia 24 M

(Persona)
María Pérez 52 F

Gráfico tomado de Rumbaugh [2]

2.2.3. Operaciones y Métodos

La operación es la transformación que se aplica a la clase por parte de los objetos. Todo objeto "conoce" su clase y la implementación correcta de la operación.

Todos los objetos de una clase comparten las mismas operaciones. Una misma clase puede ser aplicada a varias clases distintas tomando el nombre de operación polimórfica.

Un método es la implementación de una operación para una clase.

Si una operación tiene métodos para distintos clases se debe tener presente que todos los métodos tengan igual número, tipo de argumentos y tipo de valor resultan.

Su notación es en la parte inferior del cuadro de clase, el nombre de cada operación pueden ir acompañado de detalles opcionales, como lista de argumentos entre paréntesis separados por comas, se puede indicar el tipo y nombre, también el tipo de resultado. Estos van precedidos por dos puntos. Las operaciones pueden omitir en diagramas de alto nivel.

Persona
Nombre Edad
Cambiar Nombre Cambiar edad

Objeto geométrico
Color Posición
Mover (delta:vector) Seleccionar(p:Punto):boolean Rotar(ángulo)

En resumen la notación de una clase en el modelo de objetos es el siguiente:

[2] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p49

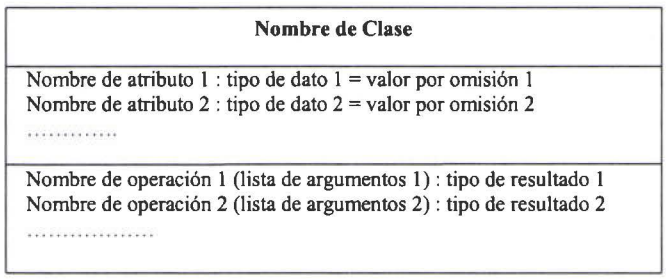
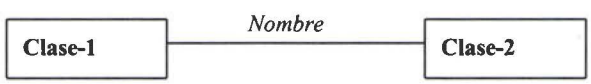


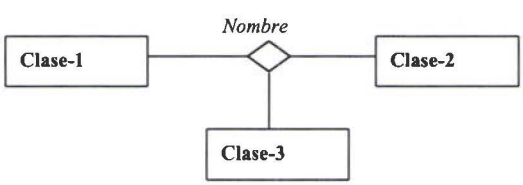
Gráfico tomado de Rumbaugh[3]

2.2.4. Enlaces y Asociaciones

Los enlaces muestran las relaciones entre instancias de objetos, o asociaciones. Las asociaciones describen enlaces con semántica y estructura comunes. Estas suelen aparecer como verbos dentro de la definición del problema.
La notación para las asociaciones es una línea entre las clases, los nombres de las asociaciones se pone en cursiva y se lo puede omitir si las clases tiene una solo asociación y su significado es obvio. Es recomendable organizar las clases para su lectura de izquierda a derecha.



Las asociaciones pueden ser binarias, ternarias, o incluso de orden superior. Sin embargo, la mayoría son binarias. Las asociaciones ternarias (n-arias) se representan mediante un diamante de cuyos vértices (tres de ellos) salen tres líneas hacia las clases respectivas



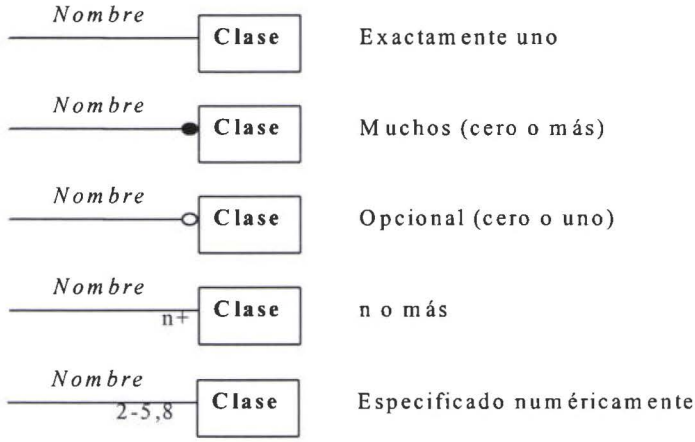
2.2.4.1. Multiplicidad

La multiplicidad limita el número de objetos relacionados, describe como "una" o "muchas" aunque en la realidad puede ser un número entero no negativo o incluso mediante un intervalo.

Ejemplos: "1" (exactamente una)
"1+" (uno o más)
"3-5" (entre tres y cinco inclusive los extremos)
"2,4,8" (dos, cuatro ocho)

La notación utilizada símbolos especiales al extremo o extremos de la línea de asociación. Los terminadores especiales que representan valores frecuentes de multiplicidad son:

[3] RumbaughJ,Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos, 1995, p52*

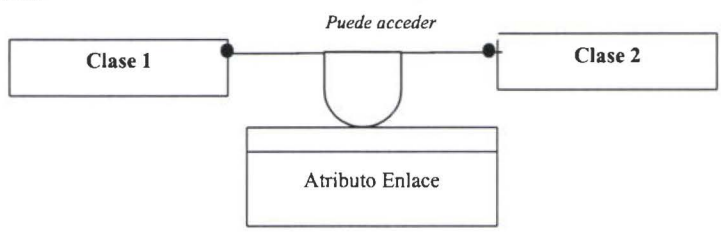


La multiplicidad depende de los límites del problema y de suposiciones, por tal motivo requisitos débiles hacen multiplicidad incierta.

2.2.4.2. Atributos de los Enlaces

Un atributo es una propiedad de los enlaces de una asociación. En OMT la notación es un cuadro ligado a la asociación mediante un lazo; el cual puede tener uno o más atributos de enlace en la segunda región del cuadro. Esta notación tiene que tener coherencia entre los atributos de los objetos y los atributos de los enlaces.

Las asociaciones muchos-a-muchos son las que mejor representan estos atributos de enlaces ya que una propiedad del enlace no se puede asociar a uno solo de los objetos sin perder información.



Por lo general, las asociaciones que necesitan atributos son asociaciones muchos-a-muchos, pues no es posible hacer que el atributo pase a formar parte de alguna de las entidades relacionadas. Sin embargo, también es posible que sean necesarios realizar en otros tipos de multiplicidad, o en asociaciones ternarias.

2.2.4.3. Roles

Un nombre de rol es el que identifica de forma única un extremo de una asociación. En una asociación binaria, se tienen dos posibles roles, o papeles que juegan las entidades relacionadas. Los diagramas pueden incluir los nombres de los roles de las asociaciones. Cada rol en una asociación binaria que identifica un objeto o conjunto de objetos asociados con el objeto al otro extremo

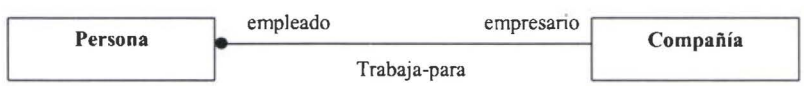


Gráfico tomado de Rumbaugh[4]

[4] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p62

Los roles sirven para distinguir entre los objetos que están directamente conectados con un objeto, por tal motivo estos nombres deben ser únicos. Además los roles son un atributo derivado de la clase fuente.

La utilización de nombres de roles es opcional pero puede resultar más fácil que asignar un nombre de asociación. Es decir, a veces se puede prescindir del nombre de asociación si se utilizan nombres de roles. Por otro lado, el rol es redundante dependiendo del nombre de la clase, por lo que también podría prescindirse de él.

En asociaciones de grado tres o más, no se pueden recorrer desde un extremo hasta el otro, así que los nombres de rol no representan atributos derivados de las clases participantes.

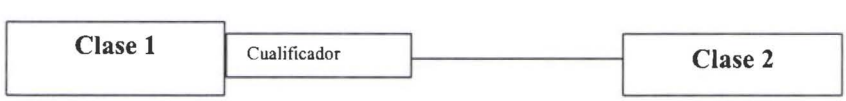
2.2.4.4. Clasificación

En la mayoría de las asociaciones que involucran más de un elemento, el orden de los mismos no es importante. Cada uno se procesa como si fuera simplemente un miembro de un conjunto no ordenado. Sin embargo, en algunas ocasiones el orden es importante o están ordenados explícitamente. Si se tiene por ejemplo una tarjeta que contiene varios puertos (digamos, para procesamiento de voz), cada uno de ellos puede requerir ser identificado en forma especial por un número particular. En este caso, mantener el orden de los elementos en un extremo de la asociación es obligatorio. Como lo muestra la figura, el ordenamiento se representa simplemente escribiendo "{orden}" en el extremo "muchos" del enlace.

2.2.4.5. Cualificación

Es un atributo que reduce la multiplicidad uno-a-muchos y muchos-a-muchos de una asociación, el cualificador se aplica el extremo "*muchos*" pasando la multiplicidad a uno-a-uno, esto no siempre es posible realizarlo.

Su notación es un cuadrado pequeño en el extremo de la línea de asociación de la clase a la cual calificamos.



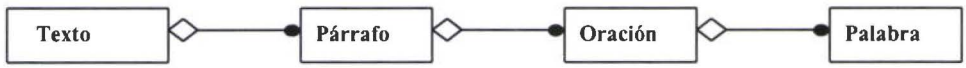
2.2.4.6- Agregación

Agregación significa "composición". Se utiliza en asociaciones donde uno de los objetos juega el papel de componente, o parte, de otro. También puede verse como si una clase estuviera compuesta por un conjunto de otros objetos. Un ejemplo es la especificación de las partes que componen un producto ensamblado.

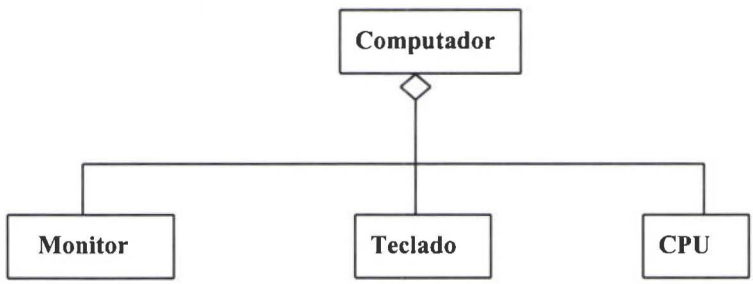
Se identifica con la ley transitiva y la no-simétrica. Es decir, si A es parte de B, y B es parte de C, entonces A es parte de C. Por otro lado, si A es parte de B, B no puede ser parte de A. Además, es común que las partes obtengan cierta información del objeto al que pertenecen.

La agregación es simplemente un caso particular de una asociación, pero se hace la diferencia en la notación para agregar mayor información al modelo. Se dibuja de la misma

forma que la asociación pero se agrega un pequeño diamante del lado del objeto "unidor" de partes, como se muestra en el gráfico.



También se puede utilizar el gráfico siguiente cuando se tiene múltiples composiciones.

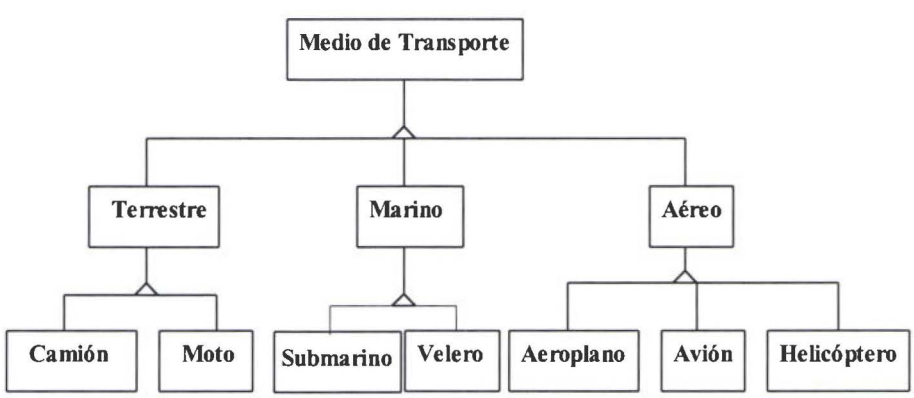


La decisión de utilizar la agregación es subjetiva y muchas veces arbitrarias, pero a la postre no debería causar problemas serios.

2.2.5. Generalización y Herencia

La herencia, la generalización y la especialización son conceptos que, representan una de las diferencias principales de la orientación a objetos con respecto a los desarrollos tradicionales. Provee uno de los mecanismos fundamentales para lograr el objetivo de máxima reutilización y fácil mantenimiento del sistema.

La notación para representar la herencia es un triángulo que conecta la superclase con sus subclases.



El símbolo indica que las subclases tendrán la misma estructura y entenderán los mismos mensajes que su superclase, además de los elementos de especialización que se decida incluir.

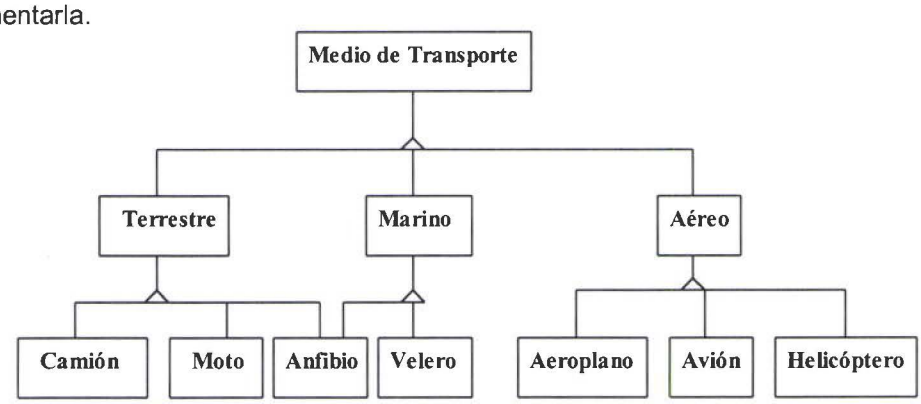
En algunos casos puede requerirse especificar qué criterio fue el que se usó para determinar la especialización. Si es así, el criterio puede escribirse cerca del triángulo de herencia. Sin embargo, por lo general el criterio es lo suficientemente claro por lo que se omite del diagrama.

Hay que tener cuidado de que la cantidad de niveles dentro de una jerarquía de herencia no sea demasiado alto. Si esto ocurre, probablemente se pueda solucionar meditando un poco y reestructurando el diseño, para que sea más fácil de entender (y de reutilizar). No existe

un número de niveles, pero: "una jerarquía de herencia de dos o tres niveles de profundidad es ciertamente aceptable; diez niveles de profundidad serían probablemente excesivos; cinco o seis niveles podría o no resultar apropiado" [5]

Muchas veces surge la pregunta de cuando utilizar agregación y cuándo utilizar generalización. Claramente, ambos son conceptos diferentes, pero pueden usarse para lograr un mismo objetivo funcional. La agregación relaciona dos instancias, dos objetos diferentes donde unos es parte del otro. La generalización relaciona clases y es una forma de estructurar la definición de un mismo objeto. Tanto la superclase como la subclase se refieren a propiedades de un mismo objeto. La generalización es una relación "is-a" en tanto que la agregación es una relación "part-of".

Si una clase hereda características de más de una superclase, se presenta un fenómeno conocido como herencia múltiple. La utilización de herencia múltiple siempre crea discrepancias pues produce ciertas situaciones que es necesario enfrentar a la hora de implementarla.



De la mano con la herencia, se encuentra el concepto de clase abstracta, que en este contexto se define como una clase que posee al menos un método abstracto o diferido. Un método abstracto es aquel que no se encuentra implementado, sino que lo será en las subclases. El objetivo de manejar clases abstractas es organizar características comunes a varias clases. También puede resultar útil para unir clases que participan en la misma asociación o agregación. Algunas clases abstractas podrían surgir incluso naturalmente a partir del contexto de la aplicación; otras son introducidas artificialmente para reforzar la reutilización de código. Existe una notación particular para representar operaciones abstractas, que deber ser implementadas por las subclases para que tomen sentido. Simplemente se escribe "{abstracto}" junto al método.

2.3. Modelado Dinámico

El modelo de objetos nos permite entender la estructura del sistema. Nos dice cómo está conformado y cuáles son las relaciones entre sus partes. Sin embargo, es necesario entender sus relaciones a través del tiempo. ¿Cuál es la secuencia de pasos que se dan al ocurrir cierto evento?. ¿Qué otros eventos internos se generarán?. ¿Cómo reacciona un determinado objeto ante un estímulo?. ¿Qué mensajes y en qué orden se generan en determinado punto de ejecución?.

"Los conceptos más importantes son los sucesos(eventos), que son estímulos externos, y los estados, que son los valores de los objetos."[6]

[5] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p71

[6] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p123

2.3.1. Sucesos (eventos)

Un suceso es algo que ocurre en un punto en un período de tiempo. Un suceso no posee duración. Por ejemplo: "el usuario presiona una tecla", "la comunicación se interrumpe". En realidad no es que los sucesos no posean duración, sino que es tan pequeña, que pueden ser considero como nulos.

Los sucesos se dan en secuencia, uno después de otro, o se dan en forma simultánea, en cuyo caso se supone que no se relacionan entre sí. Se dice que dos sucesos son concurrentes cuando no tiene relación causal o cuando no se afectan entre sí.

También puede verse como una transmisión de información en un solo sentido, de un objeto a otro, el cual puede generar otro suceso, controlado ahora por él mismo, en respuesta al primer suceso.

Todo suceso transmite información a otro objeto. Algunos son señales de que ha sucedido algo y otros son aporte de valores de datos, a esto se lo conoce como atributo. Se muestra entre paréntesis "( )", después del nombre de la clase de suceso, también es opcional escribir los atributos.

De acuerdo con el problema en particular, se podría establecer una especie de modelo de objetos, o al menos la jerarquía de clases que relaciona los sucesos. Como instancias, cada suceso puede contener atributos, cuyo valor representa la información que se pasa de un objeto a otro. Sin embargo, dependiendo del contexto, no necesariamente resultarán en clases, pero es bueno no perder de vista este punto.

También se debe tomar en cuenta las condiciones de error que no tiene nada distinto solo muestra o interpreta un "error".

2.3.2. Estados

Un estado representa el intervalo entre sucesos o intervalos de tiempo ya que tardan un tiempo en concluir. El estado de un objeto depende de que suceso le ha afectado anteriormente, los estados y los sucesos son duales entre sí; un suceso separa a dos estados, y un estado separa dos sucesos.

El estado de un objeto se refiere a una combinación particular de los valores de sus atributos. A través del tiempo, el estado de los objetos cambia debido a los estímulos que recibe por parte de otros objetos en tiempo de ejecución. Un estímulo particular de un suceso sobre otro se conoce como sucesos. A raíz de un suceso, y dependiendo del estado en que esté un objeto, se desata una determinada secuencia de operaciones. El diagrama de estados es una herramienta que abstrae el conjunto de estados, sucesos y transiciones entre estados para una clase determinada. El modelo dinámico consiste de una serie de diagramas de estados, correspondientes a aquellas clases que presentan un comportamiento dinámico importante.

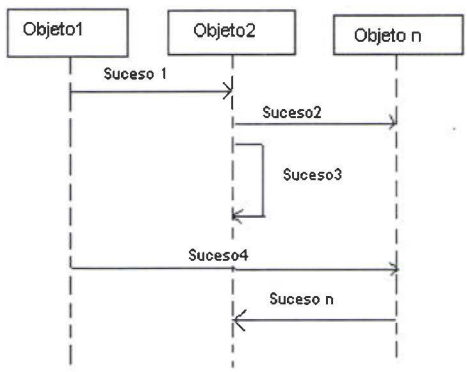
2.3.3. Escenarios y Secuencia de Sucesos

"Un escenario es una secuencia de sucesos que se producen durante una ejecución concreta de un sistema"[7]. En un inicio, se pueden considerar casos de interacción con el usuario, en donde éste inicia la secuencia de sucesos. Se supone que el sistema se encuentra en cierto estado, y se determinan los sucesos que genera el sistema para con el usuario (por ejemplo, presentación de cierta información). Este enfoque puede verse incluso como una construcción de un prototipo del sistema, pues define la forma en que el sistema se comporta a partir de las entradas del usuario.

---

[7] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p126

Una vez definido un escenario, se deben identificar los objetos que generan los eventos, y los objetos que los reciben. Para visualizar mejor la secuencia de eventos, se pueden escribir las clases involucradas, y trazar líneas de una a otra, etiquetadas con el nombre del suceso, y con una flecha que indica la dirección. Las líneas se dibujan siempre una debajo de la otra para no perder de vista la secuencia que siguen los sucesos y no su temporización exacta. Un diagrama como el descrito se conoce como seguimiento de trazo de sucesos.



Un objeto puede estar en un estado determinado, y cambia hacia otro ante un evento particular. Es decir, los eventos cambian el estado en que se encuentra un objeto, lo que quiere decir que cambian los valores de los atributos.

Si los sucesos representan "puntos de tiempo", los estados representan "líneas o intervalos de tiempo" cuya longitud depende del tiempo en que ocurran dos eventos dados. Por lo tanto, un estado posee una duración, y por lo general está asociado a una actividad. Es decir, mientras el objeto está en un estado, probablemente esté realizando alguna operación en respuesta a un estímulo.

Al definir los estados, no consideramos todas las combinaciones posibles de valores que pueden tomar los atributos, sino que todas las combinaciones que se comportan de la misma forma ante el mismo conjunto de eventos se agrupan dentro de un único estado.

2.3.4. Diagramas de estados

Los diagramas de estados relacionan sucesos y estados. En general, pueden verse como grafos cuyos nodos son rectángulos con esquinas redondeadas que representan los estados, y las aristas son flechas correspondientes a los sucesos de un estado a otro. Cada nodo contiene el nombre del estado que representa, y cada flecha está etiquetada con el nombre de los sucesos. Los cambios de estado producidos por los sucesos también son llamados transiciones.

Un estado por lo general reacciona ante un conjunto de sucesos. La transición que se activa es la que corresponde a los sucesos que ocurra primero. Así, dada una secuencia de sucesos predeterminada, el diagrama contiene la información acerca de la secuencia de estados por la que pasa un objeto de la clase en estudio.

Cada diagrama de estados corresponde al comportamiento que comparten todas las instancias de una única clase en particular.

Los diagramas pueden corresponder a dos tipos de comportamientos: comportamientos que involucran un ciclo continuo, o que involucran un ciclo finito. En cualquier caso es necesario conocer cuál es el estado inicial del diagrama. Esto se hace dibujando una flecha que sale de un círculo relleno hacia el estado inicial puede o no estar rotulado para indicar las condiciones iniciales. De la misma forma, si el objeto llega en algún momento a un estado final, en donde termina su actividad, esos estados deben ser resaltados. Los estados finales se representan como un círculo relleno dentro de un círculo más grande, que puede estar rotulado para indicar condiciones finales.

El modelo dinámico del sistema es el conjunto de los diagramas de estado que fueron construidos, los cuales representan un patrón (como las clases) correspondiente a un rango probablemente infinito de posibles secuencias de sucesos. Puede decirse que un escenario es al modelo dinámico, como un diagrama de instancias es al modelo de objetos.

2.3.5. Condición

Algunas veces puede que sea necesario verificar el cumplimiento de cierta condición booleana lógica antes de cambiar de estado, aún cuando ocurra el suceso asociado a la transición. Las condiciones actúan como protección en la transición. Una transición con protección se dispara cuando se produce su suceso, pero si la condición de protección es verdadera. En estos casos, la condición se especifica entre corchetes junto al nombre del suceso asociado a la transición. Puede ser que de un mismo estado salgan transiciones diferentes con el mismo suceso, pero con diferentes condiciones (complementarias).

2.3.6. Operación

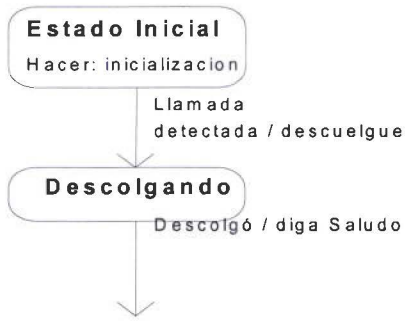
Los diagramas considerados hasta ahora sólo describen los estados y sus transiciones. Sin embargo, eso resulta de poca utilidad si no se especifican también las operaciones que disparan los sucesos.

Las operaciones pueden ser de dos tipos:

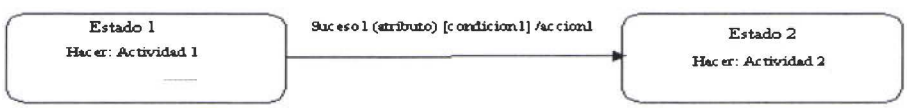
- **Actividades**
  Una actividad es una operación que toma un intervalo de tiempo para realizarse. Las actividades se asocian a los estados. Involucran operaciones continuas como desplegar una imagen, decir un mensaje, etc. Así como las operaciones secuenciales que terminan por sí solas después de un cierto tiempo. La notación "hacer: A" dentro de la caja de un estado indica que la actividad A empieza al entrar al estado y termina al salir de él. La finalización de la actividad puede estar dada por la interrupción de un suceso, o porque simplemente fue completada. El estado mismo puede establecer el control sobre el desarrollo de la actividad.

- **Acciones**
  Una acción es una operación instantánea. Las acciones se asocian a los sucesos, por ejemplo, colgar o descolgar una línea telefónica. Su estructura y control internos no interesan, de lo contrario, estaríamos ante una actividad. Las acciones también involucran operaciones internas de control, como incrementar una variable

o fijar un atributo. El nombre o la descripción de la acción aparece en el diagrama junto al nombre del suceso, luego de un slash ("/").



En síntesis la notación para los diagramas de estados quedaría así:



Como se puede observar el nombre del estado se escribe en negritas, las actividades se escriben dentro del cuadro de estado anteponiendo la palabra reservada "hacer:", que describe la actividad. Sobre la flecha transición se escribe el suceso opcionalmente se escribe entre paréntesis uno a más atributos, entre corchetes pueden ir las condiciones y las acciones van antecedidas de un "/". Esto no significa que estas estructuras sean obligatorias en todos los diagramas, sino que son opcionales.

2.3.7. Anidamiento de Diagramas

Cuando se manejan sistemas de cierto grado de complejidad resulta conveniente estructurar los diagramas de estado de forma que sea más fácil su entendimiento y construcción. En particular, una actividad dentro de un estado puede ser expandida como un diagrama de estados de un nivel más bajo, donde cada estado representa un paso de la actividad hasta llegar a un estado final. También un estado puede ser simplemente el nombre de otro diagrama de estado. Cada evento que entra al estado debe aparecer también como entrada en el subdiagrama; cada evento que sale del estado debe aparecer como salida del otro, y viceversa. La carencia de un rótulo de suceso indica que la transición se dispara automáticamente cuando ha terminado la actividad del estado.
Por otro lado, si los estados y los sucesos mismos se modelan como clases, es posible simplificar el diagrama aplicando relaciones de herencia. Los estados pueden tener subestados que heredan las transiciones de sus padres, por lo tanto, no sería necesario especificar las transiciones de los superestados en los subestados. Un superestado se dibuja como un gran rectángulo con las esquinas redondeadas que contiene todos sus subestados, los cuales a su vez pueden ser subestados. Las transiciones que salen del rectángulo grande del superestado son heredadas, o sea, realizadas por todos los subestados por lo que no es necesario representarlas.

Por ejemplo, el super estado "Llamada Activa" contiene una serie de estados que heredan la transición de su superclase. Por lo tanto, en cualquier momento que se detecte el evento "usuario cuelga" en cualquiera de los subestados, se causará una transición hacia el estado final. Un diagrama Anidado es una forma de generalización de estados y es una "relación–o". Para que un objeto este en el diagrama de alto nivel, tiene que estar precisamente en uno de los estados del diagrama anidado.

Entonces toda transición o acción que se aplica a un estado es aplicado también a todos los subestados, salvo que se invalide por una transición equivalente desde el subestado.

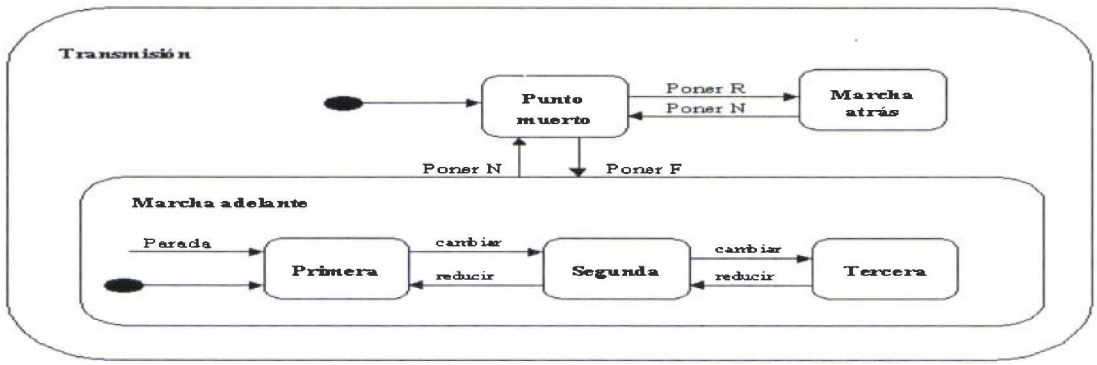


Gráfico tomado de Rumbaugh[8]

En cuanto a los sucesos, también pueden ser modelados como clases, y por lo tanto es posible establecer una jerarquía de generalización con herencia entre ellos. Los subsucesos heredarían la estructura de sus padres y también pueden disparar las transiciones que dispararían sus ancestros. La representación de la jerarquía se hace al igual que se representa en los diagramas de objetos.

Se debe también tener muy en cuenta los conceptos de Acciones de entrada y salida, Acciones internas, Transición automática y Sincronización de actividades concurrentes, los mismos que nos permitirán refinar más la notación y el modelo.

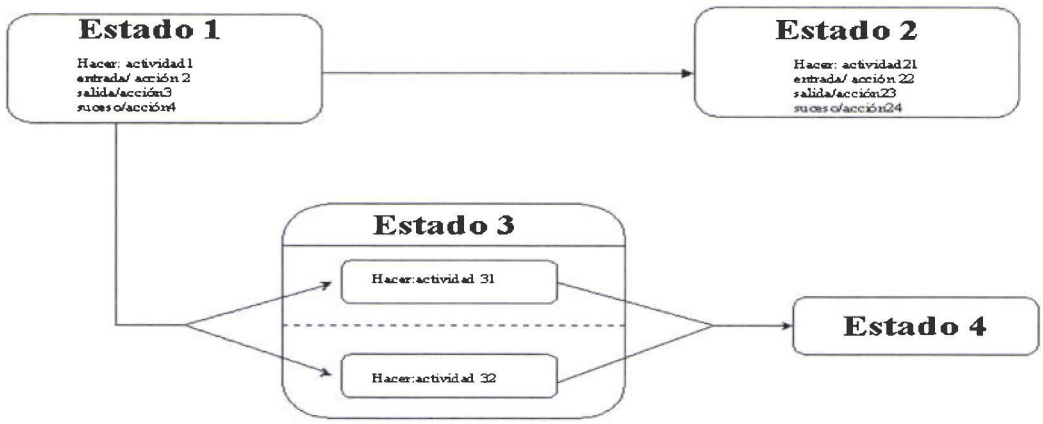
- **Acciones de entrada y salida.-** Son alternativas que nos permite asociar acciones de un estado que muestran las transiciones de entrada o salida en un estado. Las acciones de entrada son más comunes tener que las de salida. Su muestran dentro de cuadro de estado, después de las palabras reservadas "entrada o salida" y del carácter "/". Siempre que se entra o sale del estado se ejecuta primero la acción de entrada o salida.

- **Acciones internas.-** Estas transiciones hacen que al realice un suceso no se cambie su estado, es decir no se producen una salida del estado al no ejecutarse las acciones de entrada y de salida. Se muestran dentro del cuadro de estado y van seguidos por un "/" con el nombre de la acción.

- **Transición automática**.- Estas transiciones se ejecutan cuando se satisface las condiciones de protección, sin no se cumple estas condiciones la transición esta activa hasta que se cumplan o hasta que un suceso active otra transición. Su notación es una fecha sin nombre.

- **Sincronización de actividades concurrentes**.- En ocasiones es necesario que un objeto lleve a cabo dos o más actividades concurrentemente. Entonces es necesario que se termine las actividades para que el objeto avance al otro estado. Su notación es un cuadrado con las esquinas redondas dividido en dos zonas, la superior para el nombre de la actividad compuesta, en parte inferior se subdivide por una línea intercortada que divide a las actividades independientes. Al entrar y al salir se tiene unas fechas bifurcada.

[8] Rumbaugh J, Blaha M, Premerlani W, Eddy W, Lorensen W. *Modelado y diseño orientado a objetos*,1995, p139



2.4. Modelado Funcional

El modelo funcional es el tercero y último de los diagramas de la metodología OMT para lograr determinar los aspectos básicos de un sistema: su constitución, su comportamiento, y su funcionalidad.

En él se muestran los flujos de información que entran y salen del sistema, y la forma en que realizan las transformaciones computacionales del caso para que se produzcan las salidas adecuadas.

Los modelos funcionales especifican los resultados de un cálculo o proceso sin importar cuándo o cómo se producen. En esta metodología, el modelo especifica el significado de las operaciones del modelo de objetos y las acciones en el modelo dinámico.

Al igual que el modelo dinámico, no todas las aplicaciones requieren de un modelo funcional. A continuación se describen los elementos principales diagramas que utiliza.

2.4.1 Diagrama de flujo de datos DFD´s

Los diagramas de flujo de datos que especifican las operaciones y restricciones del sistema. Un diagrama de flujo de datos DFD es un grafo que muestra el flujo de elementos de información (datos) a través de un conjunto de procesos. Los procesos realizan operaciones que transforman los datos de entrada en datos de salida, hacia otros procesos, hacia entidades (como empresas u oficinas), o hacia repositorios de información (lugar donde se almacenan hasta que sean requeridos para otros cálculos).

El DFD no muestra información de control ni de tiempo, como cuándo se realiza una operación. Eso se especifica en el modelo dinámico. Tampoco se preocupa por la constitución ni la relación entre los objetos, lo cual es parte del modelo de objetos mismo.

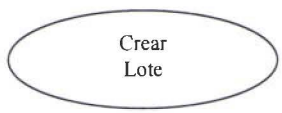
Se tiene los siguientes elementos de un DFD:

- Procesos
- Flujos de datos
- Actores
- Almacenamientos de datos

2.4.2 Procesos

Un proceso es una serie de pasos que transforma los valores de los datos. Los procesos más simples no son más que funciones que toman valores de entrada y generan un resultado sin efectos laterales. Por ejemplo, funciones matemáticas. Los procesos más complejos pueden estar compuestos de DFD's mismos, y tener efectos sobre almacenamientos de datos u objetos externos.

Los procesos se dibujan como elipses que contienen en su interior una descripción de la transformación que realizan; idealmente, su nombre. Cada proceso posee un número fijo de datos de entrada y de salida, representados como flechas con la identificación de los datos.



Los diagramas solo muestran qué entra, y qué sale de un proceso determinado, pero la forma en que se producen las salidas a partir de las entradas también debe especificarse como se verá más adelante.

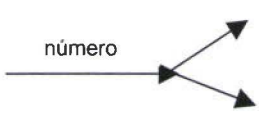
Algunos procesos complejos pueden descomponerse en DFD's separados, compuestos de procesos que a su vez pueden subdividirse, hasta llegar a un DFD que contiene procesos "atómicos" los cuales no necesitan dividirse más.

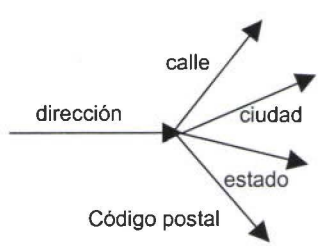
Los procesos son implementados como métodos en el modelo de clases del sistema. La clase muchas veces corresponderá a uno de los flujos de entrada al proceso. Sin embargo, en otros casos la clase no aperecerán en forma tan directa.

2.4.3 Flujo de datos

Un flujo de datos conecta la salida de un objeto o proceso con la entrada de otro objeto o proceso. El flujo simplemente funciona como una tubería que transporta el dato de un lado a otro sin modificarlo.

Un flujo se dibuja como una flecha entre el productor y el consumidor. La flecha se identifica con el nombre del dato. Si el dato se envía a varias partes a la vez, se puede dibujar como una flecha que se divide en algún punto para formar dos o más flechas que transportarán las copias del valor





Algunas veces, los diagramas no muestran los objetos de donde vienen o hacia donde se dirigen los flujos de datos. Si esto ocurre es porque el diagrama se encuentra en un nivel inferior con respecto al diagrama principal, por lo que deberá evaluarse en el contexto del diagrama de nivel más alto para determinar el origen o destino de los flujos.

2.4.4 Actores

Un actor es un objeto que dirige un flujo de datos de entrada o de salida, según el objeto produzca o consuma información. También son llamados terminadores, pues hacen que el

grafo de flujos de datos termine, al representarse como generadores o recolectores de datos.

Un actor se dibuja como un rectángulo para mostrar que es un objeto. Los actores se considera que se ubican en los límites del grafo, aunque en la realidad están fuera de él.

2.4.5 Almacenamiento de datos

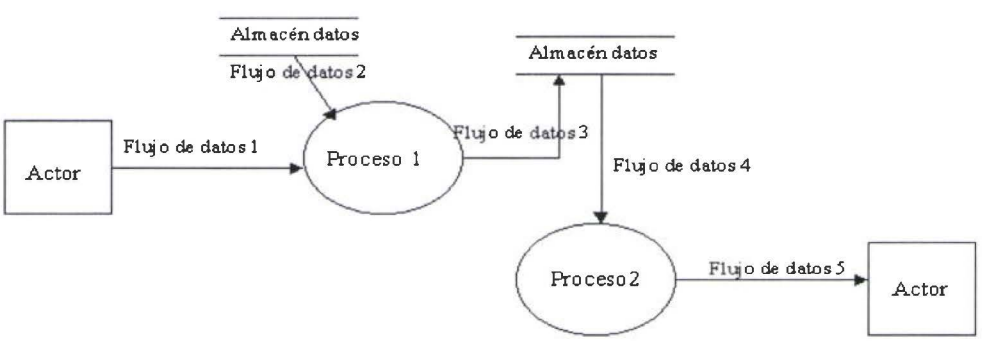
Es un objeto que guarda los datos para que sean accesados posteriormente. A diferencia de un objeto activo, estos objetos "pasivos", responden a solicitudes de almacenamiento y acceso a los datos.

Los almacenamientos se dibujan como un par de líneas paralelas que encierran el nombre del repositorio. Hacia este diagrama pueden dirigirse flechas, que significan una modificación entro en la estructura.

Las modificaciones pueden ser individuales o masivas, borrados, o actualizaciones. Si la flecha más bien sale, significa que se permiten operaciones de recuperación de información, que también pueden ser de algún campo, de un registro, o masivas. Las flechas pueden estar etiquetadas con el nombre del dato que transportan hacia y/o desde.

La estructura del repositorio y la descripción de las operaciones debe hacerse en el modelo de objetos del sistema.

Los DFD´s son útiles para mostrar la funcionalidad de un sistema. Los diagramas se pueden anidar hasta una profundidad arbitraria pero cada nivel debe ser coherente y compresible con la funcionalidad global del modelo.



2.4.6 Especificaciones de operaciones

Los procesos contenidos en los DFD's deben corresponder, en forma directa o indirecta, a operaciones en el diagrama de objetos. Si no pasan a ser operaciones directamente, debe garantizarse que el proceso es realizado por alguna función de algún objeto en el modelo. De lo contrario, el sistema estaría perdiendo funcionalidad.

Los procesos de los niveles más bajos son operaciones. Los de niveles más altos también pueden ser estructurados de esa forma, aunque no necesariamente.

La descripción o especificación de las operaciones puede realizarse de varias maneras como:

- Funciones matemáticas
- Tablas de valores de entrada y de salida (enumeración, para casos pequeños)
- Precondiciones y postcondiciones
- Tablas de decisión
- Pseudocódigo
- Lenguaje natural

La especificación, además debe incluir los siguientes elementos:

- Nombre de la función
- Entradas: parámetros de entrada; debe incluirse el nombre, tipo, y orden
- Salidas: parámetros de retorno; debe incluir el nombre, tipo, y orden
- Transformación: especificación de la función, Los valores de entrada y salida y los efectos que se tuvo en los objetos que servirán de operandos.
- Restricciones: incluye los supuestos que se hacen para garantizar el correcto comportamiento de la función, las relaciones que deben mantenerse entre dos objetos al mismo tiempo, y las relaciones entre los valores de dos objetos.

Si se describe la transformación en términos de un pseudocódigo, hay que tomar en cuenta que existen muchos algoritmos para resolver un problema, y que se debe tratar de escoger el más adecuado en términos de ciertos criterios de eficiencia.

La especificación descrita anterior solo muestra la interfase externa de la función. Se dice que recibe, que produce, y qué es lo que hace, así como los efectos laterales que puede tener. Sin embargo, internamente, a nivel de implementación, puede contener muchas otras cosas, como objetos temporales para realizar su labor. La estructura interna no es relevante para las personas que la utilizan, sino solo para las encargadas de darle mantenimiento. La descripción externa debe contener estrictamente lo necesario para que pueda ser utilizada la función de la forma en que fue concebida, ni más ni menos.

2.5 Relación entre los diagramas

Como resultado de análisis se obtiene tres modelos que describen cada uno la situación del problema a ser resulto, el modelo de objetos describe la parte estática y su relación de los objetos es decir muestra los 'hacedores' de las operaciones, el modelo dinámico aquellas situaciones que cambian con el tiempo, es útil para implementar aspectos de control al sistema y el modelo funcional es la transformación de los valores. Entonces los tres modelos se relacionan entre sí.

2.5.1 Modelo de objetos y dinámico

El diagrama de estados representa el comportamiento de algunas de las clases de objetos del modelo estático. El comportamiento se refiere a los valores de los atributos del objeto en un determinado estado, y la forma en que cambian dependiendo de las operaciones y/o sucesos que recibe. Los estados del modelo dinámico restringirán los valores que pueden tener los atributos de los objetos, y los sucesos pueden pasar a ser operaciones dentro del modelo de objetos.

La jerarquía de estados que se construya en el modelo dinámico implicaría una jerarquía restrictiva sobre el modelo de objetos, en el sentido de que la jerarquía de estados restringe o especializa los valores que pueden tomar los atributos de un objeto. Un subestado puede

significar contar con más o menos atributos, lo que implicaría especializar la clase en el modelo de objetos.

Por otro lado, el modelo dinámico de una clase es heredado por sus subclases. Estas heredan tanto los estados como las transiciones de sus ancestros. Sin embargo, también pueden tener su propio diagrama de estados, el cual no debería entrar en conflicto con el diagrama que posee por herencia.

2.5.2 Modelo objetos y funcional

El modelo funcional muestra a través de los diagramas de flujo de datos, ciertas relaciones que existen entre los objetos. Muchas veces es posible identificar flujos que entran a un proceso como instancias de clases en el modelo de objetos. El proceso probablemente sea un método de otro objeto. Por lo tanto, el modelo funcional determina las relaciones a nivel de implementación de los objetos.

Los almacenamientos de datos también son objetos, o quizá partes de objetos. Un flujo de entrada a un repositorio representa una operación de actualización, y uno de salida es una consulta.

Los flujos de datos son en su mayoría valores en el modelo de objetos. Muchos flujos de datos son simplemente valores puros, como números, secuencias de caracteres, o listas de valores. Los valores puros pueden ser modelados como clases e implementados como objetos, pero no poseen identidad. Un valor puro no posee estado ni modelo dinámico. Las operaciones sobre ellos generan solo valores también puros y no tienen efectos laterales ejemplo: operaciones aritméticas. Sin embargo, otros flujos de datos sí representan objetos "normales" en el modelo estático.

2.6 Resumen

La meta del análisis es desarrollar un modelo que representa lo que el sistema debe hacer. El modelo está compuesto de tres submodelos: objetos, dinámico, y funcional.

1. Obtenga una descripción del problema.

2. Elabore un modelo de objetos.

   - Identifique clases.
   - Construya un diccionario de clases, atributos, y asociaciones.
   - Agregue asociaciones.
   - Agregue atributos de objetos y asociaciones.
   - Organice y simplifique utilizando herencia.
   - Agrupe clases en módulos, con base en funcionalidad y acoplamiento.
   - Verifique caminos de acceso usando escenarios e itere los pasos anteriores hasta que sea necesario.

   Modelo de objetos = diagrama de objetos + diccionario

3. Construya un modelo dinámico.

   - Prepare escenarios de secuencias típicas de interacción.
   - Identifique eventos entre objetos y prepare un seguimiento de eventos para cada escenario.
   - Prepare un diagrama de flujo de eventos para el sistema.
   - Desarrolle un diagrama de estados para cada clase que tenga un comportamiento dinámico importante.

- Verifique la consistencia y completitud de los eventos compartidos entre los diagramas de estados.

Modelo dinámico = diagramas de estados + diagrama de flujo de eventos general.

4. Construya un modelo funcional.

   - Identifique valores de entrada y de salida.
   - Construya diagramas de flujo de datos para mostrar las dependencias principales.
   - Describa cada función.
   - Identifique restricciones.
   - Especifique criterios de optimización.

   Modelo funcional = DFDs + restricciones

5. Verifique, itere y refina los tres modelos.

6. Agregue operaciones descubiertas durante la construcción del modelo funcional al modelo de objetos. No muestre todas las operaciones durante el análisis; sólo las más importantes.

7. Verifique que las clases, asociaciones, atributos y operaciones son consistentes y completas según el nivel de abstracción actual. Compare los tres modelos con la definición del problema, y verifique los modelos utilizando los escenarios.

8. Desarrolle escenarios más detallados incluyendo condiciones de error, como variaciones de los escenarios básicos. Verifique más los tres modelos con estos escenarios.

9. Itere los pasos anteriores hasta que sea necesario.

Documento de Análisis = Definición del problema + Modelo de Objetos + Modelo Dinámico + Modelo funcional

2.7. Proceso y Herramienta

Para el desarrollo de un software de alta calidad y en un tiempo previsto no solo es necesario la metodología sino también se debe considerar el *proceso, métodos* y las *herramientas* que se pueden utilizar.

Gráfico tomado de Pressman[9]

[9] Roger S. Perssman *Ingeniería del Software un enfoque práctico*,McGraw Hill cuarta edición, p18

Los *procesos* definen el marco teórico para el conjunto de áreas claves del proceso, estas áreas son la base del control de la gestión del proyecto de software y establece el contexto en el que se aplican los métodos técnicos.

Los *métodos* indica como construir técnicamente el software, abarca tareas que van desde el análisis de requerimientos, diseño, construcción de programas, pruebas y mantenimiento. Estos métodos dependen de un conjunto de principios básicos, tecnológicos, actividades de modelado u otras técnicas descriptivas.

Las *herramientas* proporcionan un soporte automático o semi-automático para el proceso y los métodos. "Cuando se integran herramientas para que la información creada por una herramienta la pueda utilizar otra"[10], se tiene un sistema de soporte para el desarrollo del software llamado *ingeniería del software asistido por computadora* (Computer Aided Software Engineering CASE).

Para resolver los problemas reales se debe incorporar una estrategia de desarrollo la misma que se llama modelo de procesos o paradigmas de la ingeniería de software

Para lo cual se selecciona un modelo de procesos de acuerdo a la naturaleza de la aplicación, así como los métodos, herramientas, los controles y las entregas a realizarse.

Cada uno de los modelos de procesos se caracteriza por ayudar y coordinar un proyecto de software, entre los que tenemos:

- Modelo lineal o secuencial
- Modelo de construcción del prototipo
- Modelo DRA
- Modele de Procesos evolutivos
  - Incremental
  - Espiral
  - Ensamblaje de componentes
- Modelo Formal
- Modelo de cuarta generación

Modelo lineal o secuencial.- Se llama también *ciclo de vida básico* o en *cascada*, se sugiere un desarrollo de software que comienza en el nivel de sistema y continua con el análisis, diseño, codificación, pruebas y mantenimiento.

Este paradigma es el más antiguo y más extensamente utilizado, al mismo que se le a puesto en duda por:

- Rara vez se tiene proyectos secuenciales.
- Los usuarios no exponen todos sus requerimientos.
- Los clientes deben tener paciencia para ver los sistemas.
- Se tiene atrasos innecesariamente e involuntarios cuando se trabaja en equipo.

Modelo de construcción del prototipo.- Se utiliza cuando el cliente define los requerimientos de forma general sin llegar al detalle o cuando el responsable no esta familiarizado con el proyecto a desarrollase.

El prototipo se inicia con la recolección de requerimientos en la cual se definen y encuentran objetivos globales del software. Este prototipo es usado y evaluado por parte del usuario para refinar los requerimientos del software.

La utilización de construir prototipos puede ser problemática por las siguientes razones:

- El cliente muchas veces no sabe que es un prototipo

[10] Roger S. Perssman *Ingeniería del Software un enfoque práctico*,McGraw Hill cuarta edición, p18

- Por hacer lo más pronto no se toma en cuenta la calidad del software o la facilidad de mantenimiento.
- Se puede llegar a utilizar sistema operativo y/o lenguajes de programación inadecuados.

Modelo DRA.- Desarrollo rápido de aplicaciones (Rapid Application Developnemt RAP) es una adaptación del modelo lineal pero a alta velocidad, en el cual se utiliza un enfoque de construcción de componentes que se pueden integrar después en un solo conjunto. Esto se logra cuando se comprenden los requerimientos y el ámbito del proyecto.

Se presenta inconvenientes en los siguientes casos:

- En proyectos grandes, donde se requiere un gran número de equipos de trabajo.
- Se requiere el compromiso de parte del cliente y desarrollado.
- Cuando los riegos técnico son altos, uso de nuevas tecnologías.

Modele de Procesos evolutivos.- Este modelo es interactivo permite desarrollar software cada vez más completos, se tiene los siguientes tipos:

Incremental.- Se combina el modelo lineal (repetidamente) con la construcción de prototipos. En cada secuencia lineal se produce un incremento de software. Al utilizar este modelo el primer producto es esencial (núcleo), en el cual se afronta requerimientos básicos.

En cada una de las entregas o incrementos se tiene un producto operacional. Este modelo es útil cuando no se cuenta con el personal suficiente para todo el tiempo, es así los primeros incrementos se puede implementar con pocas personas y si es el caso se puede ir añadiendo personas de acuerdo al incremento que se planea entregar.

Espiral.- " El software se desarrolla en una serie de versiones incrementales"[11]. Las primeras versiones pueden estar en papel o prototipo, llegando a tener las últimas versiones más completas. Este modelo maneja las siguientes actividades:

Comunicación con el cliente, se establece una comunicación entre el desarrollador y el cliente
Planificación, se define recursos, tiempos y otras tareas útiles para el proyecto.
Análisis de riesgos, se evalúa los riesgos técnicos y de gestión.
Ingeniería, se construye una o más representaciones de la aplicación.
Construcción y adaptación, se construye, se prueba, se da soporte a os usuarios.
Evaluación del cliente, se extrae las reacciones del cliente.

Es un enfoque realista del desarrollo de sistemas y a gran escala, este modelo demanda la consideración directa de los riesgos técnicos antes de que estos se conviertan en problemas.

Ensamblaje de componentes.- Incorpora muchas de las características del modelo espiral y exige un enfoque interactivo para la creación del software, configura las aplicaciones desde componentes llamados clases.

Se inicia con la identificación de clases, para lo cual se examina los datos que va a manejar la aplicación y el algoritmo que se va aplicar. Para de esta forma ir creando la biblioteca de clases, las mismas que las podrán ir utilizando de acuerdo a las necesidades, llegando a reducir en un 70% el desarrollo y un 84% el costo esto esta en relación directa con la robustez de la biblioteca.

Modelo Formal.- Esta acompañado por un conjunto de actividades matemáticas que conducen a la especificación del software, entonces se proporciona un mecanismo para

[11] Roger S. Perssman *Ingeniería del Software un enfoque práctico*,McGraw Hill cuarta edición, p28

eliminar problemas que son difíciles de superar con otros paradigmas, al aplicar análisis matemático.

Se puede llegar a tener un software libre de errores, sin embargo se tiene las siguientes preocupaciones:

- Es actualmente caro.
- Es difícil utilizar con clientes que no tienen conocimientos técnicos.
- No existe desarrolladores que conozca y dominen el modelo.

Modelo de cuarta generación.- El modelo de T4G se orienta hacia la posibilidad de especificar software usando formas de lenguaje especializado o notaciones gráficas que describan el problema a resolver en términos entendibles para el cliente.

Al igual que los otro modelos se inicia con la descripción de requerimientos por parte del cliente, que son traducidos directamente al prototipo operativo, hacer esto no es tan seguro puesto que el cliente puede no estar seguro de lo que quiere, o dar ambiguamente las especificaciones, entonces no se podrá utilizar una herramienta T4G. Por esta razón el dialogo cliente-desarrollador descrito por los otro modelos es muy útil para el enfoque T4G.

En aplicaciones pequeñas se puede ir directamente a un lenguaje no procedimental de cuarta generación. Pero en proyectos grandes el uso de T4G sin diseño causa más dificultades (poca calidad, mantenimiento pobre, poca aceptación del cliente).

Este modelo puede llegar a ser un modelo dominante cuando se lo combina con el modelo de ensamblaje de componentes.

2.7.1 Modelo Utilizado

El modelo utilizado para el desarrollo del software es modelo incremental el cual permite combinar elementos del modelo lineal o secuencial repetidamente y la construcción de un prototipo, consiguiendo de esta forma tener un producto operacional en cada uno de los incrementos realizados.

Con esto se consigue que en cada secuencia lineal se produce un incremento de software, logrando de esta forma entregar un producto operacional. Para que el usuario pueda realizar evaluaciones.

2.7.2 Herramienta

La ingeniería de software asistida por computadora puede ser tan sencilla como usar un software o tan compleja como administrar una base de datos.

Las herramientas CASE nos permiten "utilizar representaciones gráficas para ayudar a automatizar la planeación, el análisis, el diseño y la generación de software"[12].

La mayoría de herramientas CASE presten su apoyo en una actividad de ingeniería de software concreta, es decir no forma parte de un entorno integral. Aunque existe este inconveniente con los CASE es bastante eficiente utilizar.

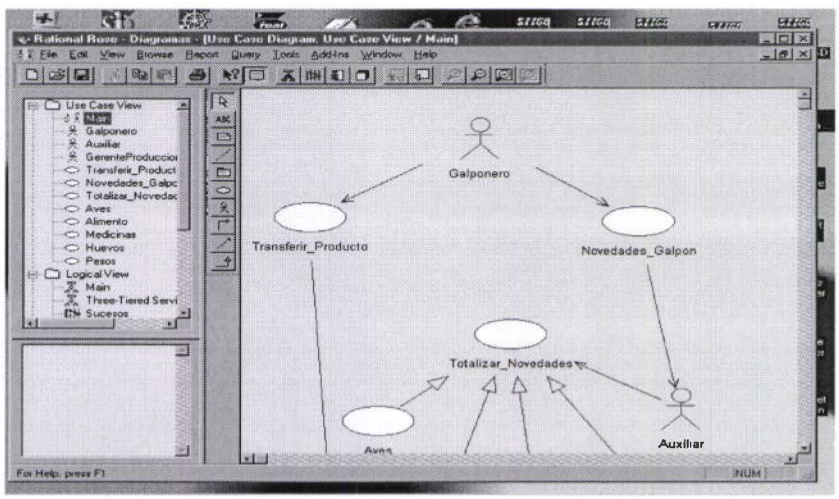
2.7.2.1 Rational Rose 98

Rational Rose permite una visualización, que se entienda, y que se refine los requerimientos antes de confiarles al código. Es una herramienta que modela a través del ciclo de vida, asegurando que se construya un sistema correctamente.

[12] Martin, J., *Análisis y Diseño Orientado a Objetos;* Prentice Hall HispoAmericana, 1994, p70

Con rational rose se puede crear los diagramas que se utiliza en UML así como los de OMT.

Su presentación es en una interface gráfica, la misma que esta dividida en tres ventanas. La primera una ventana ubicada en la parte superior izquierda es el explorador, en la cual están los diagramas y sus componentes ya creados. La segunda ubicada en parte inferir izquierda utilizada para ingresar comentarios u observaciones. La última y más grande ventana ubicada en parte derecha de la pantalla se utiliza para visualizar los diagramas construidos.



Rational Rose 98 visualiza los modelos en formas de vistas como se puede apreciar en la ventana del explorador.

Vista de Casos de Uso (Use Case View).- Se usa para crear los casos de uso, y de esta forma visualizar el sistema y su ambiente. En esta vista contiene el "Main" que nos muestra la vista global del modelo de caso de uso.

De esta forma el diagrama de caso de uso permite ver el sistema, que esta dentro del sistema y fuera así como reaccionan a los elementos externos, sin ver su funcionamiento interior.

Vista Lógica (Logical View).- Permite ver la estructura lógica del sistema con sus clases y sus relaciones.

Vista de Componentes (Component View).- Se visualiza la estructura física del sistema. Conecta clases a componentes y componentes a proyectos en un lenguaje de programación orientado a objetos.

Vista de Despliegue (Deployment View).- Muestra la conexión entre los procesadores, periféricos y ubicación de los procesos a los procesadores. Entonces contiene el Diagrama de Despliegue.

Además considera todas las etapas del ciclo de vida (análisis, diseño y parte de la implementación), permite la reutilización de clases, tiene repositorio centralizado, se lo puede usar con la metodología UML, OMT, BOOCH.

Rational Rose genera de un diccionario de datos desde un modelo usando objetos de Microsoft Word Object Link Embedding. Lo cual nos permite:

- Obtener el reporte de Vista Lógica con la especificación de clase.

- Obtener el reporte de Vista de Componentes con la especificación de componentes.
- Personalizar un reporte para colocar la sintaxis de atributos y la inclusión de campos de documentación.

Las ultimas versiones permite trabajar en formato HTML, para de esta forma facilitar el trabajo en equipo. Se publican los modelos en la internet y se los puede visualizar con cualquier explorador.

Requisitos de Hardware y Software

Sistema operativo	Microsoft Windows 95/98/NT
Procesador	Pentium II o Superior
Memoria	32 MB pero es recomendable 64MB
Espacio Disco	120 MB

Además se tiene versiones disponibles para Unix y Linux

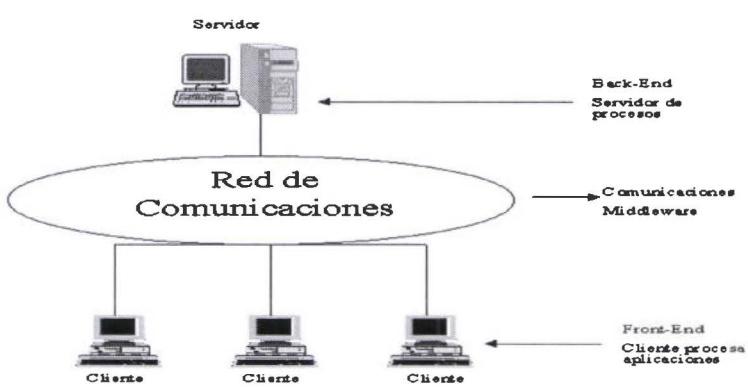
2.8. Software Cliente Servidor

El concepto de cliente/servidor (C/S) proporciona una forma eficiente de utilizar los recursos de máquina, de tal forma que la seguridad y fiabilidad de entornos mainframe se traspasa a la red de área local. A esto hay que añadir la potencia y simplicidad de los ordenadores personales.

La arquitectura C/S es modelo para el desarrollo de sistemas de información, en el cual las transacciones se dividen en procesos independientes que cooperan entre sí para intercambiar información, servicios o recursos.

2.8.1 Estructura de los Sistemas Cliente Servidor

En una estructura C/S se tiene un Computador Servidor el que se encarga de proporcionar la información, y las computadoras de nivel inferior denominados clientes que se encargan de solicitar servicios.



Como se puede apreciar los componentes principales de este esquema C/S son tres clientes, los Servidores y la infraestructura de comunicaciones.

Los *Clientes* interactúan con los usuarios con interfaces gráficas. Frecuentemente se comunican con procesos auxiliares que se encargan de establecer conexión con los servidores, enviar el pedido, manejar las fallas y realizar actividades de sincronización y seguridad.

Generalmente se tiene las siguientes funciones:

- Manejo de la interface de usuario.
- Captura y validación de los datos de entrada.
- Generación de consultas e informes sobre las bases de datos.

Los *Servidores* proporcionan un servicio a los clientes y devuelven los resultados. En algunos casos existen procesos auxiliares que se encargan de recibir las solicitudes del cliente, verificar la protección, activar un proceso servidor para satisfacer el pedido, recibir su respuesta y enviarla al cliente. Además deben manejar los interbloqueos, la recuperación ante fallas, y otros aspectos afines.

Generalmente se tiene las siguientes funciones:

- Gestión de periféricos compartidos.
- Control de acceso concurrentes a bases de datos compartidas.
- Enlaces de comunicación con otras redes de área local o extensa.
- Siempre que un cliente requiere un servicio lo solicita al servidor y este le responde.

La implementación de servidores puede ser:

- **Servidor de archivos**. El cliente solicita registros específicos de un archivo. El servidor los transmite a través de la res estos registros.

- **Servidor de base de datos**. El cliente envía en SQL. El servidor procesa, localiza la información y pasa a través de red únicamente los resultados al cliente.

- **Servidor de transacciones**. El cliente envía una solicitud que implica procedimientos remotos, los cuales pueden ser un conjunto de sentencias SQL, y el resultado es devuelto al cliente.

- **Servidor de grupo de trabajo**. El servidor permite la comunicación entre los clientes, mediante textos, imágenes, etc. que son del grupo de trabajo.

Además para que los clientes y el servidor se puedan comunicar se requiere de una *infraestructura de comunicaciones* la cual es la responsable de manejar los mecanismos de direccionamiento y transporte. La mayoría de sistemas C/S se basan en redes de área local las cuales tienen protocolos no orientados a conexión, lo cual implica que las aplicaciones deben hacer las verificaciones. La red debe tener características adecuadas de desempeño, confiabilidad, transparencia y administración.

Las características de una arquitectura C/S son:

- El servidor presenta a todos sus clientes una interface única y bien definida.
- El cliente no necesita conocer la lógica del servidor, solo su interface externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente.

2.8.2 Ventajas y desventajas de sistemas Cliente Servidor

Ventajas

- Aumenta la productividad.- usuarios utilizan herramientas que son familiares y fáciles de usar
- Menores costos de operación.- los costos de hardware son de distintos proveedores y cada vez más baratos.
- Mejora el rendimiento de la red.- se elimina la necesidad de transferir bloques de información por la red hacia la estación de trabajo. El servidor solo transfiere los datos requeridos por el cliente.
- Sistemas abiertos.- se pueden soportar múltiples ambientes, plataformas, sistemas de interfaces gráfica, protocolos de usuario, manejadores de bases de datos.
- Flexibilidad y escalabilidad.- son fáciles para migrar a nuevas tecnologías. Con bajo impacto en las aplicaciones.
- Reducción del trafico.- se descongestiona la red, la manipulación de los datos se lo realiza el cliente.

Desventajas

- Complejidad tecnológica.- al integrar una gran variedad de protocolos al tener hardware y software de distintos fabricantes.
- Existencia de costos ocultos.- como son licencias, nueva tecnología, que encarecen su implantación.
- Existen pocas herramientas que ayuden a determinar la mejor forma de dividir la aplicación entre el cliente y el servidor.
- Son muy complejos aplicarlos en sistemas de bases de datos distribuidas.

2.8.3 Estrategias de pruebas en sistemas Cliente Servidor

Para realizar las pruebas se debe tener considerar los recursos con los que se cuenta, ya que estos son muy variados, por un lado se tiene las personas que es el más importante, por tal razón se debe tener muy en cuenta las habilidades y conocimientos de la persona que va a realizar las pruebas. Otro es el hardware y demás equipos físicos. El Software y los datos de prueba. Si uno de ellos falla se reduce la eficiencia de las pruebas.

Entonces la comprobación de software C/S se realiza en tres niveles diferentes

- Las aplicaciones de cliente individuales se comprueban de modo desconectado, es decir no se considera el funcionamiento del servidor.
- Las aplicaciones de software de cliente y del servidor asociado se prueban al mismo tiempo, pero no se ejercitan específicamente las operaciones de red.
- Se comprueba la arquitectura C/S, incluyendo el rendimiento y funcionamiento de la red.

Aún cuando se efectúen muchas pruebas en cada uno de los niveles anteriores, es frecuente encontrar los siguientes enfoques de comprobación para aplicaciones C/S.

- **Comprobaciones de función de aplicación.** Se comprueban la funcionalidad de las aplicaciones cliente, con el fin de descubrir errores de funcionamiento.

- **Comprobaciones de servidor**. Se comprueban la coordinación y las funciones de gestión de datos del servidor. Se toma muy en cuenta el rendimiento del servidor (tiempo de respuesta y trasvase de datos).

- **Comprobaciones de bases de datos**. Se comprueban integridad de los datos almacenados en el servidor. Se examinan las transacciones enviadas por las aplicaciones cliente para asegurar que los datos se guardan, modifican y recuperen adecuadamente.

- **Comprobación de transacciones**. Se crea una serie de comprobaciones sobre las transacciones. Se tienen especial interés en la corrección de procesamiento, y también en los temas de rendimiento.

- **Comprobación de comunicaciones a través de la red**. Se verifica que la comunicación entre los nudos de la red sea la correcta, y que el paso de mensaje, las transacciones y el tráfico de red relacionado no tenga errores. También se pueden efectuar comprobaciones de seguridad de la red como parte de esta actividad.

2.8.3.1 Pruebas del cliente

Su objetivo es localizar o descubrir errores de software en el menor tiempo y esfuerzo posible, los cuales se presentan en el manejo de una aplicación por parte del usuario.

- Tiene la intención de descubrir errores
- Se tiene la probabilidad de descubrir errores hasta ahora no descubiertos.
- Se tiene existo si se encuentra un error no descubierto.

"Las pruebas no pueden asegurar la ausencia de defectos, sólo puede demostrar que existen defectos en el software"[13]

- **Prueba de interfaces gráficas de usuario (IGU)**

Se tiene las siguientes interrogantes que pueden servir de parámetros para la realización de una serie de pruebas IGU.

- Se permite al usuario utilizar el teclado o el ratón
- Se permite interrumpir su tarea y continuar más tarde
- Se puede ajustar el tamaño, mover y desplegar la ventana
- Está todo el contenido de la información dentro de la ventana accesible adecuadamente con el ratón, teclas de función, flechas de dirección y teclado
- Están operativas todas las funciones relacionadas con la ventana
- Están disponibles y desplegados apropiadamente en la ventana todos los menús emergentes, barras de herramientas, barras deslizantes, cuadro de diálogo, botones, iconos y otros controles.
- Cuándo se despliegan varias ventanas se resalta la ventana que esta activa
- Se permite una fácil navegación y salir del mismo sin causar inconvenientes
- Se cierra adecuadamente la ventana

- **Prueba de Documentación y de ayuda**

La prueba se enfoca en dos etapas. La primera una revisión técnica. La segunda prueba en vivo, se utiliza la documentación junto al uso del programa. El manejo del programa se sigue en función del manual:

- Describe con exactitud como conseguir cada modo de empleo
- Es exacta la descripción de cada secuencia de iteración

[13] Roger S. Perssman *Ingeniería del Software un enfoque práctico*,McGraw Hill cuarta edición, p352

- Los ejemplos son buenos y exactos
- Es relativamente fácil localizar ayuda en la documentación
- Se pueden solucionar problemas fácilmente con la documentación
- El documento es de fácil comprensión y rápida asimilación de la información
- Están descritos con gran detalle los mensajes de error para el usuario en el documento

- **Prueba de menús emergentes y operación con el ratón**

  - Se muestra la barra de menú apropiada y con el contexto apropiado
  - Son todas las funciones del menú accesibles con el ratón
  - Es correcto el tamaño, tipo y formato del texto
  - Es posible invocar las funciones del menú usando su orden alternativa
  - Son suficientemente claros los nombres de las funciones del menú
  - Si son necesario múltiples clics, estos son apropiadamente reconocidos
- **Entrada de datos**

  - Funcionan adecuadamente los modos gráficos de entrada de datos
  - Se reconocen adecuadamente los datos no válidos
  - Son claros los mensajes de entrada de datos

2.8.3.2 Pruebas del servidor

Su función es localizar mediante las pruebas de software y hardware errores que degraden el rendimiento del servidor.

Se puede realizar las siguientes pruebas:

- Verificar que el servidor no este apagado o desconectando (utilizar el comando ping)
- Reducir al máximo los recursos del servidor (espacio en disco, memoria, etc.)
- Realizar el ingreso de un registro completo y almacenarlo
- Consultar el registro anteriormente ingresado y comprobar su almacenamiento
- Realizar la actualización de un registro
- Consultar el registro actualizado y comprobar su correcto almacenamiento
- Borrar un registro y Consultar el registro
- Establecer conexiones adicionales al servidor (con otras aplicaciones)
- Solicitar un gran número de reportes al servidor simultáneamente

2.8.3.3 Pruebas de la base de datos

Se realiza la verificación de la base de datos en la arquitectura C/S.

- **Control de Acceso**.- Para el acceso a los usuarios, se debe comprobar que la aplicación tenga previsto el acceso a través de Login y Password, con el objetivo de mantener la seguridad de la información de la base de datos, controlar el acceso de intrusos, se controla que la aplicación tenga previsto un número máximo de oportunidades para ingresar el password respectivo, y que en caso de no hacerlo, se termine la aplicación y se emitan el mensaje de error.
- **Comprobar Logs y bitácoras de la base de datos**.- Se comprueba la existencia de los archivos logs de la base de datos para realizar un seguimiento de todas las transacciones que se realizan sobre la base de datos, permitiendo "monitorear" la base de datos.

- **Pruebas de integridad y consistencia**.
  - Anomalías por acceso concurrente a la base de datos

- Revisar caídas durante el procesamiento de las transacciones
- Errores lógicos que violan la suposición de que las transacciones respetan las limitaciones de consistencia de la base de datos
- Intentar violar una clave primaria
- Verificar si se realiza el borrado adecuado (en cascada)
- Ingresar datos corruptos en la base de datos (ya sea vía SQL o por la aplicación ver si este viola o no una regla)

2.8.3. 4. Pruebas de comunicaciones

Se procede a verificar que la comunicación entre los nodos de la red y el tráfico de red no tenga errores, para lo cual se comprueba:

- **Hardware de red**. Se deben comprobar todos y cada uno de los dispositivos físicos que conforman la red, cables de red, Hubs y tarjetas de red.

  Cables de red
  - Se identifica la topología, que topología se esta usando, bus lineal o estrella
  - Se verifica la continuidad en los cables coaxial o UTP (con ayuda de un multimetro)
  - En el cable UTP se verifica la correspondencia de filamentos en el conector RJ45

  Adaptadores de Red
  - Se verifica que físicamente se encuentre instalada correctamente en el zócalo.
  - Se verifica que no tenga ningún tipo de conflicto con su dirección (utilizar el comando ping)

  Hubs
  - Se verifica que este difundiendo correctamente la información (broadcast y multicast)

- **Software de Red**

  Adaptadores de Red
  - Se verifica que los drivers sean los correctos.
  - Se verifica que los protocolos se encuentren instalados correctamente.
  - Se verifica que la configuración de los protocolos sea los correspondientes al tipo de red que utiliza (TCP/IP, NetBios, IPX/SPX, etc).

- **Prueba de ODBC**

Se deben comprobar que el nombre de la base de datos sea el correcto, así como el administrador sea el adecuado.

- Se verifica que el nombre y el path de la base de datos estén correctos
- Se verifica que el administrador de base de datos con el que se va a interactuar sea el correcto
- Se verifica que se ejecuten correctamente las rutinas de los clientes, que abran la base de datos del servidor
- Se prueba si se realiza de manera apropiada la apertura de la base de datos

Si al abrir la base de datos y no poder ejecutar el procedimiento respectivo en el cliente, se debe realizar consultas a la base utilizando los utilitarios propios del administrador de la base de datos para los clientes.

2.8.3. 5. Pruebas del sistema

Verificación de la instalación del software tanto en el cliente como en el servidor

- En el cliente
  Se debe tener muy en cuenta:
  - Se crea automáticamente el directorio en el disco duro
  - Se modifica automáticamente los archivos del sistema tales como Win.ini y el autoexec.bat
  - Se crea automáticamente el icono de la aplicación en la instalación
  - Se copia fácilmente desde el CD-ROM al disco duro

- En el Servidor
  - Se ejecuta por una persona con experiencia y con el password del administrador
  - Se debe seguir el manual de instalación paso a paso
  - Se verifica que no se haya alterado algún software instalado previamente en el servidor
  - Se verifica que se cumpla con cada uno de los requerimientos detallados

Además se puede realizar las siguientes pruebas:

- **Pruebas de Recuperación.** Se fuerza a que el software falle y se verifica su recuperación. La corrección es automática entonces se evalúa los mecanismos de recuperación del estado del sistema. La recuperación requiere la intervención humana, se evalúa los tiempos y si están dentro de los límites.

- **Pruebas de Seguridad.** Se verifica que mecanismo de protección esta incorporado en el sistema, se bloquea el sistema negando así el servicio a otras personas.

- **Pruebas de Rendimiento.** Se prueba el rendimiento del software, tiempo de ejecución. La prueba de rendimiento se da durante todos los pasos de procesamiento de la prueba. Se debe asegurar el rendimiento de los módulos a medida que se llevan acabo las pruebas.

CAPITULO III

3. Introducción

3.1 La empresa

3.1.1 Misión y Visión

El Grupo ORO es un conjunto de empresas avícolas de carácter privado, que nace con el objeto de satisfacer la demanda de productos avícolas (huevos, pollos y pavos) en el país.

Los propietarios de esta empresa, conscientes de la necesidad de contar en nuestro país con una alternativa de productos avícolas y acabar de esa forma con el monopolio existen, crea este grupo el 10 de Octubre de 1992.

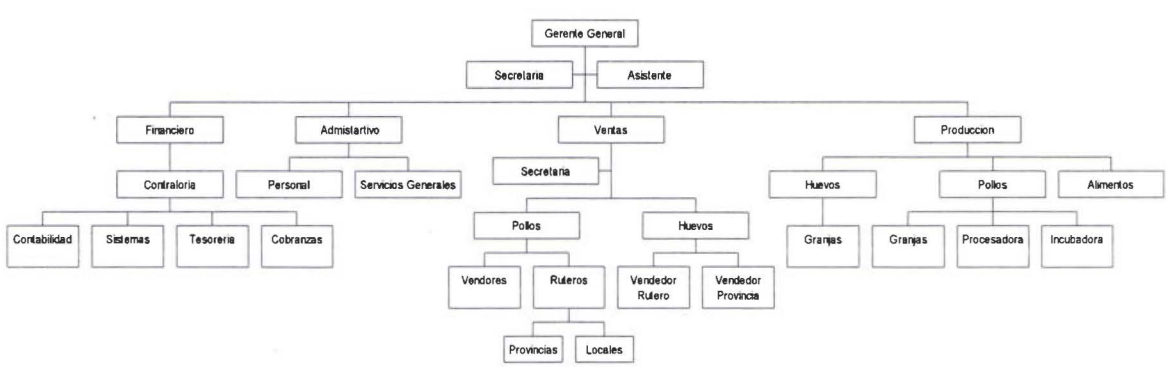
En la actualidad las oficinas administrativas del grupo se encuentran ubicados en la ciudad de Quito en la calle Darquea Terán 16-34 y Av. 10 de Agosto sector de Sta. Clara.

Al momento el grupo esta conformado por las empresas; de huevos, pollos y pavos, planta de alimentos, la planta de embutidos y la planta de fritos. La empresa de huevos a la presente fecha está conformada por tres granjas, una de inicio y desarrollo y dos de producción. La empresa de pollos y pavos esta conformada por 12 granjas, una incubadora y una planta procesadora. La planta de alimentos es la encarga de suministrar todo el alimento que consume en el grupo. La planta de embutidos es la encargada de producir embutidos de distintas presentaciones. La planta de fritos se encarga de la producción de cueritos, papitas y todo lo relacionado con fritos.

Considerándose a las empresas de embutidos y fritos como empresas independientes y no relacionadas con el área avícola.

Su visión es consolidarse como una empresa de reconocido prestigio a nivel nacional, cumpliendo con los estándares de bioseguridad y garantizando a nuestros clientes un producto de alta calidad.

3.1.2 Organigrama



3.1.3 Situación Actual

El presente estudio es para automatizar el control de producción avícola del Grupo Oro, empresa avícola dedicada a la producción de huevos y pollos-pavos, cada una de esta actividades pertenece a una empresa así las aves de engorde(pollos-pavos) a Isabel Almeida, y las ponedoras(huevos) a Iván Larrea empresas encargadas de la producción y comercialización de los productos. El proceso actualmente es totalmente manual en todas sus granjas a excepción de una donde existe una aplicación de control de producción desarrollada en Fox, aplicación que no satisface con algunos requerimientos de los usuarios.

El proceso de producción se inicia con la asignación de un galpón vacío a un lote, en este galpón y lote se ingresará alimento, medicinas y desinfectantes durante toda su estadía en la granja de un lote, el número de aves por lote depende de la edad de las aves y del tamaño del galpón.

Durante el tiempo que las aves están en una granja consumen alimento de diferente tipo de acuerdo a la edad, así también se consumen medicinas y los galpones son desinfectados, toda estas información es llenada en el mismo galpón en un documento denominado de "consumo producción" por el jefe de galpón diariamente en esta documento tiene tres zonas bien definidas donadas, la primera relacionada con la producción de huevos, la segunda el control de aves en el galpón (que novedades se presentaron con las aves), y la tercera relacionada con el consumo de alimento, estos documentos son entregados en las oficinas de administración de la granja.

En la oficina se procede a realizar la totalización de los consumos de alimentos, medicinas y desinfectantes para un lote, se elabora un reporte de consumos por lote, cual fue la producción diaria, cual fue la mortalidad del lote y de la granja.

En las granjas de ponedoras se elabora un gráfico por galpón en el cual se representa la producción diaria vs. los datos del proveedor para esa raza. El gráfico tiene en eje "x" la edad y en el eje "y" el porcentaje de producción. Además se representa cual es el porcentaje consumo de alimento en el lote.

Para realizar el gráficos se aplica las siguientes formulas:

Para la producción = ((hvs. Producidos)*100) /Número de aves vivas

Para alimento = ((kilos consumidos)*100)/Número de aves vivas

Al fin del mes se obtiene un resumen detallado de los consumos y novedades en los lotes, para ser enviados a las oficinas centrales en Quito.

Al dar de baja un lote se obtiene un resumen de todo lo consumido durante su estadía en un galpón.

El procesamiento de datos no es la adecuada debido al volumen de datos y proyección de crecimiento de la empresa por lo que es necesario desarrollar un sistema computarizado para apoyar el registro de datos y control de las granjas y de esta forma tomar a tiempo las decisiones correctivas por parte de la gerencia de producción y la gerencia general.

3.2. Especificación de Requerimientos de Software

3.2.1 Propósito

La especificación de requerimientos del software tiene como propósito describir de manera concreta la funcionalidad y el rendimiento que debe tener el Sistema Control de Producción Avícola (OWL Producción - Granjas) que pueda ser la base para su posterior revisión y aprobación por parte del usuario final. También servirá de guía para los desarrolladores de software.

3.2.2. Ambito

La finalidad del proyecto "OWL Producción - Granjas" es convertirse en una herramienta que servirá de apoyo al proceso de toma de decisiones al contar con información confiable y oportuna que ayudará a los administradores de las granjas, como a la gerencia de producción a tomar los correctivos que sean necesarios y de esta forma evitar gastos innecesarios y realizar una planificación apropiada de producción.

El Sistema de Control de Producción Avícola se aplicará en las granjas del grupo ORO tanto granjas de huevos como en la de pollo-pavos, se compondrá de los siguientes módulos:

- Administración de galpones.
- Consumo y producción
- Movimientos de producto(ingresos, egresos, transferencias)
- Reportes de Movimientos
- Estadísticas de producción.

El almacenamiento de esta información es muy importante ya que con estos datos guardados se puede determinar cual ha sido la producción de cada una de las granjas y dentro de ésta como se ha consumido alimento, medicinas y desinfectantes, cual es la mortalidad. En síntesis cual es la realidad de producción y consumo en cada una de la granja.

3.2.2. Definiciones y abreviaturas

Uni.	Unidades
Kg.	Kilogramos
Fra	Frascos
Lit	Litros
L#####	Número de identificación de un lote, antepuesto la letra "L".

3.2.3. Referencias

Todos los documentos utilizados en los distintos procesos se encuentran en el Anexo A.

- Control de producción y consumo diario
- Movimiento de Huevos
- Consumo de alimento
- Movimiento de alimento mensual
- Movimiento de aves mensual
- Movimiento de huevos mensual
- Liquidación de lote

3.2.4. Descripción general

3.2.4.1. Perspectiva del producto

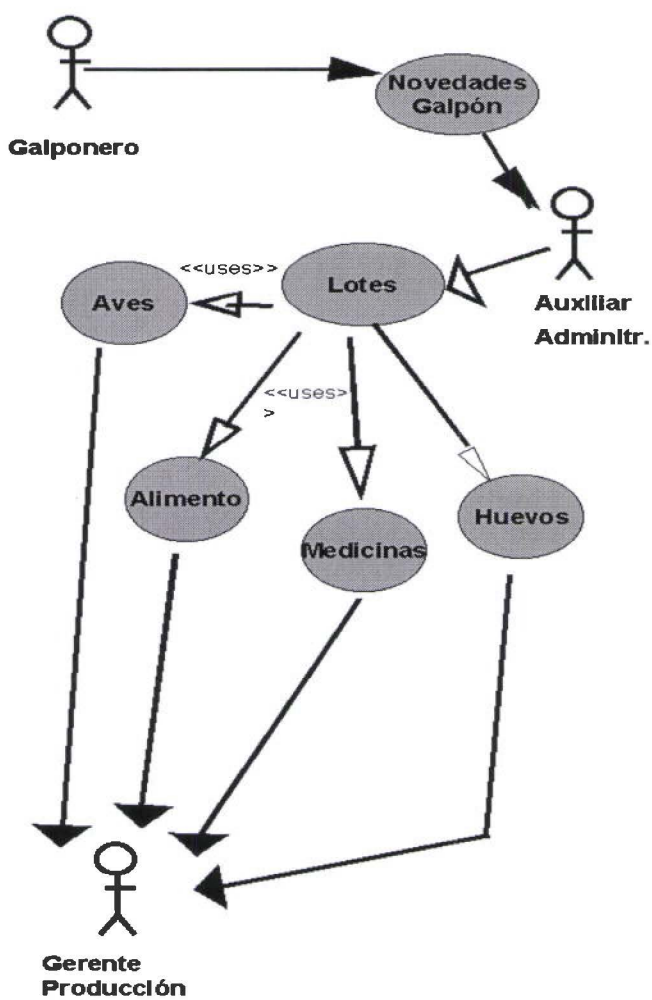
Al no contar con un sistema computarizado, en las granjas de la empresa este proyecto servirá de base para un posterior desarrollo de módulos adicionales que integren la información de la granja.

El sistema es autónomo aunque expandible. El sistema será desarrollado en Microsoft Visual Basic V6.0 como front-end e Informix como back-end en una arquitectura cliente/servidor. La empresa cuenta con las licencias respectivas.

3.2.4.1.1 Diagramas de caso de uso

Los diagrama de caso de uso proporciona información sobre el comportamiento del sistema desde el punto de vista del usuario por medio de acciones y reacciones, determinándose las funciones del producto.

Los componentes del diagrama son: los casos de uso, actores y sus relaciones.



3.2.4.2. Funciones del producto

La función primordial del sistema es brindar un servicio de alta calidad, mejorando de esta forma la administración de una granja y la toma de decisiones muy oportuna por parte de la gerencia de producción y la gerencia general.

En función al diagrama de casos de uso se refleja los principales actores que interactúan en el sistema, este diagrama se tendrá presente a lo largo del desarrollo del sistema

3.2.4.3. Características de los usuarios

Como este sistema se va a implantar en granjas y al no contar con personal calificado y en muchos casos sin experiencia en el manejo de computadores es necesario comenzar con un plan de capacitación.

La información de consumo - producción será ingresada por el auxiliar de administración que existe en cada granja.

Los administradores de granja y el gerente de producción serán los encargados de controlar y modificar la información ingresada, además serán los que más provecho obtengan del sistema al realizar las consultas y reportes que les ayude a tomar las decisiones más oportunas.

Además se requiere que las personas que utilicen el sistema cumplan con:

- Conocimientos básicos del Sistema Operativo Windows
- Contar con cierta experiencia en el manejo de una interfaz gráfica como es el caso de Microsoft Office, con sus diferentes aplicaciones, como son: Word, Excel, etc.
- Conocimiento de las políticas, normas y procedimientos implementados en las granjas y en la empresa.

3.2.4.4. Restricciones generales

El mayor limitante que va a tener el sistema, es que va a manejar cantidades y en ningún instante valores monetarios. Ya que este va a servir de base para obtener los datos para calcular los costos en el modulo de costeo de cada una de las empresas a la cual pertenecen las granjas. Además este sistema va a depender de sistemas de control de inventarios en granjas el cual se lo desarrollara después.

Debido a que es un sistema de datos muy sensibles para la evaluación de la granja se incorporará contraseñas para de los usuarios a realizar ciertas tareas, las mismas que serán administradas por el departamento de sistemas de la empresa.

3.2.4.5. Asunciones y dependencias

Sistema operativo cliente: Microsoft Windows 95/98, Cliente de Informix para Windows
Procesadores en la computadora: Pentium II en adelante
Memoria disponible: 32 MB de preferencia 64
Disco duro: sobre 6 GB de capacidad
Protocolo de red: Protocolo Microsoft TCP/IP
Instalación: En cada una de las granjas.

3.2.5. Requerimientos Específicos

3.2.5.1 Requerimientos funcionales

- ***REQUERIMIENTO FUNCIONAL 1: Creación de Parámetros***

Introducción
Creación de Granjas.- en este requerimiento se pretende registrar las granjas con las cuales se va manejar una empresa.

ENTRADA.- Se ingresará los siguientes datos:
Código de tipo de granja
Descripción de tipo (engorde, ponedoras, inicio-desarrollo)
Número de galpones de la granja

PROCESO.- Se ingresara los datos y se verifica y valida los datos ingresados, no se permitirá blancos o nulos.

SALIDA.- No se obtendrá ninguna salida ya que son datos de para el manejo de la granja.

Introducción
Creación de tabla de producción.- En este requerimiento se pretende registrar los parámetros con los cuales se maneja una raza de aves ponedoras, este dato es tomado de una tabla suministrada por el proveedor de las aves.

ENTRADA.- Se ingresará los siguientes datos:
Raza
Edad
Cantidad

PROCESO.- Se debe seleccionar la raza de las aves. Se digita la edad en semanas no se permite valores negativos. La cantidad es tomada de las tablas y representa el porcentaje de puesta por gallina presente (alojada), no puede ser valores negativos. En ambos caso se aceptara el cero como un valor.

SALIDA.- No se obtendrá ninguna salida ya que estos datos nos servirán de comparación para realizar un gráfico estadístico de la producción diaria de un lote por galpón en la granja de tipo ponedoras.

- ***REQUERIMIENTO FUNCIONAL 2: Granja***

Introducción
Para tener un control de la granja es muy importante registrar su nombre así como la actividad de la granja.

ENTRADA.- Se ingresara los datos que nos permita identificar de manera única a la granja, se ingresará los siguientes datos:
Código de la granja
Nombre de la granja
Tipo de Granja
Número de galpones en la granja

PROCESO.- Una vez ingresado los datos se verifican la validez de cada dato, Deberá tener código nombre, número de galpones, en caso de que un dato no este ingresado se desplegará un mensaje de error.

SALIDA.- Dado de que este es requerimiento que sirve para registra a la granja en la base de datos, el sistema no despliega ninguna salida.

- ***REQUERIMIENTO FUNCIONAL 3: Tipos de Insumos***

Introducción
Es muy importante tener ingresados los insumos (alimentos y medicinas) que se utilizan el galpón

ENTRADA.- Se ingresara los siguientes datos
Tipo de insumo
Código
Nombre(descripción)

PROCESO.- Se procede a seleccionar el tipo de insumo alimento o medicina), su código y su nombre.

SALIDA.- Una vez seleccionada el insumo este desplegara los insumos que se encuentran ingresados permitiendo movernos en ellos.

- ***REQUERIMIENTO FUNCIONAL 4: Asignación Lotes a Galpones***

Introducción
Una vez creado los parámetros de la granja se deberán asignar a los galpones un lote.

ENTRADA.- Se ingresará los siguientes datos:
Granja (código)
Estado del lote
Número del galpón (código)
Número del lote
Raza
Fecha de nacimiento
Fecha de ingreso
Fecha de baja
Estado del galpón

PROCESO.- Se ingresara los datos de identificación de galpón, lote que va a contener y se verifica y valida cada uno de ellos, además se deberá comprobar la disponibilidad de galpón en la granja. Cuando se ingresa un lote a un galpón este se pone en estado de ingreso, cuando termina de ingresar el lote al galpón se debe cambiar su estado a producción después de un periodo de tiempo en este estado será el lote dado de baja.

SALIDA.- Este requerimiento sirve para registrar los datos de un galpón y lote en la base de datos, el sistema no despliega ninguna salida a impresora pero si a pantalla presentando los lotes que se encuentran en el estado seleccionado.

- ***REQUERIMIENTO FUNCIONAL 5: Ingreso Egresos***

Introducción
Este es un requerimiento en el cual se registran las novedades que se presentadas en cada uno de los galpones y por fecha.

ENTRADA.- Se ingresará los siguientes datos:

Granja
Fecha del movimiento
Número del lote
Número (código) del galpón
Aves
Producción
Alimento y Medicinas

PROCESO.- Se seleccionara la granja, la fecha del movimiento, se seleccionara el lote y galpón en el cual vamos a registrar los consumo producción. Dependiendo del tipo de granja que estamos trabajando se ingresara la producción de las aves.

En la parte relacionada con aves nos permitirá escoger las operaciones de ingreso y egreso (donaciones, venta y muerte) de aves.

En la parte relacionada con alimento y medicinas se debe escoger que si el consumo en el galpón es de alimento o de medicina y dentro de estos el producto específico, permitiendo escoger si es un ingreso o un consumo en ese lote galpón.

En la parte relacionada con producción esta tarea se la realizara si la granja que estamos trabajando es del tipo ponedoras. En esta se debe realizar las tareas de Ingresos(producción) y egresos (envíos centro de acopio, lugares específicos).

SALIDA.- Se desplegara los saldos iniciales del galpón para ese fecha, el tipo de movimiento, saldo inicial, cantidad del movimiento, así como se realizara los cálculos para obtener el saldo final fecha a fecha. Si la granja es del tipo engorde se ingresara el peso promedio de las aves.

Además se obtendrá los reportes de Aves, Producción y Alimento - Insumos, de acuerdo aun rango de fechas solicitado.

- ***REQUERIMIENTO FUNCIONAL 6: transferencia de Insumos***

Introducción
Este es un requerimiento que nos permite transferir alimento desde un galpón a otro galpón, y en ocasiones muy esporádicas medicinas o desinfectantes, en una fecha.

ENTRADA.- Se ingresará los siguientes datos:
Granja
Fecha del movimiento
Lote origen
Tipo de operación (Egreso)
Galpón Origen
Lote destino
Galpón destino
Tipo de insumo(Alimento, Medicina)
Producto
Cantidad

PROCESO.- Se selecciona la granja, la fecha del movimiento, seleccionar el lote origen del movimiento, Se debe seleccionar el lote origen y cual es el galpón destino.
Se selecciona el tipo de producto a transferir y producto, y la cantidad del movimiento.

SALIDA.- Se desplegara la fecha del sistema y el tipo de operación que se va ha realizar(egreso por transferencia), todos los datos se guardan. Estos datos afectarán al lote destino cuando en este se registre la novedad. No se podrá realizar

en este requerimiento un ingreso por transferencia puesto que es el resultado de realizar la transferencia al lote destino.

En el reporte Alimento - Insumos se imprimirá las transferencias de aquellos lotes que lo realizaron en la fecha solicitada.

3.2.5.2 Seguridades.

El sistema manejará la seguridad basada en la definición de perfiles de usuario de acuerdo a las funciones asignadas.

Los usuarios que van a usar el sistema contarán con una única identificación de usuario y un password. A cada cuenta de usuario se le será asignado un conjunto de privilegios o permisos para utilizar los diferentes objetos, para desempeñar adecuadamente sus tareas.

Los usuarios son:

- Auxiliar contable encargada de:
  - Ingresar los datos de novedades
  - Generar reportes de producción

El auxiliar es el único que podrá ingresar a todos los módulos del sistema.

- El Administrador de la granja encargado de:
  - Modificar errores de digitación
  - Modificar los parámetros de la granja
  - Ingresar y modificar los parámetros de producción

- El Gerente de producción
  - Consultar acumulados de producción

El gerente de producción podrá además consultar todos los datos

3.2.5.3 Requerimientos de Interfaces externos

Se desplegará todos los módulos a través de pantalla con una resolución de 800 x 600 pixels y colores 16-bits, siempre se debe mantener la resolución antes indicadas pues la interface ocupará toda la pantalla del monitor.

Los error serán cortos y explícitos, la conexión entre los computadores se la realizará mediante el protocolo TCP/IP de Microsoft el mismo que se lo configurara desde el sistema operativo.

3.2.5.4 Requerimientos de Software

Los equipos que se destinarán para el desarrollo deberán tener las siguientes herramientas.
Rational Rose 98
Power Builder 6.5
Informix
Visual Basic 6.0
Cristal Report 4.2

Conclusiones y Recomendaciones

- OMT demostró ser una herramienta muy útil dentro de la tecnología orientada a objetos. Su colección de diagramas, tiene cualidades para modelar diferentes aspectos del sistema, los mismos que interactúan entre sí a través de los modelos. El modelo de objetos representa la estructura estática del sistema, el modelo dinámico es la estructura de control del sistema y el modelo funcional describe los cálculos existentes dentro del sistema. Los distintos problemas hacen diferentes grados de hincapié en los tres modelos, pero los tres son complementarios y necesarios en todo desarrollo de sistemas.

- En un desarrollo de software con OMT se puede suprimir algunos de los diagramas y extensiones cuando estos no son necesarios.

- Se utilizó la tecnología orientada a objetos para el desarrollo del sistema por se una metodología que permite desarrollar sistemas independientes del lenguaje final de implementación y en el menor tiempo, al utilizar representaciones gráficas para cada uno de los requerimientos de los usuarios y el sistema en general.

- Al tener el informix su propio 4GL para generar código fuente no se utilizo para el desarrollo del sistema por utilizar como interface la clásica pantalla de Unix y no la interface gráfica que hoy en día tiene mayor aceptación y comprensión por parte de los usuarios.

- Se debe considerar la transportabilidad que se va a tener al usar un lenguaje de programación como Visual Basic. Los costos que representan la capacitación y licencias del 4GL en comparación con los de Visual Basic.

- El modelo de caso de uso en el trabajo práctico refleja el funcionamiento general del sistema, logrando de esta forma dar un entendimiento fácil para los usuarios que no tienen conocimiento técnico.

- En él capitulo de pruebas de la arquitectura cliente/servidor no se tomo en cuenta todas las pruebas existentes, sino solo aquellas a las cuales se sometió al software.

- Las recomendaciones para un cliente informix es de windows 95/98, 32MB en RAM y un procesador Pentium, sin embargo para que su rendimiento sea optimo se sugiere utilizar en equipos con mínimo de 64MB en RAM.

- Para que exista una adecuada visualización de las ventanas del software se sugiere configurar el monitor en el área de la pantalla con 800 por 600 píxeles y colores de alta densidad (16 bits).

- Al diseñar las interfaces de usuarios se considero que estas sean lo más simples posibles, los controles deben tener una fuente de letra grande y deben estar cargadas con la menor cantidad de controles.

Bibliografía

ROGER PRESSMAN, Ingeniería de Software un enfoque práctico Cuarta Edición
Mc GrawÄHill, 1996.

JAMES MARTIN,JAME J. ODELL, Análisis y Diseño Orientado a Objetos
Prentice Hall, 1992.

JAMES RUMBAUGH, Modelo y Diseño Orientado a Objetos
Prentice Hall, 1997.

BOOCH G. Análisis y Diseño Orientado a Objetos
Addson-Wesley,1996

BERSON, A. Cliente/Servidor
McGrawAHill, 1992.

RUALESS B. Juan Carlos, YANEZ U. Geoconda Cliente Servidor Guía de pruebas
Tesis EPN, Octubre 2000

ORFALIi Robert, HARKEY Dan, Edwards Cliente Servidor Guía de supervivencia
McGrawÄHill, 1998.

OMT, http://www.monografias.com/omt

Cliente/Servidor, http://www.rspa.com

Cliente/Servidor, http: //www.omg.org

RATIONAL ROSE9 8, http: //www.rational.com/support/documentation/manuals

ANEXO A

DOCUMENTOS BASICOS

IVAN LARREA
Avicultor
MERAPEC

72001 / 74 100 x 2 - IV - 2001

CONTROL DE PRODUCCION Y CONSUMO DIARIO

Lote# 36-00A

Galpón#: 1

Fecha: 28 (DIA) 06 (MES) 2001 (AÑO)

DETALLE		Unidades
Huevos:		
SALDO ANTERIOR		47060.
Producción: Sanos		11370.
Transferencias a Bodega		60.
Saldo Final		58370
Aves:		
SALDO ANTERIOR		14069.
Ingresos		
Muertas		6
Vendidas		
Saldo Final		14063
Consumo de Alimento balanceado en Kilos:		
SALDO ANTERIOR		10815.
Ingreso:	Ponedoras 1	
	Ponedoras 2	
	Ponedoras 3	
	Prepostura	
Consumo:	Ponedoras 1	
	Ponedoras 2	1530.
	Ponedoras 3	
	Prepostura	
Transferencias		
Saldo Final		9285

Observaciones: .....

Distribución:	
1 Verde:	Contabilidad
2 Rosada:	Oficina Granja

Gráficas Guevara Telf.: 641-417

Elaborado por

Revisado por

Jefe de Granja

Asistente de Oficina

GRUPO ORO

Isabel Almeida
AVICULTOR
"DIVISION POLLOS"

CONTROL PARA POLLOS DE CARNE

GAVILAN

LOTE No. 1 2
Galponeo AURELIO SANCHEZ

Fecha Ingreso 04 MAYO 2001
Procedencia ..... Raza .....
Cantidad 14530 Valor .....

Semana	Fecha	Alimentos				Medicinas y Vacunas	Causas Mortalidad			Mortalidad				OBSERVACIONES
		Clase	Diario	Acum.	Saldo					Diario	Acum.	Saldo	%	
1	4	50 c	2	2	48	VITAESENCC				5	5	14525		35 gr
	5		3	5	45	VITAESENCC				15	20	14510		
	6		3	8	42	PANELA				15	35	14495		Cons: 14 gr
	7		5	13	37					7	42	14488		CV = 0.32
	8		6	19	31					5	47	14483		Cons = 14 gr
	9		7	26	24	Oh Angel				10	57	14473		
	10	50 c	7	33	64					8	65	14465	0.45	125 gr
2	11	120 c	8	41	39	al G3				10	75	14455		
	12		9	50	30					15	90	14440		
	13		8	58	22					5	95	14435		Cons: 33 gr
	14		9	67	13	Oh Angel				7	102	14428	0.38	CV = 1.21
	15	30 c	12	79	31					5	107	14423		Cons = 22 gr
	16	100 c	15	94	116					8	115	14415		
	17	40 c	15	109	141					5	120	14410	0.83	281 gr
3	18		16	125	125	VERMITIS				5	125	14405		
	19		17	142	108	GUMBORO				3	128	14402		
	20		19	161	89	VENCOBOTAE				7	135	14395		Cons: 64 gr
	21		20	181	69					5	140	14390	0.21	CV = 1.51
	22	120 c	22	203	167					7	147	14383		Cons = 35 gr
	23		24	227	143	FORMOL				8	155	14375		
	24	57 c	27	254	173	Oh Angel				5	160	14370	1.10	526 gr
4	25		25	279	148					7	167	14363		
	26		26	305	122					8	175	14355		
	27		27	332	95					10	185	14345		Cons: 65 gr
	28	100 c	28	360	167	NEWCASTLE				15	200	14330	0.48	CV = 1.49
	29		28	388	139					7	207	14323		Cons = 63 gr
	30		28	416	111	Oh Angel				16	223	14307		
	31	180 c	29	445	262	YODO				7	230	14300	1.58	361 gr
5	1		31	476	231	YODO				8	238	14292		
	2		33	509	196					8	246	14284		Cons = 71 gr
	3		35	544	163					4	250	14280		Cons = 174 gr
	4		37	581	126	TYLOCOR				13	263	14267		CV = 1.64
	5	120 c	39	620	207	TYLOCOR				11	274	14256		Cons = 54 gr
	6	100 c	42	662	265	TYLOCOR				13	287	14243		
	7	100 c	43	705	322	Oh Angel				13	297	14233	2.04	1362 gr

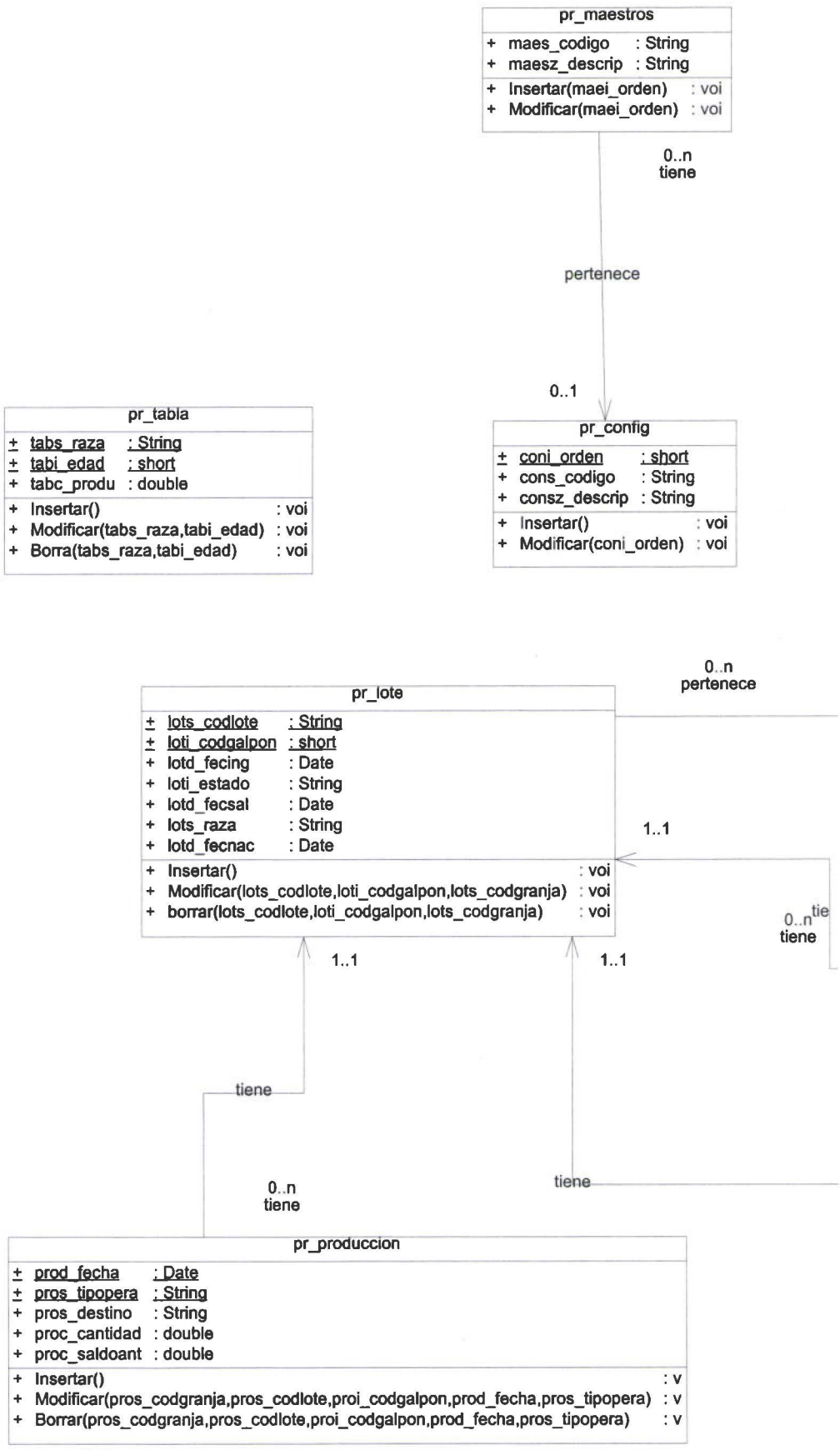
Kilos alimento consumido .....
Kilos carne producida .....
Peso promedio del pollo .....
"Conversión" .....
Mortalidad % .....
Factor Eficiencia .....

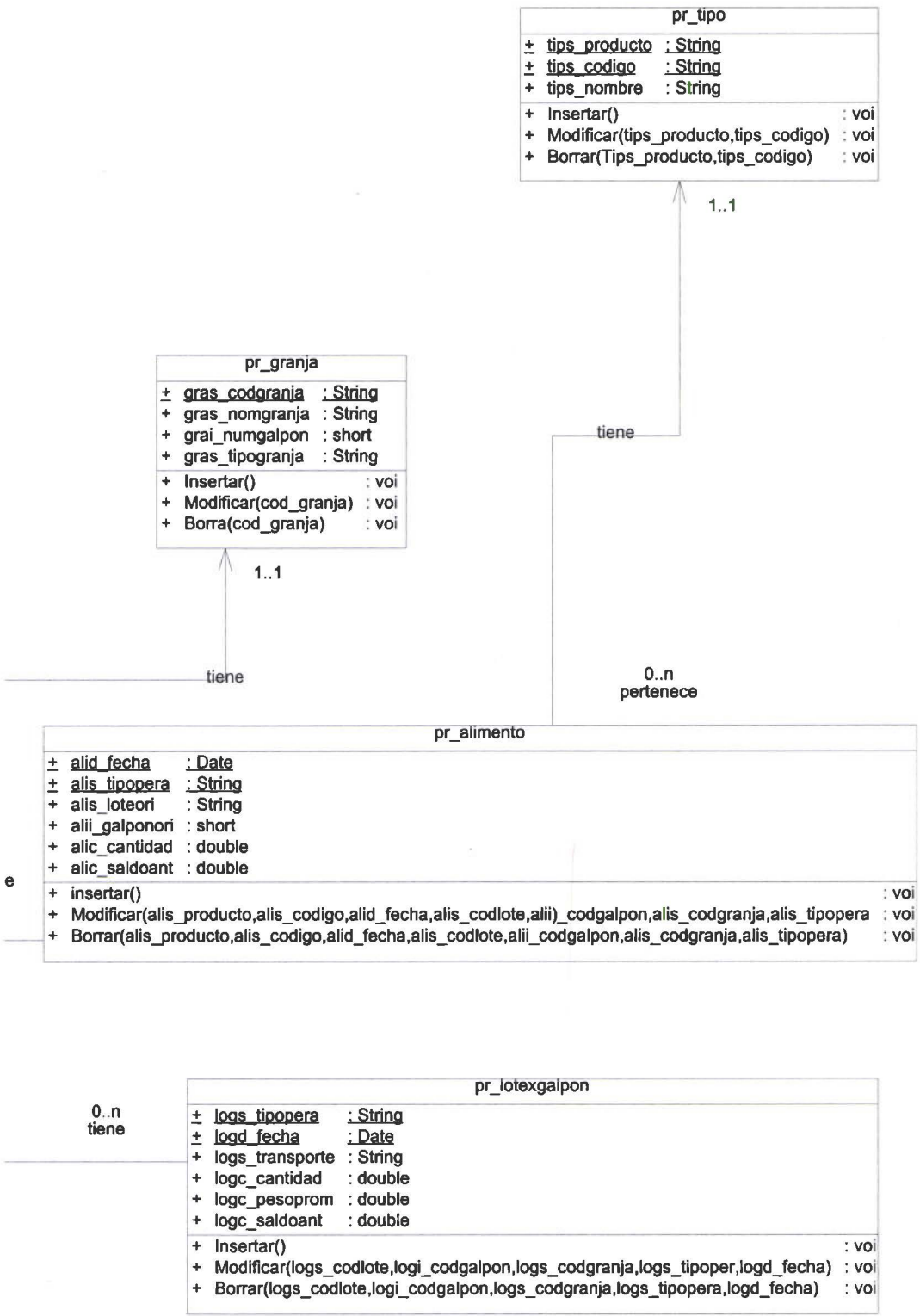
Valor venta de pollos S/. .....
SE DEDUCE:
Costo de los pollitos S/. .....
Costo del alimento consumido S/. .....
Costo vacunas y medicinas S/. .....
Costo mano de obra y servicios S/. .....
Gastos acondicionamiento y otros S/. .....
Utilidad obtenida S/. .....
Utilidad por pollo S/. .....

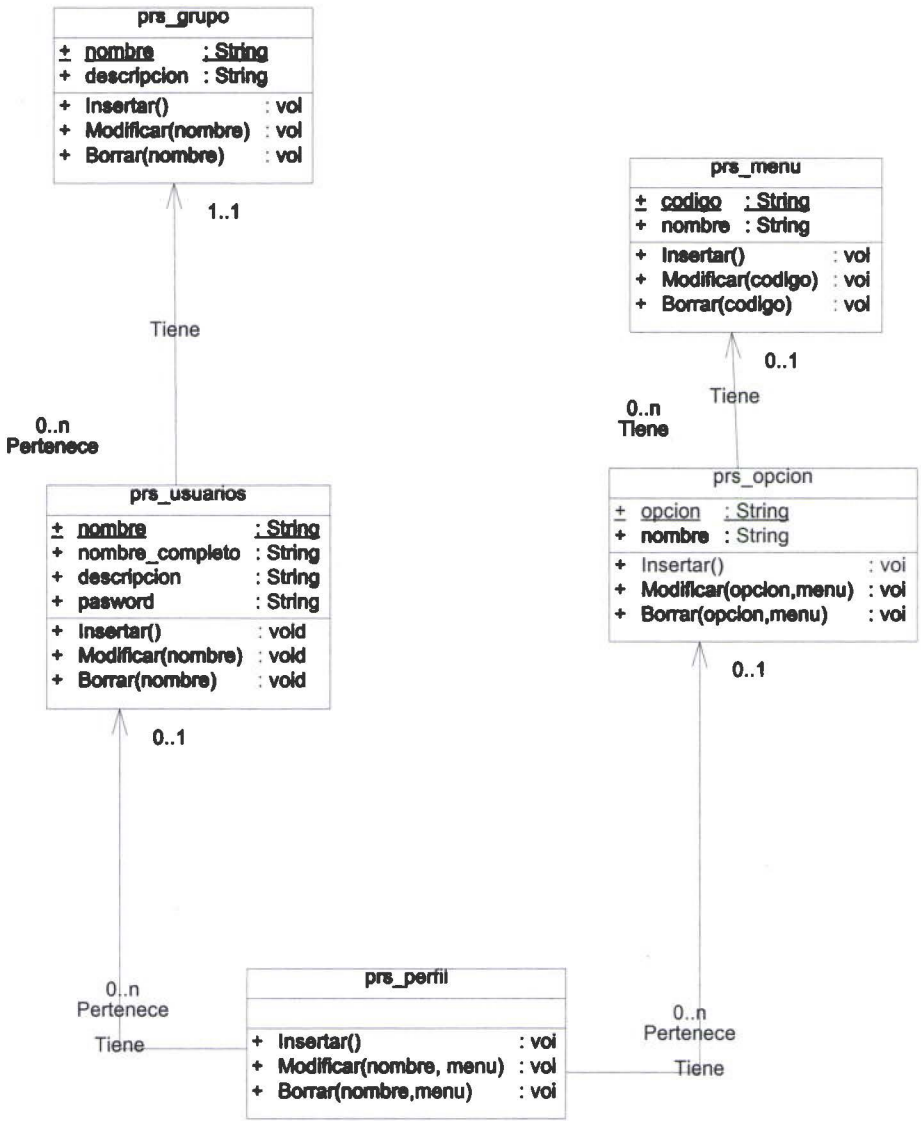
APLIGRAF . junio 2000. 1000 ejps

ANEXO B

DIAGRAMA DE MODELO DE OBJETOS


pr_maestros
+ maes_codigo : String
+ maesz_descrip : String
+ Insertar(maei_orden) : voi
+ Modificar(maei_orden) : voi
0..n
tiene
pertenece
0..1
pr_tabla
+ tabs_raza : String
+ tabi_edad : short
+ tabc_produ : double
+ Insertar() : voi
+ Modificar(tabs_raza,tabi_edad) : voi
+ Borra(tabs_raza,tabi_edad) : voi
pr_config
+ coni_orden : short
+ cons_codigo : String
+ consz_descrip : String
+ Insertar() : voi
+ Modificar(coni_orden) : voi
0..n
pertenece
pr_lote
+ lots_codlote : String
+ loti_codgalpon : short
+ lotd_fecing : Date
+ loti_estado : String
+ lotd_fecsal : Date
+ lots_raza : String
+ lotd_fecnac : Date
+ Insertar() : voi
+ Modificar(lots_codlote,loti_codgalpon,lots_codgranja) : voi
+ borrar(lots_codlote,loti_codgalpon,lots_codgranja) : voi
1..1
0..n
tiene
1..1
1..1
tiene
0..n
tiene
tiene
pr_produccion
+ prod_fecha : Date
+ pros_tipopera : String
+ pros_destino : String
+ proc_cantidad : double
+ proc_saldoant : double
+ Insertar() : v
+ Modificar(pros_codgranja,pros_codlote,proi_codgalpon,prod_fecha,pros_tipopera) : v
+ Borrar(pros_codgranja,pros_codlote,proi_codgalpon,prod_fecha,pros_tipopera) : v


pr_tipo
+ tips_producto : String
+ tips_codigo : String
+ tips_nombre : String
+ Insertar() : voi
+ Modificar(tips_producto,tips_codigo) : voi
+ Borrar(Tips_producto,tips_codigo) : voi
1..1
tiene
pr_granja
+ gras_codgranja : String
+ gras_nomgranja : String
+ grai_numgalpon : short
+ gras_tipogranja : String
+ Insertar() : voi
+ Modificar(cod_granja) : voi
+ Borra(cod_granja) : voi
1..1
tiene
0..n
pertenece
pr_alimento
+ alid_fecha : Date
+ alis_tipopera : String
+ alis_loteori : String
+ alii_galponori : short
+ alic_cantidad : double
+ alic_saldoant : double
+ insertar() : voi
+ Modificar(alis_producto,alis_codigo,alid_fecha,alis_codlote,alii)_codgalpon,alis_codgranja,alis_tipopera : voi
+ Borrar(alis_producto,alis_codigo,alid_fecha,alis_codlote,alii_codgalpon,alis_codgranja,alis_tipopera) : voi
e
0..n
tiene
pr_lotexgalpon
+ logs_tipopera : String
+ logd_fecha : Date
+ logs_transporte : String
+ logc_cantidad : double
+ logc_pesoprom : double
+ logc_saldoant : double
+ Insertar() : voi
+ Modificar(logs_codlote,logi_codgalpon,logs_codgranja,logs_tipoper,logd_fecha) : voi
+ Borrar(logs_codlote,logi_codgalpon,logs_codgranja,logs_tipopera,logd_fecha) : voi


prs_grupo
± nombre : String
+ descripcion : String
+ Insertar() : vol
+ Modificar(nombre) : vol
+ Borrar(nombre) : vol
1..1
Tiene
0..n
Pertenece
prs_usuarios
± nombre : String
+ nombre_completo : String
+ descripcion : String
+ pasword : String
+ Insertar() : void
+ Modificar(nombre) : void
+ Borrar(nombre) : void
0..1
prs_menu
± codigo : String
+ nombre : String
+ Insertar() : vol
+ Modificar(codigo) : voi
+ Borrar(codigo) : vol
0..1
Tiene
0..n
Tiene
prs_opcion
± opcion : String
+ nombre : String
+ Insertar() : voi
+ Modificar(opcion,menu) : voi
+ Borrar(opcion,menu) : voi
0..1
0..n
Pertenece
Tiene
prs_perfil
+ Insertar() : voi
+ Modificar(nombre, menu) : voi
+ Borrar(nombre,menu) : voi
0..n
Pertenece
Tiene

ANEXO C

DICCIONARIO DE DATOS

Listado de clases

Nombre	Tipo de Clase
Pr_maestro	Clase
Pr_config	Clase
Pr_granja	Clase
Pr_lotexgalpon	Clase
Pr_lote	Clase
Pr_tipo	Clase
Pr_alimimento	Clase
Pr_produccion	Clase
Pr_tabla	Clase

Asociaciones

Nombre Asociaciones	Clase A	Clase B	Roles B	Multiplicidad A	Multiplicidad B
Pertenece	Pr_config	Pr_maestro	Tiene	0..1	0..n
Tiene	Pr_tipo	Pr_alimento	Pertenece	1..1	0..n
Tiene	Pr_lote	Pr_alimento	Tiene	1..1	0..n
Tiene	Pr_lote	Pr_lotexgalpon	Tiene	0..1	0..n
Tiene	Pr_lote	Pr_produccion	Tiene	0..1	0..n
Tiene	Pr_granja	Pr_lote	Pertenece	1..1	0..n

Información de las clases

Clase: pr_granja

Atributos

Nombre	Tipo de dato	Descripción
Gras_codgranja	Char(2)	Código de la granja
Gras_nomgranja	Varchar(20)	Nombre de la granja
Grai_numgalpon	Smallint(20)	Número de galpones en la granja
Gras_tipogranja	Varcgar(25)	Tipo de granja a la que pertenece la granja EN=engorde, PR=producción

Clase: pr_lotexgalpon

Atributos

Nombre	Tipo de dato	Descripción
Logs_codlote	Char(20)	Código del lote
Logi_codgalpon	Char(2)	Número de galpón
Logd_fecha	Date	Fecha del movimiento
Logi_codgranja	Chra(2)	Código de granja
Loga_tipopera	Varchar(25)	Tipo de opración (ING=ingreso, EVE=venta, EMU=muetras,EDO=donadas)
Logc_cantidad	Money(6,4)	Cantidad del movimiento
Logc_pesoprom	Money(6,4)	Peso promedio del lote por galpón
Logc_saldoant	Money (6,4)	Saldo anterior al movimiento
Logs_transporte	Varchar(25)	Código de transporte

Clase: pr_lote

Atributos

Nombre	Tipo de dato	Descripción
Lots_codlote	Char(20)	Código del lote
Lots_raza	Varchar(25)	Código de raza
Loti_codgalpon	Smallint	Número de galpón
Lots_codgranja	Char(2)	Código de granja
Lotd_fecnac	Date	Fecha de nacimiento
Lotd_fecing	Date	Fecha de ingreso del lote
Lo_estado	Smallint	Estados posibles en los que se encuentra un lote (IN=ingreso, PR=producción, BA=baja,)
Lotd_fecsal	Date	Fecha salida de un lote

Clase: pr_tabla

Atributos

Nombre	Tipo de dato	Descripción
Tabs_raza	Varchar(25)	Código de la raza
Tabi_edad	Smallint	Edad de las aves
Tabc_produ	Money(6,4)	Producción promedia por ave

Clase: pr_tipo

Atributos

Nombre	Tipo de dato	Descripción
Tips_producto	Varchar(25)	Identificador de producto
Tips_codigo	Char(5)	Código producto
Tips_nombre	Varchar(20)	Identificación del producto
Tips_unidad	Varchar(20)	Código unidad de medida

Clase: pr_alimento

Atributos

Nombre	Tipo de dato	Descripción
Alis_producto	Varchar(25)	Identificador de producto
Alis_codigo	Char(5)	Código producto
Alid_fecha	Date	Fecha del movimiento
Alis_codlote	Char(20)	Código de lote
Alii_codgalpon	Smallint	Número de galpón
Alis_codgranja	Char(2)	Código de granja
Alis_tipopera	Varchar(25)	Tipo de operación (ING=ingreso, ITR=Ingreso transferencia, ECO=consumo, ETR=Egreso transferencia)
Alii_cantidad	Money(6,4)	Cantidad del movimiento
Alic_saldoante	Money(6,4)	Saldo anterior al movimiento
Alis_loteori	Char(5)	Lote origen de la operación
Alii_galponori	Smallint	Galpón origen de la operación

Clase: pr_produccion

Atributos

Nombre	Tipo de dato	Descripción
Pros_codlote	Char(20)	Código de lote
Proi_codgalpon	Smallint	Código de galpón
Pros_codgranja	Char(2)	Código de granja
Prod_fecha	Date	Fecha del movimiento
Pros_tipopera	Varchar(25)	Tipo de operación (ING=producción(ingreso), ESA=salidas(egreso))
Proc_cantidad	Money(6,4)	Cantidad del movimiento
Proc_saldoant	Money(6,4)	Saldo anterior al movimiento
Pros_destino	Varchar(25)	Código del destino

Clase: pr_maestro

Atributos

Nombre	Tipo de dato	Descripción
Maei_orden	Smallint	Número de identificación
Maes_codigo	Varchar(25)	Código
Maesz_descrip	Varchar(50)	Descripción

Clase: pr_config

Atributos

Nombre	Tipo de dato	Descripción
Coni_orden	Smallint	Número de identificación
Cons_codigo	Varchar(25)	Construcción del código
Consz_descrip	Varchar(50)	Descripción

- El campo coni_orden y maei_orden tiene los siguientes valores:

Estado lotes de galpón	1
Tipos de insumos	2
Razas aves	3
Tipos de granja	4
Movimientos de aves	5
Movimientos producción	6
Movimientos alimentos	7
Destinos de producción	8
Movimientos transferencia	9
Unidades de Medida	10

Métodos de las clases

Método	Tipo de retorno	Clase
Insertar()	Void	Pr_tipo
Modificar(tips_producto,tips_codigo)	Void	Pr_tipo
Borrar(tips_producto,tips_codigo)	Void	Pr_tipo
Insertar(maei_orden)	Void	Pr_maestro
Modificar(maei_orden,maes_codigo)	Void	Pr_maestro
Insertar()	Void	Pr_config
Modificar(coni_orden)	Void	Pr_config
Insertar()	Void	Pr_tabla
Modificar(tabs_raza,tabi_edad)	Void	Pr_tabla
Borrar(tabs_raza,tabi_edad)	Void	Pr_tabla
Insertar()	Void	Pr_granja
Modificar(gras_codgranja)	Void	Pr_granja
Borrar(gras_codgranja)	Void	Pr_granja
Insertar(lots_codgranja)	Void	Pr_lote
Modificar(lots_codgranja,lots_codlote,loti_codgalpon)	Void	Pr_lote
Borrar(lots_codgranja,lots_codlote,loti_codgalpon)	Void	Pr_lote
Insertar()	Void	Pr_alimento
Modificar(alis_codgranja,alis_producto,alis_codigo.alid_fecha)	Void	Pr_alimento
Borrar(alis_codgranja,alis_producto,alis_codigo.alid_fecha)	Void	Pr_alimento
Insertar()	Void	Pr_lotexgalpon
Modificar(logs_cdogranja,lods_codlote,logi_codgalpon,logd_fecha)	Void	Pr_lotexgalpon
Borrar(logs_cdogranja,lods_codlote,logi_codgalpon,logd_fecha)	Void	Pr_lotexgalpon
Insertar()	Void	Pr_produccion
Modificar(pros_codgranja,pros_codlote,proi_codgalpon,prod_fecha)	Void	Pr_produccion
Borrar(pros_codgranja,pros_codlote,proi_codgalpon,prod_fecha)	Void	Pr_produccion

ANEXO D

DIAGRAMA DE SUCESOS

Diagrama de Secuencia: Aves

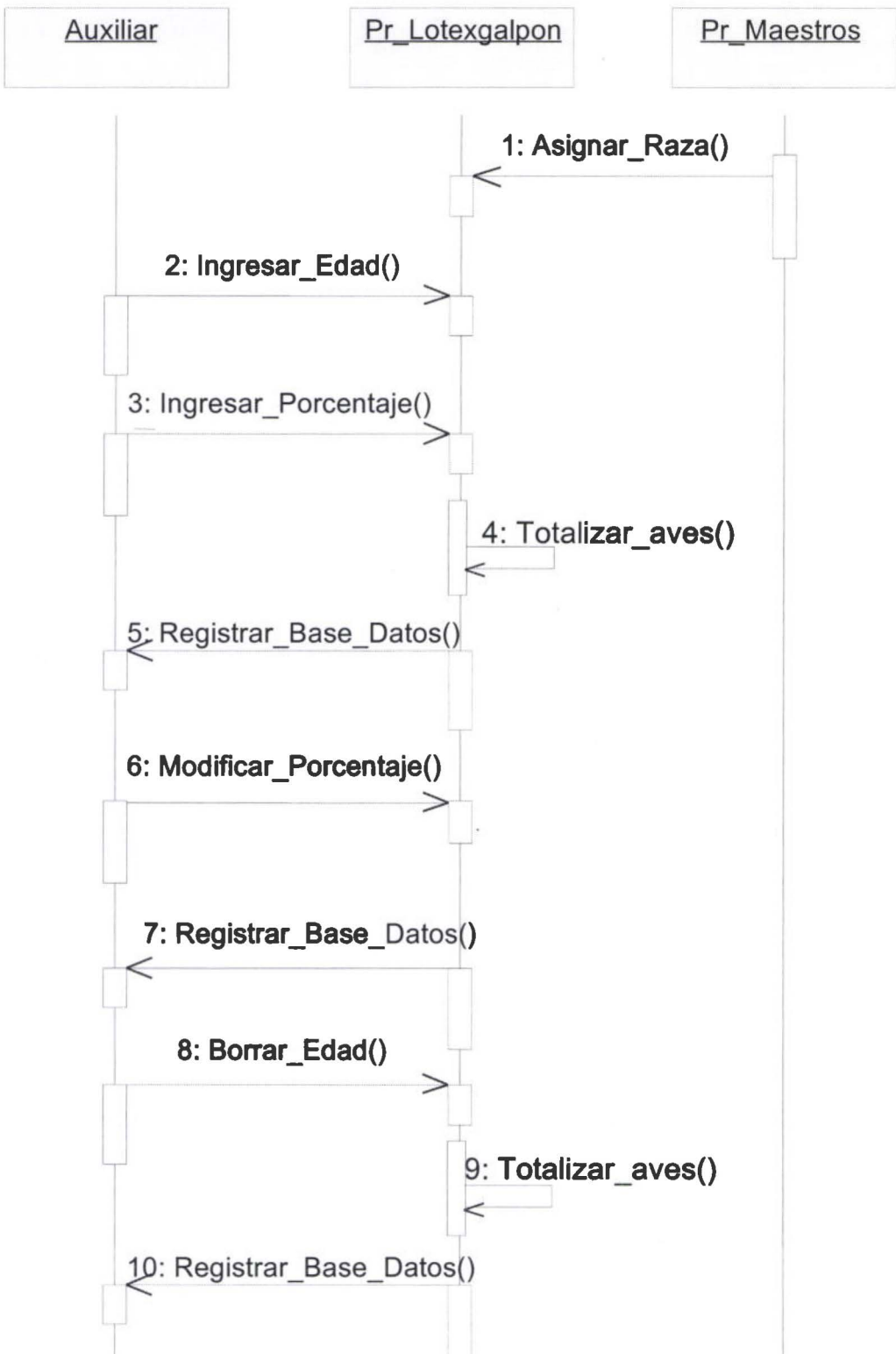


Diagrama de Secuencia: Huevos

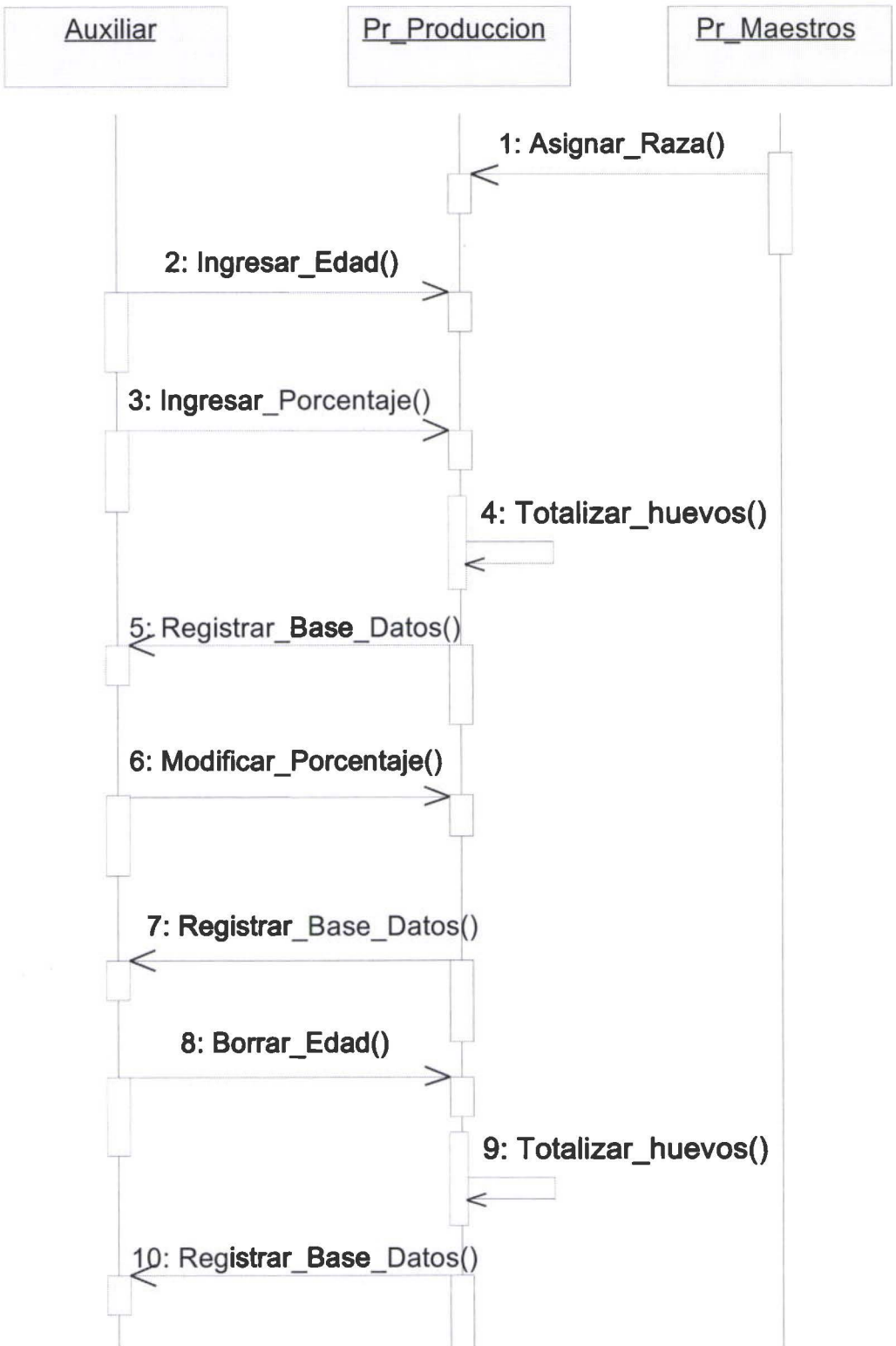


Diagrama de Secuencia: Alimentos y Medicinas

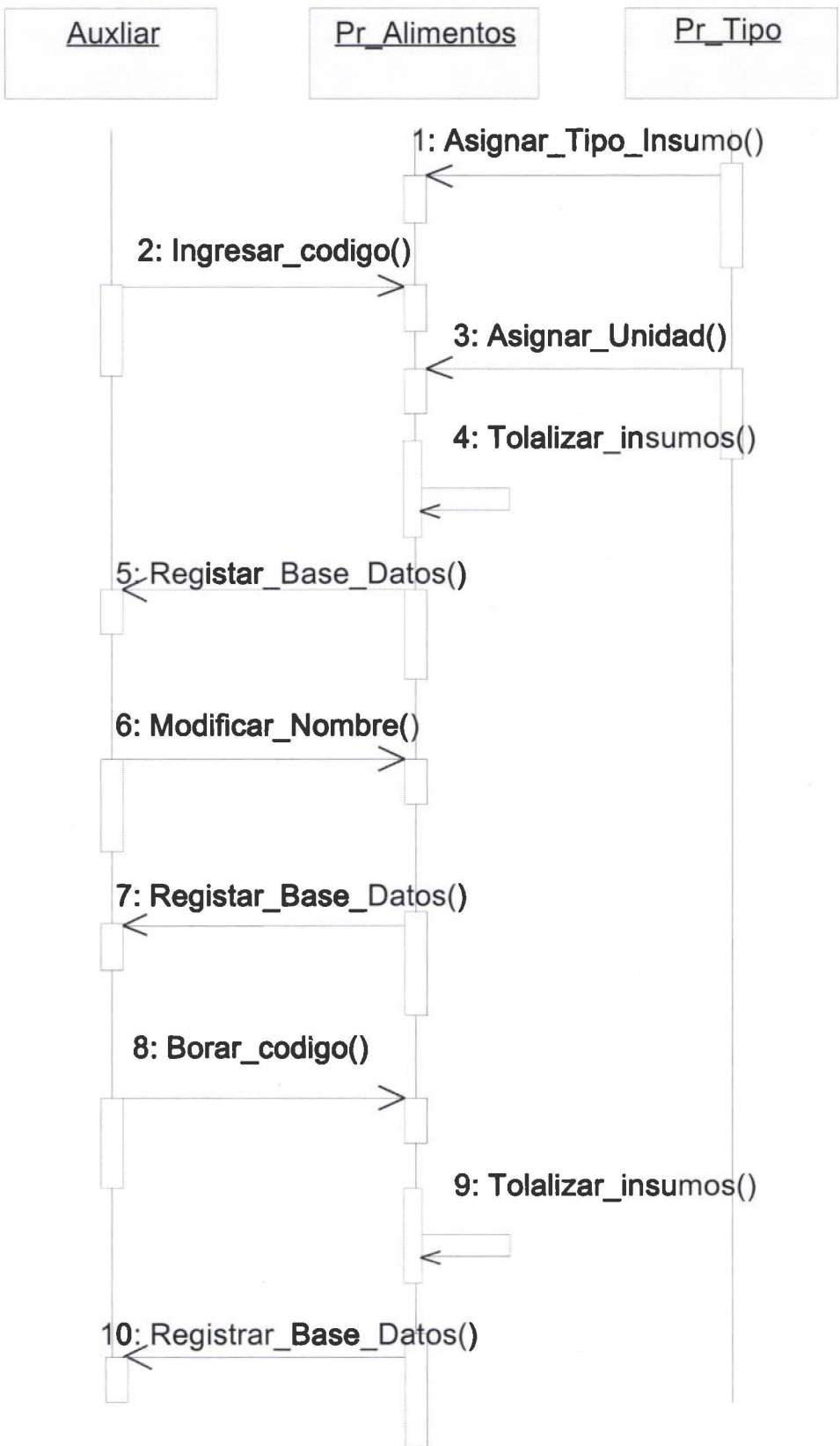
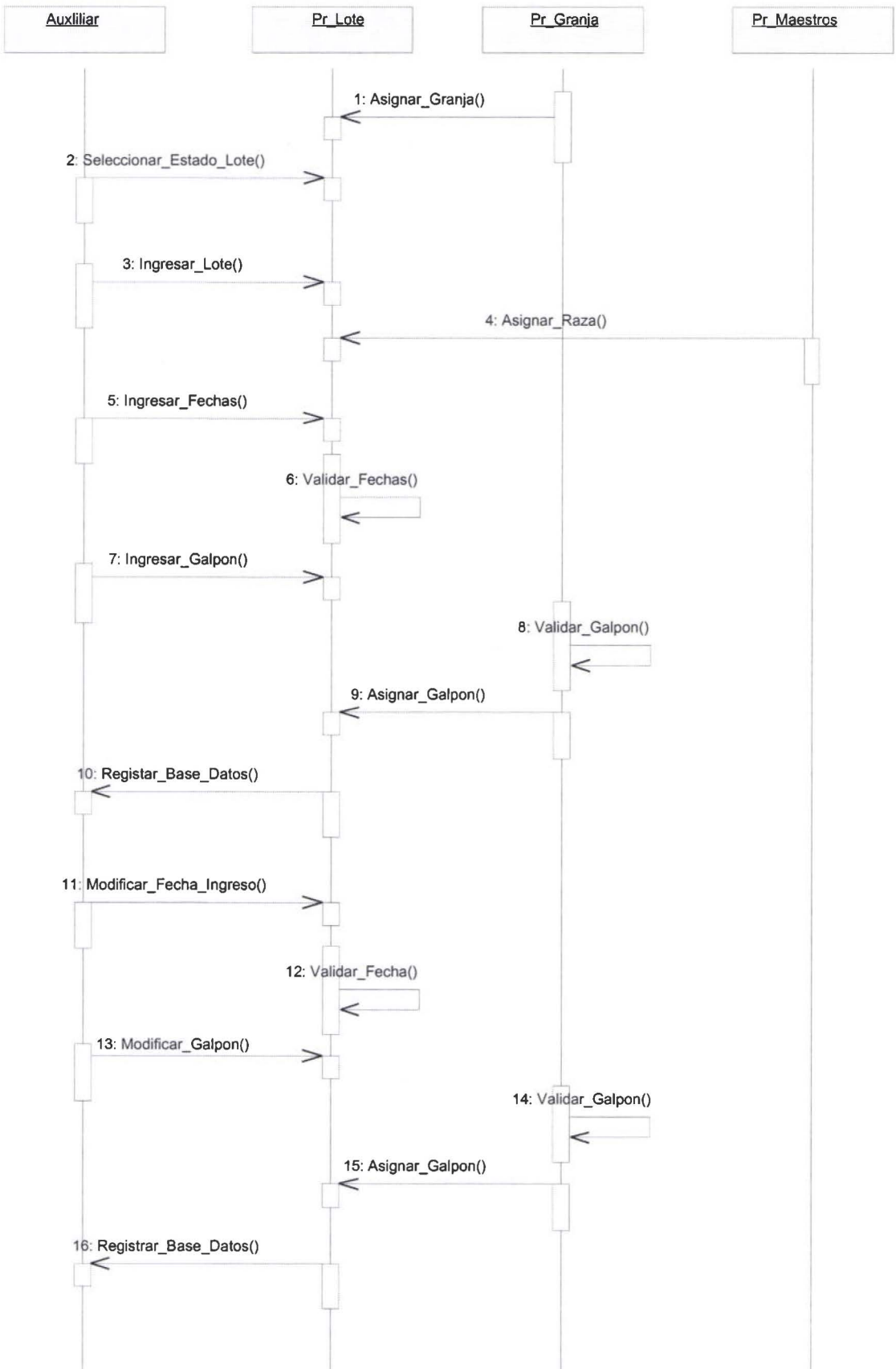


Diagrama de Secuencia: Lotes



ANEXO E

DIAGRAMA DE ESTADOS

Diagrama de Estados: Lote

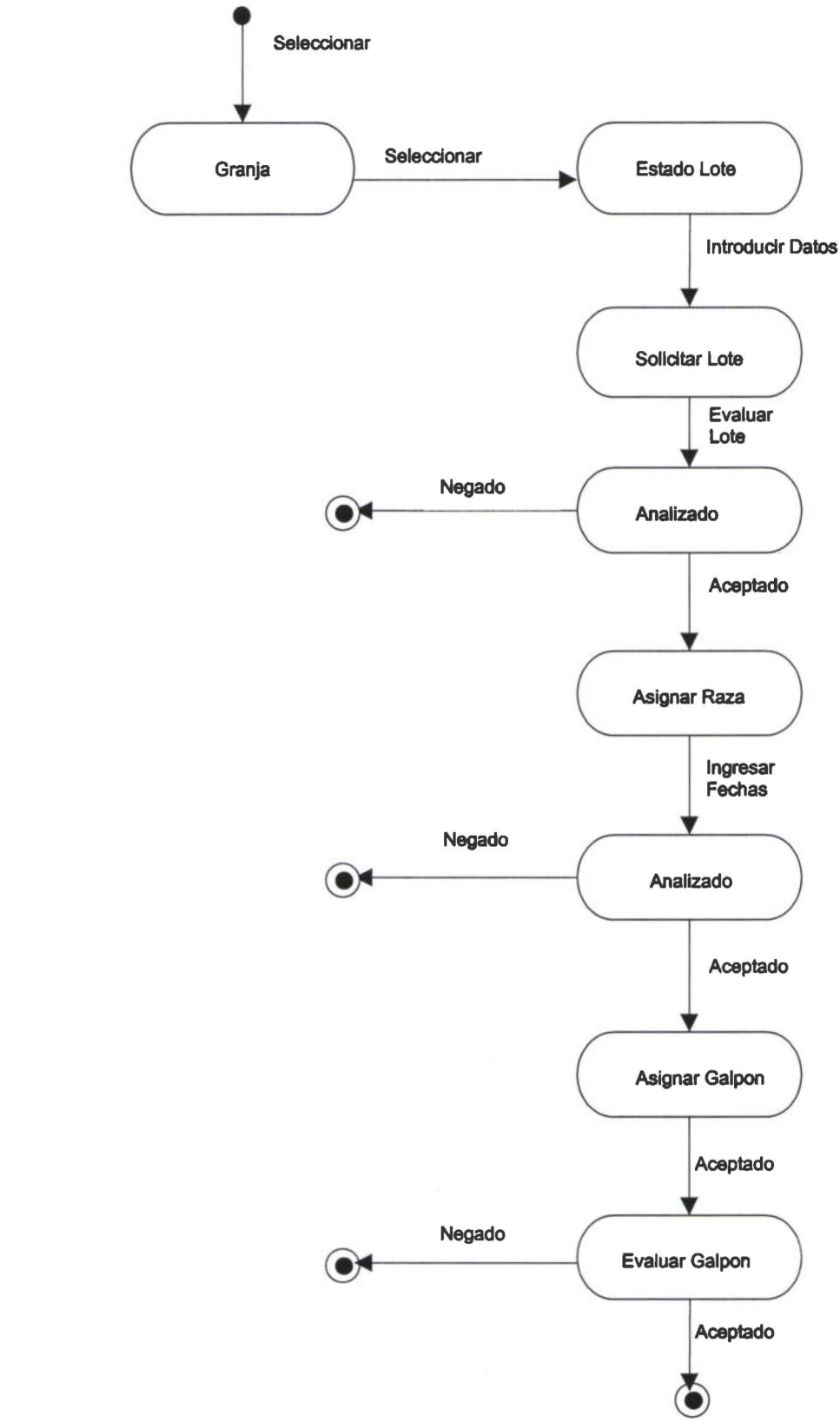


Diagrama de Estados: Insumos (Medicinas-Desinfectantes)

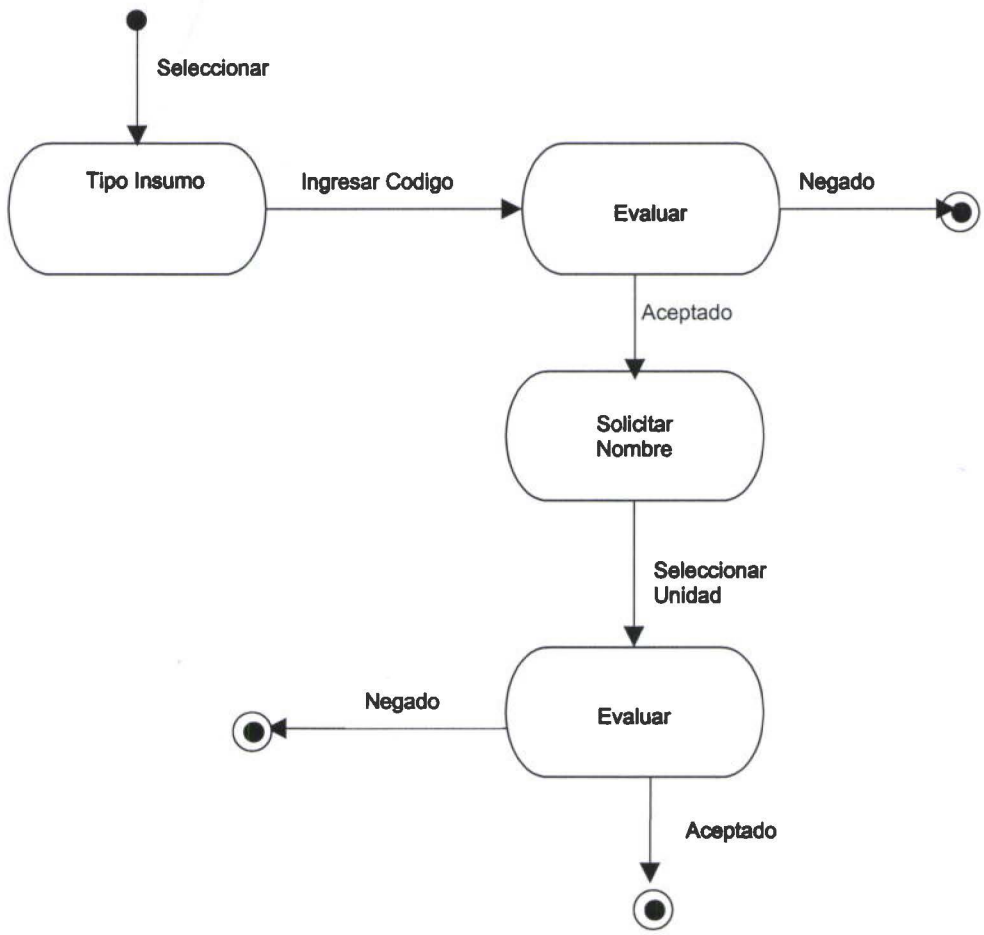
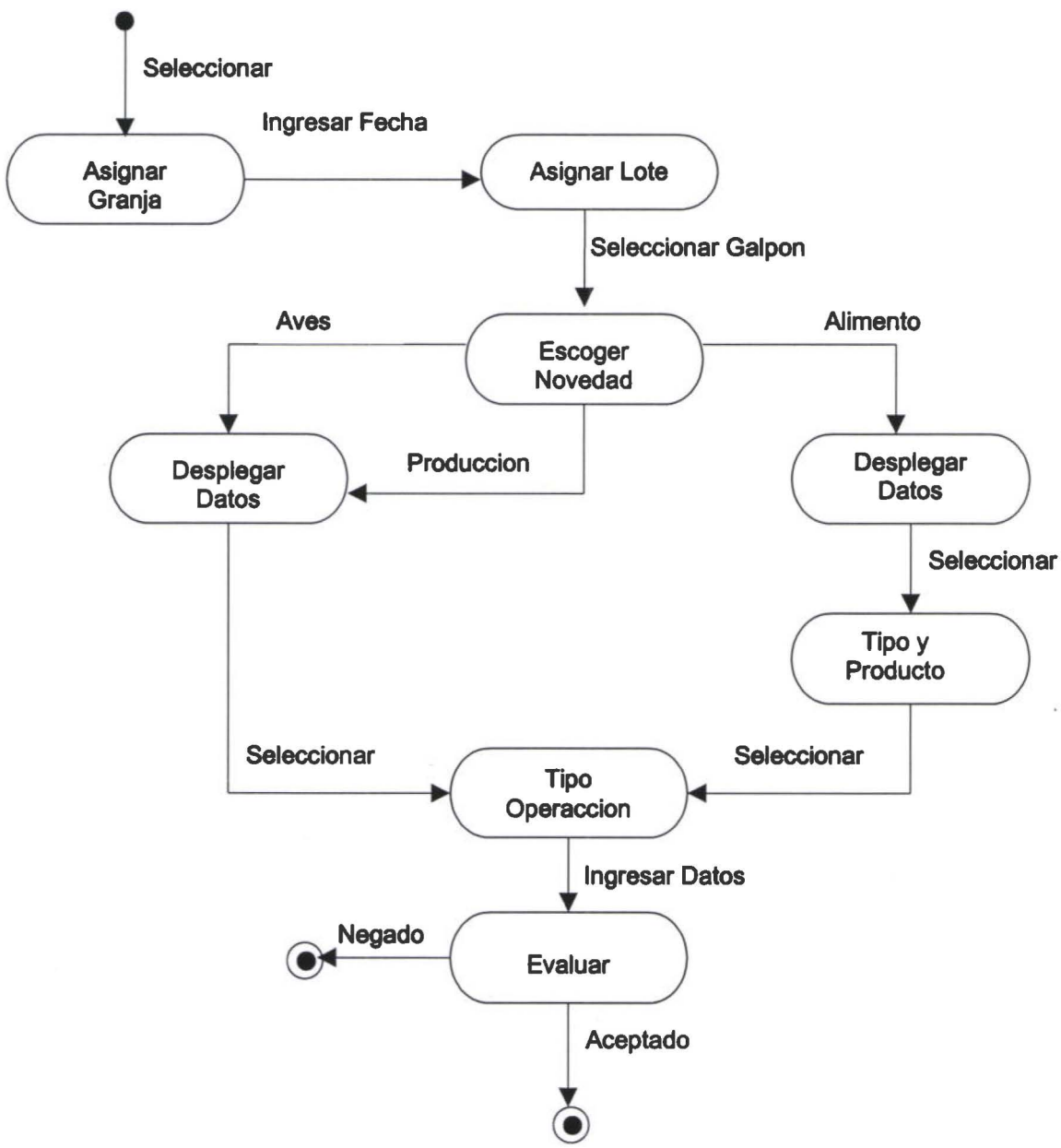
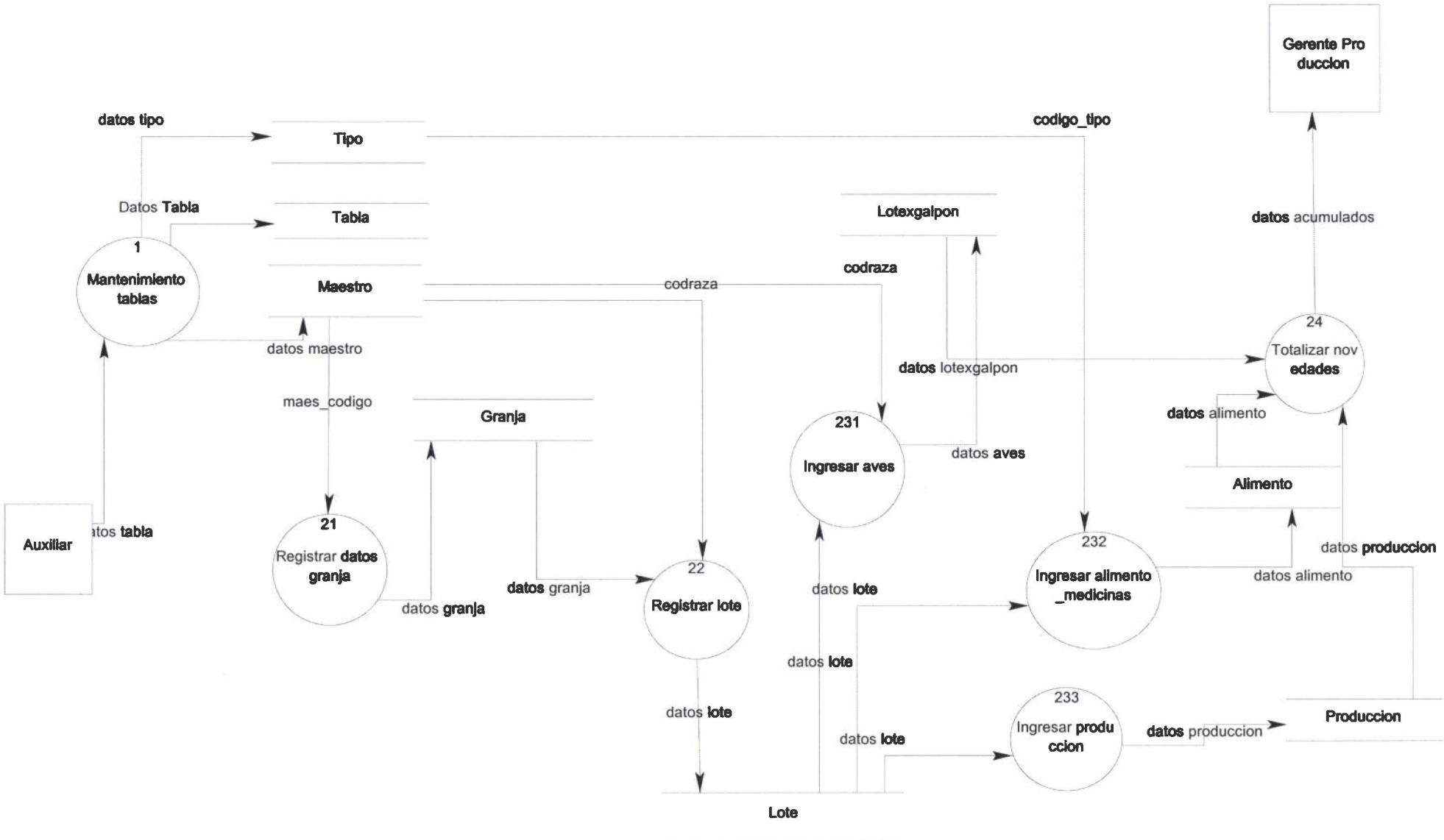


Diagrama de Estados: Movimientos



ANEXO F

DIAGRAMA DE FLUJO DE DATOS


Gerente Pro ducclon
datos tipo
Tipo
codigo_tipo
Datos Tabla
Tabla
Lotexgalpon
datos acumulados
1
Mantenimiento tablas
Maestro
codraza
codraza
24
Totalizar nov edades
datos maestro
datos lotexgalpon
maes_codigo
Granja
231
Ingresar aves
datos alimento
datos aves
Alimento
Auxiliar
tos tabla
21
Registrar datos granja
232
Ingresar alimento _medicinas
datos produccion
datos alimento
datos granja
datos granja
22
Registrar lote
datos lote
datos lote
datos lote
233
Ingresar produ ccion
datos produccion
Produccion
datos lote
Lote

ANEXO G

PRUEBAS

En el cliente

Se realizo las siguientes pruebas en el cliente para descubre errores que no detectaron oportunamente en fases tempranas.

- **Con relación a la IGU**
  - Se puede utilizar el teclado o el ratón alternadamente para avanzar en los distintos campos de las ventanas
  - El tamaño de la ventana esta predeterminado y no se permite ser ajustado, el lo referente a mover y desplegar la ventana se las realiza sin inconvenientes, además existe una opción que me permite definir la forma de despliegue de las ventanas.
  - Todo el contenido de la una ventana es accesible adecuadamente con el ratón, no existe abreviaturas de tecla ni teclas de función, las flechas de dirección y teclado se han comportado adecuadamente.
  - Se despliega en la ventana los menús emergentes, barras de herramientas, barras deslizantes, cuadro de diálogo, botones, iconos y los controles que son necesarios sin registrarse uno inconveniente
  - Al desplegar varias ventanas la que esta activa se resalta adecuadamente y se presenta siempre en primer plano.
  - Se navega, sale y cierra una ventana sin causar inconvenientes.

- **Menús emergentes y operación con el ratón**

Se probo que las funciones de los menús son accesibles al teclado y al ratón, así como con la letra de identificación de opción, los nombres son claros y muy entendibles.

- **Entrada de datos**

  - Los datos que son necesarios ser validados son reconocidos y se emites mensajes que nos permite identificar que son datos no validos.

En el servidor

- Se utilizo el comando ping con la dirección 192.168.1.240, desde el cliente obteniéndose una correcta respuesta desde el servidor

```
C:\>ping 192.168.1.240

Haciendo ping a 192.168.1.240 con 32 bytes de datos:

Respuesta desde 192.168.1.240: bytes=32 tiempo=1ms TDV=255
Respuesta desde 192.168.1.240: bytes=32 tiempo<10ms TDV=255
Respuesta desde 192.168.1.240: bytes=32 tiempo<10ms TDV=255
Respuesta desde 192.168.1.240: bytes=32 tiempo<10ms TDV=255

Estadísticas de ping para 192.168.1.240:
    Paquetes: enviados = 4, Recibidos = 4, perdidos = 0 (0% loss),
Tiempos aproximados de recorrido redondo en milisegundos:
    mínimo = 0ms, máximo =  1ms, promedio =  0ms
```

- No se pudo reducir los recursos del servidor, espacio en disco, memoria, etc. puesto que el servidor se encuentra disponibles para otra aplicaciones.
- Se ingreso de un registro completo se lo grabo, se consulto y comprobó su almacenamiento, se lo modifico un campo y se lo grabo de nuevo, se procedió a comprobar la actualización, se borro el registro. No presenta novedades al realizar estas pruebas
- Se realizaron conexiones al servidor con otras aplicaciones, no se presento algún inconveniente con las aplicaciones.

En las comunicaciones

- **Hardware de red.**

  Cables de red

  - Se dispone de una topología en estrella con cables UTP categoría 5, en el cual se verifico visualmente la correspondencia en los filamentos en cada uno de los extremos, así también con la ayuda de un comprobador de redes.

  Adaptadores de Red

  - Se verifico la existencia física de la tarjeta de red y que la misma se encuentre instalada correctamente en el zócalo. Además se comprobó que no exista ningún tipo de conflicto con su dirección utilizar el comando ping a la dirección de la maquina.

  Hubs

  - Se comprobó visualmente que en no existan colisiones en el hub.

- **Software de Red**

  Adaptadores de Red

  - Se comprobó que los drivers de la tarjeta que se encuentran instalados en el equipo sean los correctos, así como los protocolos se encuentre bien configurados (TCP/IP).

- **Prueba de ODBC**

  - Se comprobó que se encuentre instalado el ODBC para informix (informix3.32 ver2.50TC2) y si este esta interactuando correctamente con la base de datos. No se presento ningún tipo de inconveniente al realizar el test de comprobación.

En el sistema

En el cliente

- Al instalar el sistema se verifico que se creo en el cliente el directorio OWL PRODUCCION, además no se modifico el archivo autoexec.bat ni el win.ini puesto que la aplicación no utiliza estos archivos. Se creo el icono de acceso directo a la aplicación.

En el Servidor

- Esta tareas fueron realizadas y revisadas por el administrador sin presentarse problema y alteración de algún software instalado en el servidor.

- Las pruebas de recuperación no se las realizo por cuanto el servidor no puede ser dado de bajo abruptamente ya que en el se encuentra instalado otras aplicaciones de la empresa, esta tarea se la realiza manualmente (realizada por el administrador). El rendimiento se podría decir que es el normal.

ANEXO I

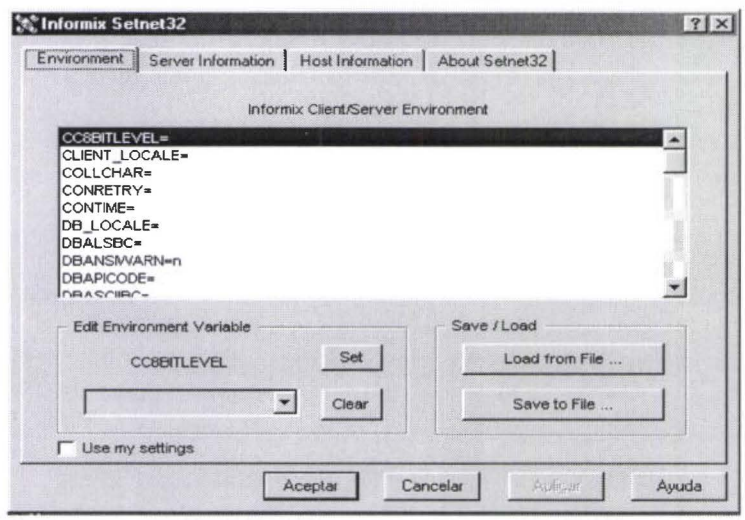
MANUALES DE INSTALACION

MANUAL INSTALACION

El primer paso es revisar si se encuentra instalado el cliente para informix, (INFORMIX Client-SDK), este software instala el cliente así como el manejador del ODBC para informix, una vez instalado este cliente se debe editar el archivo Services en el directorio windows y agregar lo siguiente:

Nombre del servicio:	turbo
Número del puerto:	1526
Protocolo:	tcp

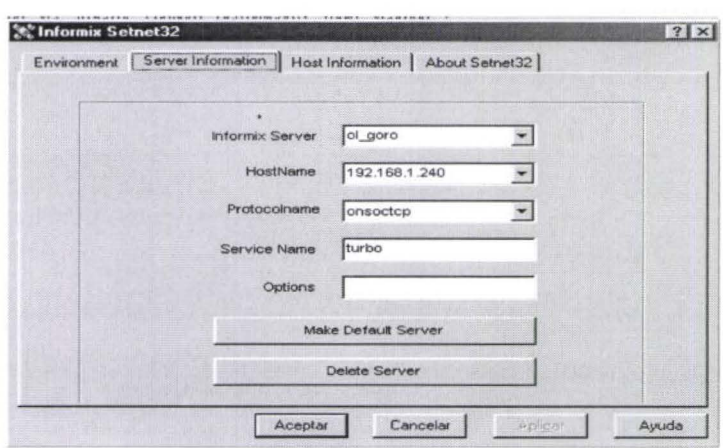
El siguiente paso es ejecutar el archivo setnet32 en el cual se definen las variables de medio ambiente y los parámetros del informix.



Las variables que se definen son:

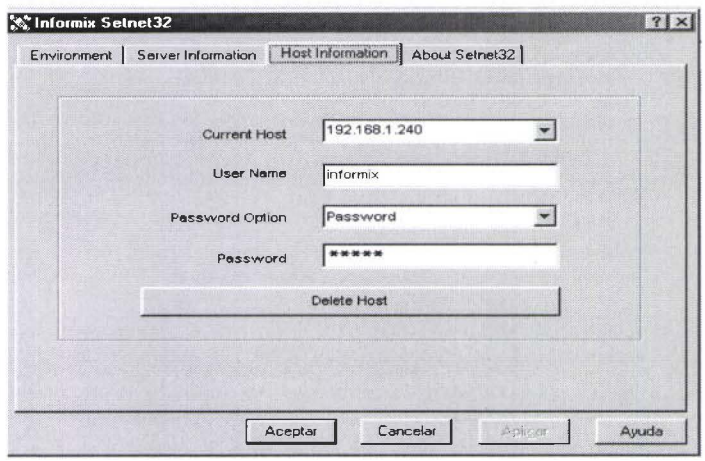
INFORMIXDIR = c:\informix\	Directorio donde se encuentra el cliente
INFORMIXSERVER= ol_goro	Nombre del servidor
DBDATE=MDY/4	Formato de la fecha

En pestaña siguiente se define la información del servidor



Informix Server	ol_goro	Nombre del servidor
HostName	192.168.1.240	De preferencia se pone la dirección IP del servidor
ProtocoloName	onsoctcp	El protocolo de comunicación que se utiliza
Service Name	turbo	El nombre del servicio, se definió en archivo services en la carpeta de Windows

Se debe definir el cliente en la pestaña (Host Information)



Current Host	192.168.1.240	Dirección IP del servidor
User Host	informix	Nombre del usuario (informix)
Password Option	Password	Definir se utiliza o no password el usuario (informix)
Password		La clave se lo define en la opción de password

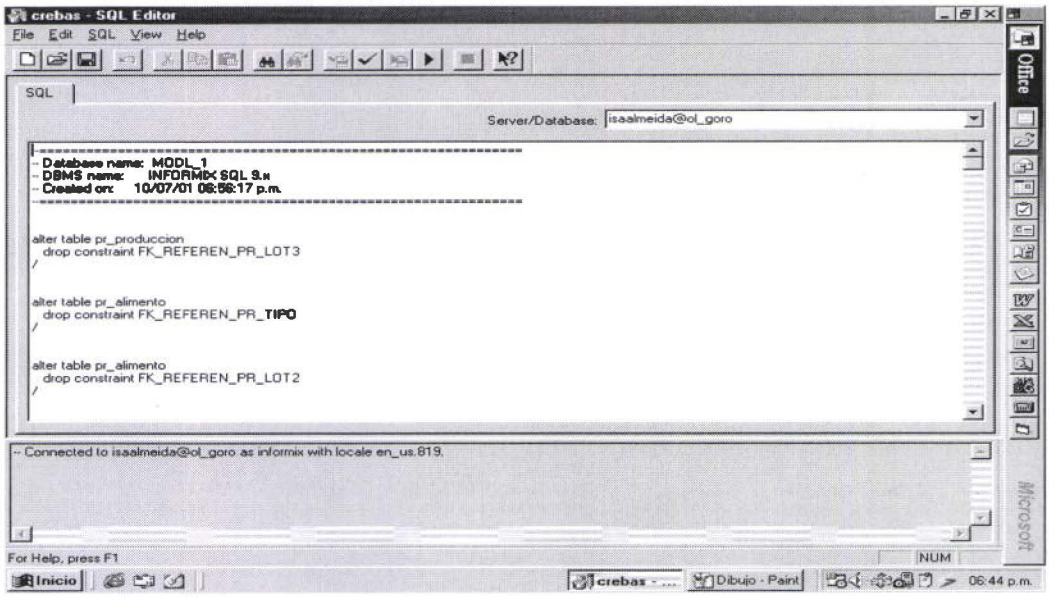
Al terminar de llenar estos parámetros se a configurado un cliente en informix.

Todos estos pasos debemos hacer si no esta instalado el cliente informix en el computador.

Si ya esta instalado el cliente informix pasaremos a instalar directamente la aplicación para lo cual ejecutaremos el programa setup directamente desde el CD.

Se contestara a las interrogantes que se van desplegando en la pantalla hasta terminar de instalar la aplicación.

Paso seguido se crea en la base de datos de la empresa respectiva las tablas con ayuda de un utilitario de informix (sqleditor) en donde se ejecutara los siguientes archivos:



- Creabas.sql crea tablas de la aplicación.
- Creabas-prs.sql crea tablas de seguridad.
- Usuarios.sql inicializa las tablas de seguridad para el manejo de los menús, y la creación del usuario Administrador.
- DatosIniciales.sql inicializa las tablas pr_config y pr_maestro con los datos.

ANEXO H

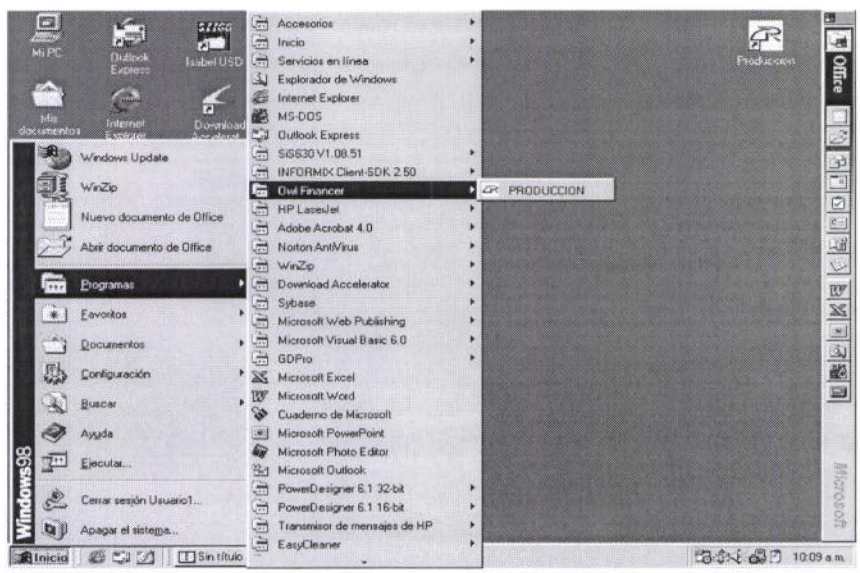
MANUALES DE USUARIO

Manual de Usuario

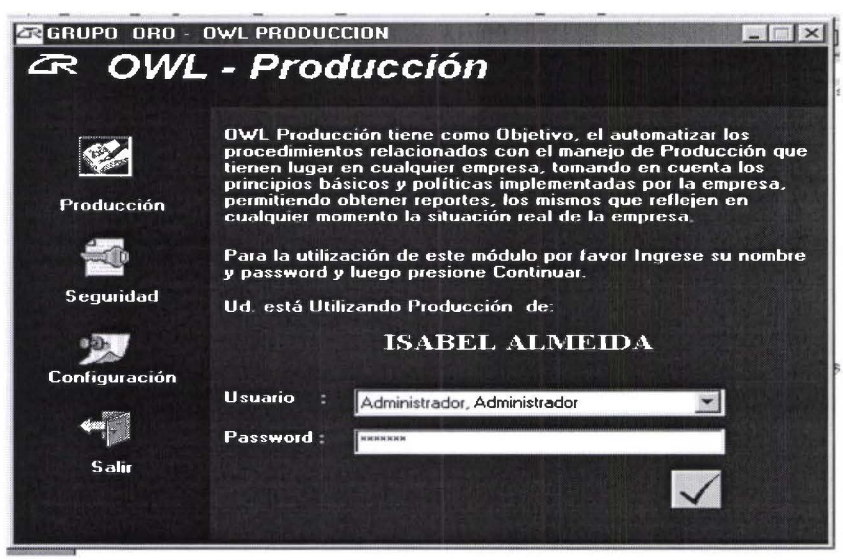
El manual de usuario presupone que el usuario esta familiarizado con ambientes gráficos, es decir sabe utilizar el ratón, los iconos y todo lo que le diferencia a esta interfaz.

Apertura del Sistema

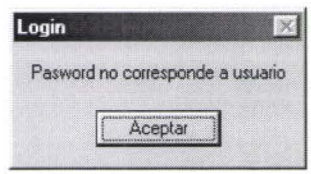
Se pude ingresar de dos formas al sistema, una se encontrara en el sistema directamente el icono de Producción, la otra forma es utilizar el ambiente Windows, elegir: Inicio/Programas/OWL Producción/Producción en la pantalla del sistema,



Se selecciona una de ellas se inicia la aplicación aparece la ventana de bienvenida y control de seguridad en la cual se deberá ingresar un nombre de usuario y la clave correspondiente, las mismas que las proporciona el administrador de sistema.

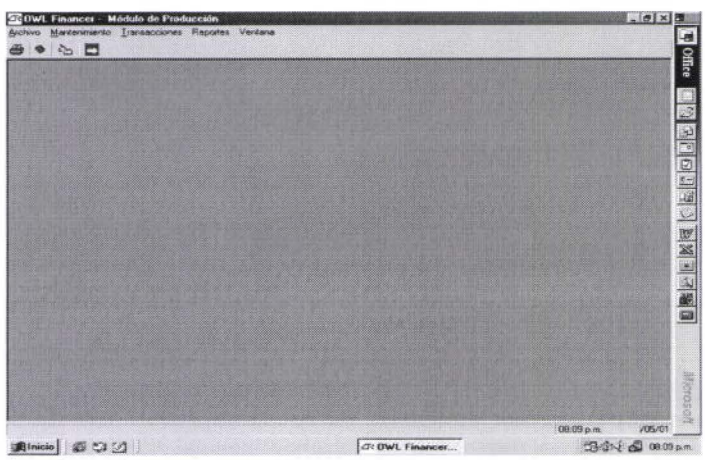


Si la clave o el usuario es incorrecto, el sistema muestra un mensaje: Password no corresponde a usuario.



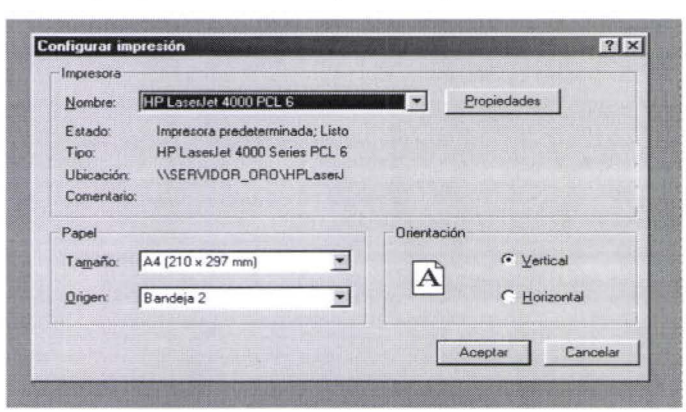
En esta pantalla se tiene cuatro opciones la primera Producción.
La opción de seguridad nos permite crear usuarios del sistema.
La opción de Configuración nos permite escoger en cual empresa vamos a trabajar.
La opción de Salir para abandonar la aplicación.

Si la clave es correcta ingresa al entorno del sistema, en la cual aparece una ventana principal con el siguiente menú: Archivo, Mantenimiento, Transacciones, Reportes, Ventana; cada uno con sus respectivos submenus.

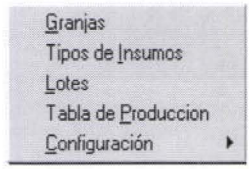


Al escoger Archivo se desplegará el siguiente menú

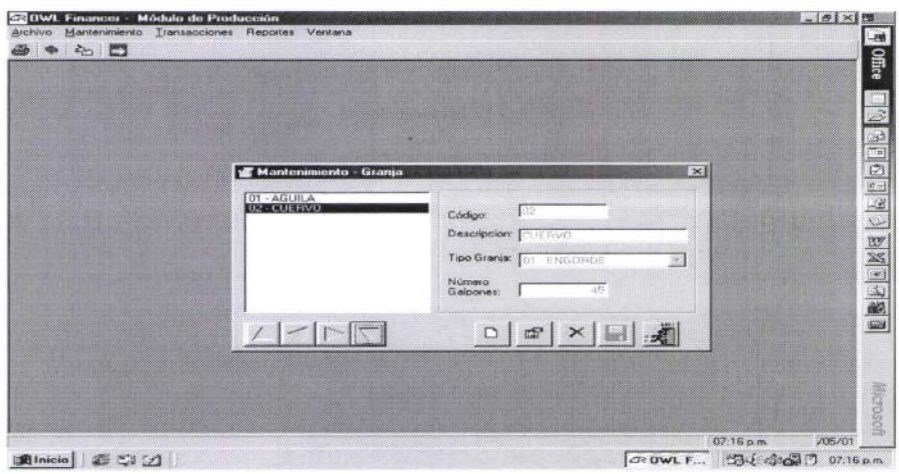
Imprimir.- Permite enviar los datos del menú de mantenimiento a la impresora.
Preparar Impresora.- Desplegará la pantalla en la cual nos permite seleccionar una impresora en particular para enviar los trabajos de impresión.



Salir.- Permite cerrar la ventana activa y salir del sistema.
Al escoger Mantenimiento se despliega el siguiente menú:



Granja.- con esta opción ingresan los datos generales de las granjas. Nos aparece la siguiente pantalla que me permite capturar los datos básicos de las granjas, añadir, modificar, o en su defecto borrarlo.



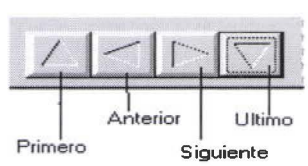
En el Código se ingresa un identificador de la granja el cual va a ser de dos caracteres, el mismo que nos va a permitir identificar de forma única a la granja.

El campo Descripción en (20 posiciones) de nueva granja ó en caso de modificación.

En el tipo de granja nos permite seleccionar a que tipo de granja pertenece esa granja en particular. (Este dato debe estar previamente creado.

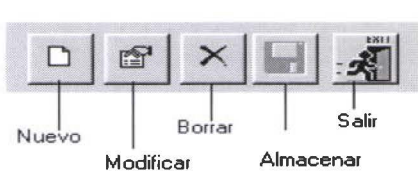
El campo Número de galpones nos permite ingresar el número de galpones totales de una granja.

En esta pantalla además nos permite navegar entre los registros ingresados con las herramientas con ayuda en línea.



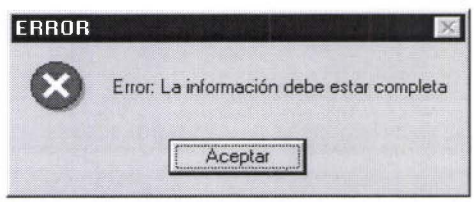
El botón de Primero, permite ubicarme en el primer registro de la pantalla activa.
El botón de Anterior, regresa al registro anterior del actual.
El botón de Siguiente, se mueve un registro delante del actual.
E botón de Ultimo, se ubica en el último registro.

Además tenemos los siguientes botones:

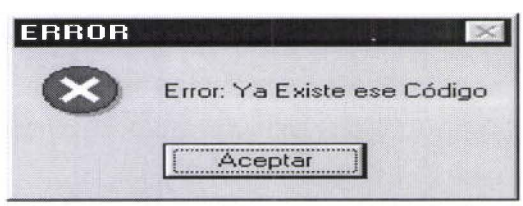


El botón Nuevo, permite ingresar un nuevo registro.
El botón de Modificar, permite realizar las modificaciones en las tablas.
El botón de Borrar, permite eliminar un registro.
El botón de Almacenar, permite guardar el registro activo.
El botón de Salir, permite cerrar la pantalla activa y regresar al menú principal.

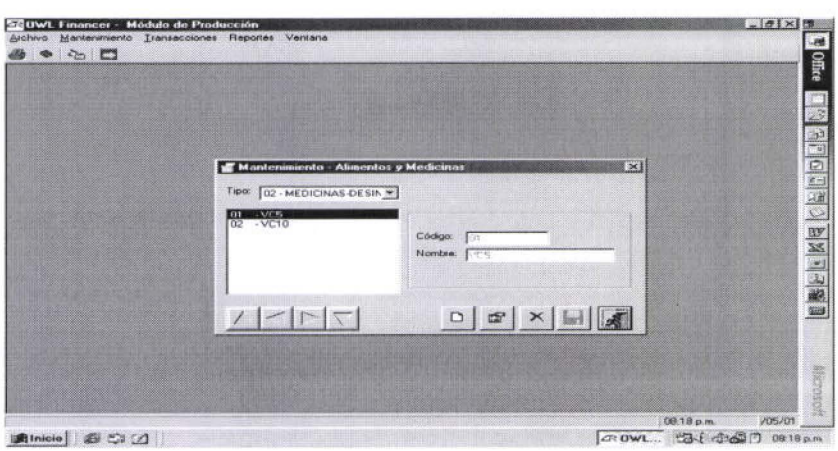
Cuando uno de los datos de entrada está en blanco o no tiene información se presenta el siguiente mensaje.



Además cuando se ingresa en el campo Código una identificación ya existente se despliega el siguiente mensaje de error.



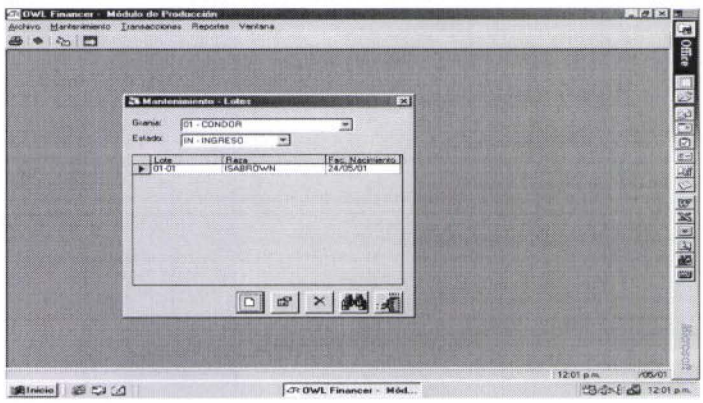
Tipos de insumos.- Cuando escogemos la segunda opción del Mantenimiento aparecerá la siguiente pantalla la misma que nos permite añadir un nuevo tipo de alimento y o medicina, modificar o borrarlo.



En el campo Tipo, se debe seleccionar el tipo de insumo (Alimento, Medicinas, etc.) y al escoger uno en particular y existir datos ya ingresado para ese tipo en el cuadro de lista se desplegará los ítems existente. Además nos permite ingresar un nuevo ítem, modificar, borrar si es el caso. Así como también movernos dentro de la lista.

En el campo Código (3 posiciones) se debe ingresar el identificador de este ítem.
El campo Nombre en (20 posiciones) de nueva descripción del ítem ó en caso de modificación la nueva descripción.

Lotes.- Cuando escogemos la tercera opción del Mantenimiento aparecerá la siguiente pantalla la misma que nos permite añadir un nuevo lote, modificar o borrarlo se permiten borrar un lote siempre y cuando no exista ningún movimiento para ese lote.



En el campo Granja, se debe seleccionar la granja en la cual se va a trabajar:
El campo estado, por omisión nos desplegará los lotes que se encuentra en ingreso.
Nos permite seleccionar el estado.

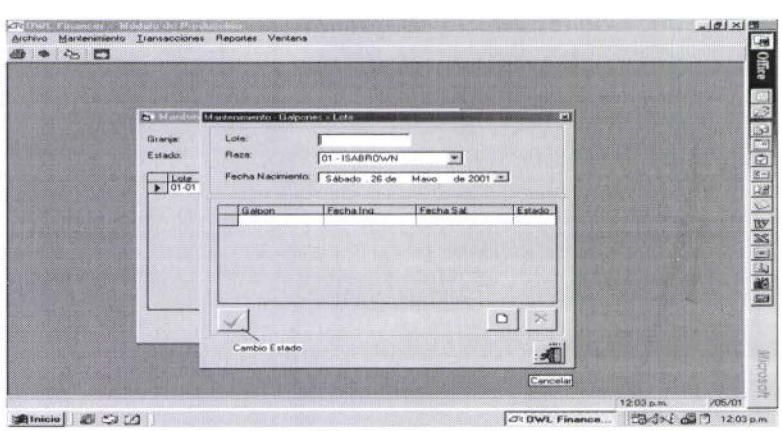
IN – Ingreso
PR – Producción
BA - Baja

En la lista se desplegará los lotes de la granja. Que se encuentran en el estado escogido.

Además tenemos los botones de:

El botón Nuevo, permite ingresar un nuevo registro.
El botón de Modificar, permite realizar las modificaciones en las tablas.
El botón de Borrar, permite eliminar un registro.
El botón de Buscar, permite localizar un lote ya sea por Lote, por Raza o por fecha de nacimiento, dentro del la lista.
El botón de Salir, permite cerrar la pantalla activa y regresar al menú principal.

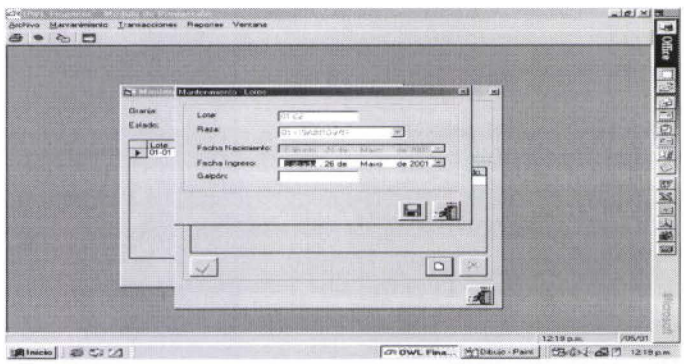
Al presionar Nuevo se desplegará la pantalla:



Se deberá ingresar un lote para identificarlo, se llene este campo Lote.
Se escogerá la raza a la cual pertenece el lote.
La fecha de nacimiento por omisión se desplegará la fecha del sistema.

En esta ventana se puede presionar los botones de nuevo y salir quedando bloqueado el de eliminar y el de cambio de estado del lote.

Al presionar nuevo se desplegará la siguiente ventana:

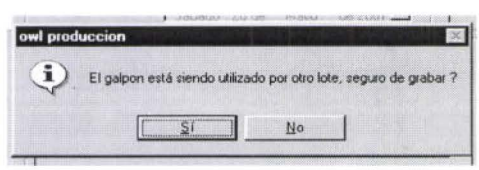


En la cual se desplegará los campos de Lote, Raza, Fecha de Nacimiento

Se deberán llenar los datos de Fecha de ingreso la cual por omisión se desplegará la fecha del sistema. Esta fecha no deberá ser menor a la fecha de ingreso si esto ocurre se desplegará el siguiente mensaje de error

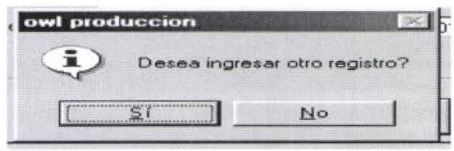
En el campo galpón se deberá ingresar el número del galpón al cual se le está asignando este lote. Este Número no podrá exceder el número de galpones que se dispone en una granja. Si esto sucede se desplegará el siguiente mensaje de error.

Además si ingresa un galpón que ya esta asignado le desplegará una ventana de dialogo a la cual Ud. Deberá contestar afirmativamente o negativamente según sea el caso.



Si cuando se va a tener más de un lote en el galpón y No cuando se anula el ingreso y regresa al campo Galpón.

Cuando se ingresa un galpón que se encuentra vacío o disponible y se presiona el botón de guardar me permite seguir asignando galpones a un lote en particular. Para lo cual se desplegará el siguiente cuadro de dialogo.



Al escoger en la pantalla de Lote modificar me permite modificar el estado de un lote pasando de ingreso a producción y de producción a baja, a este estado se pasa siempre y cuando el número de aves en el lote y galpón sea cero, alimentos - medicinas y producción igual también igual a cero.

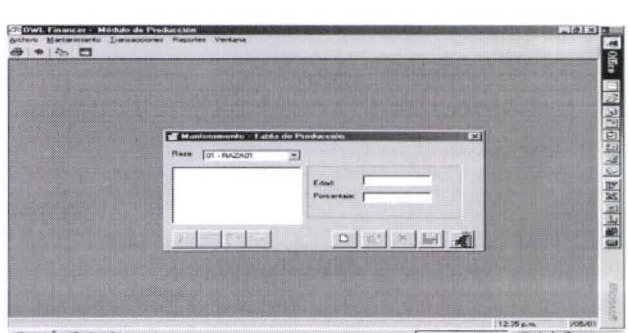
Para borrar un Lote este no debe tener ningún dato de transacciones realizado, es decir no debe tener datos asociados a ese lote y galpón para poder borrar.

El botón de cambio de estado me permite cambia en orden secuencial de un estado a otro, de Ingreso a Producción y a baja. Para que un lote galpón sea dado de baja sus saldos finales deben ser igual a cero, sino es el caso se presenta la siguiente ventana:

Al estar todo correcto se presenta la siguiente ventana en donde nos pide confirmar la fecha de baja, presentando por omisión la del sistema.

Al presionar el botón de Guardar confirmamos la fecha de salida, y al presionar salir abandonamos esta ventana para regresar ventana principal de Lote.

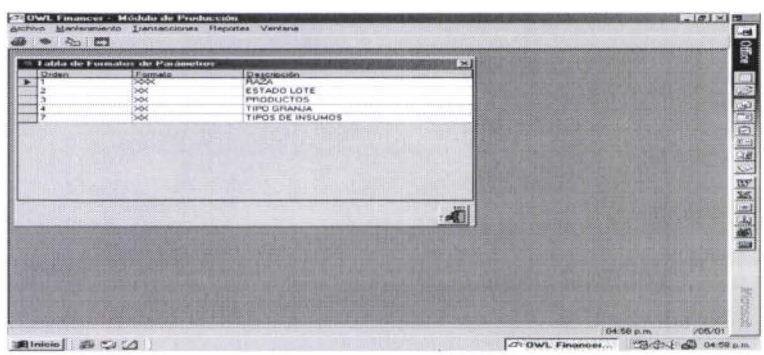
Tabla de producción.- Cuando escogemos la cuarta opción del Mantenimiento aparecerá la siguiente pantalla la misma que nos permite añadir, modificar o borrarlo los datos de postura para las aves ponedoras, estos son proporcionada por los distribuidores para una edad.



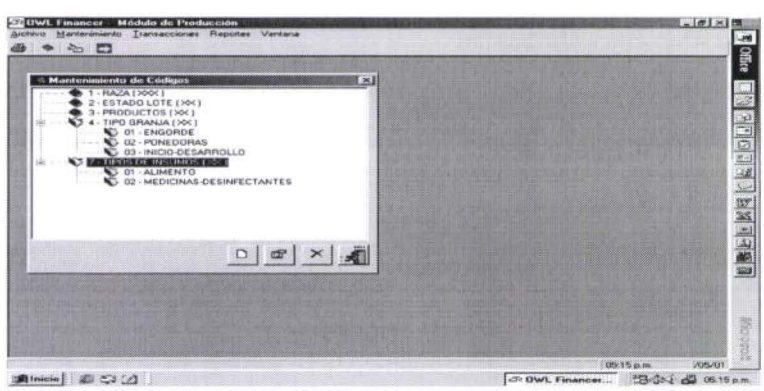
En el campo Raza, permite escoger la raza con la cual vamos a trabajar.
En el campo Edad, se ingresa el número de semanas.
El Porcentaje es un dato de referencia de producción para esa raza y edad en la cual se encuentra el ave.

Configuración.- Al escoger la quinta opción del Mantenimiento se despliega un submenú con las opciones de Tabla de Formatos, Códigos Maestros.

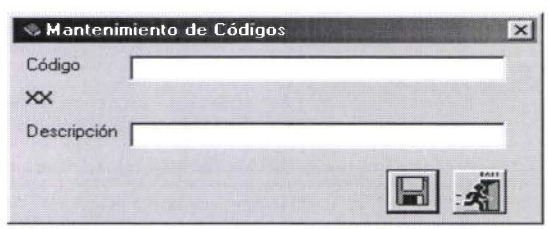
Tabla de Formatos.- Con esta opción nos permite visualizar cuales son los parámetros con los que se inicializa al sistema el instante de su instalación.



Códigos Maestros.- Con esta opción nos permite ingresar componentes que pertenecen a la tabla de formatos, así como modificar y borrar uno en particular si es el caso.

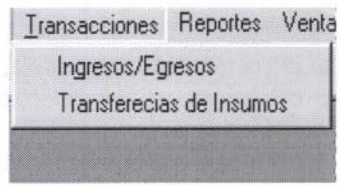


Al dar clip en nuevo aparece la siguiente ventana.

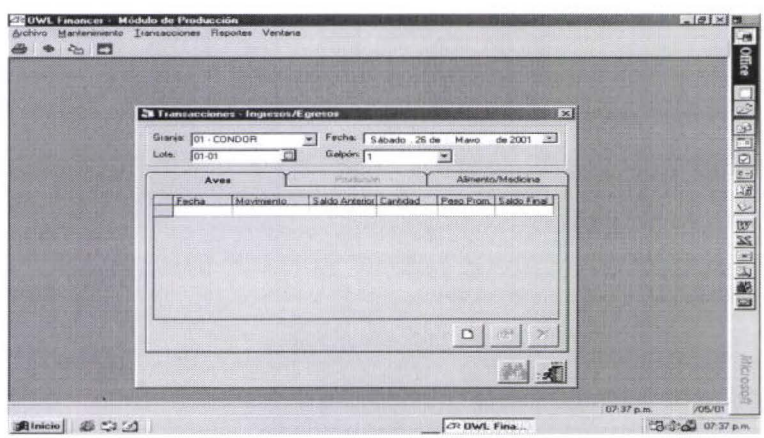


En donde Código es el identificador dentro de tabla de formatos, en la parte inferior se despliega cual debe ser el formato que se debe usar para este campo. La Descripción es el detalle del código de (20 posiciones)

Al escoger la opción del menú principal Transacciones se despliega el siguiente menú:



Ingresos/Egresos.- En el cual se registran las transacciones o novedades que se presentan en lote por galpón, en este submenú si se elige Ingresos/Egresos se desplegará la siguiente pantalla:

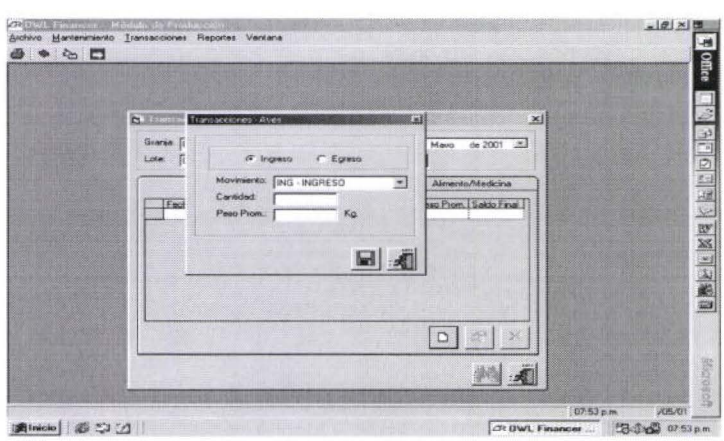


El campo Granja se selecciona la granja en la cual va ha trabajar o va ha realizar la transacción.
El campo Fecha se registra la fecha en la cual se va ha realizar la operación
El campo Lote se ingresa o se selecciona de la lista los lotes que dispone la granja.
El campo Galpón es llena automáticamente al escoger el lote.

Si la granja seleccionada es una granja del tipo Engorde no se habilitará la pestaña de producción caso contrario esta opción estará disponible.

En cada uno de las pestañas se desplegará las novedades que se están registrando para ese lote galpón y fecha.

Al escoger la pestaña de Aves se despliega la siguiente ventana:

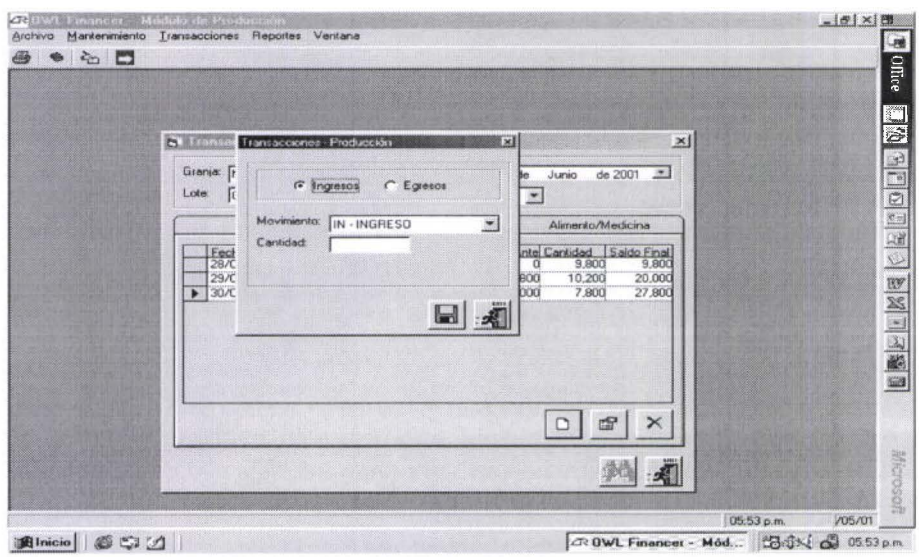


En donde se escogerá que tipo de movimiento se va ha realizar si es de ingreso o egreso. Un movimiento de ingreso se podrá realizar solo cuando el lote galpón se encuentre de estado de ingreso. Un egreso se realiza sin ninguna restricción solo controlando que tengamos saldos para realizar esta operación.

En el campo Cantidad digitar cual es el número de unidades que se van ha realizar esta operación.
En el campo Peso Prom. digitar cual es el peso promedio de la cantidad del movimiento.

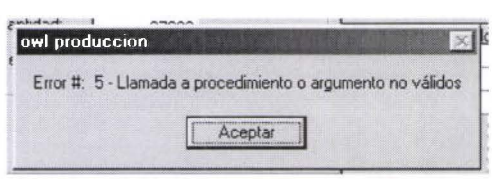
Al dar clip en el botón de aceptar se almacena este registro como una novedad de aves y del tipo que se ha elegido.

Al escoger la pestaña de Producción se despliega la siguiente ventana:

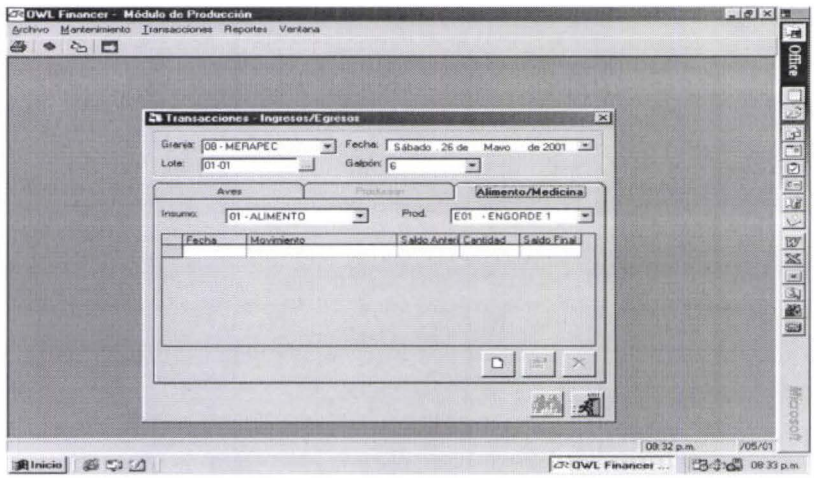


En donde se escogerá si se va ha realizar una operación de Ingreso o Egreso. Al escoger un ingreso nos permitirá ingresar la cantidad ingresada de producción. Al escoger un Egreso nos desplegará la siguiente venta.

En la cual a más de la cantidad del movimiento se debe seleccionar el destino del movimiento de lista. Si no se selecciona este el sistema desplegará el siguiente mensaje y no le permitirá continuar.

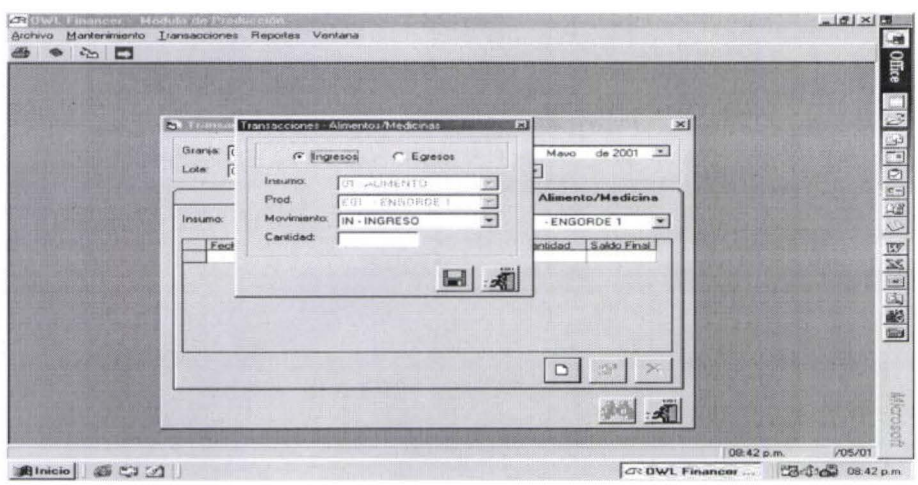


Al escoger la pestaña de Alimento/Medicina se despliega la siguiente ventana:



En el campo Insumo se seleccionará si va hacer un Alimento o una Medicina Desinfectante el que se va registrar.
En el campo Prod. Cual es el alimento o medicina que se va ha registrar.

Al dar clip en el botón de nuevo se desplegará la siguiente ventana:

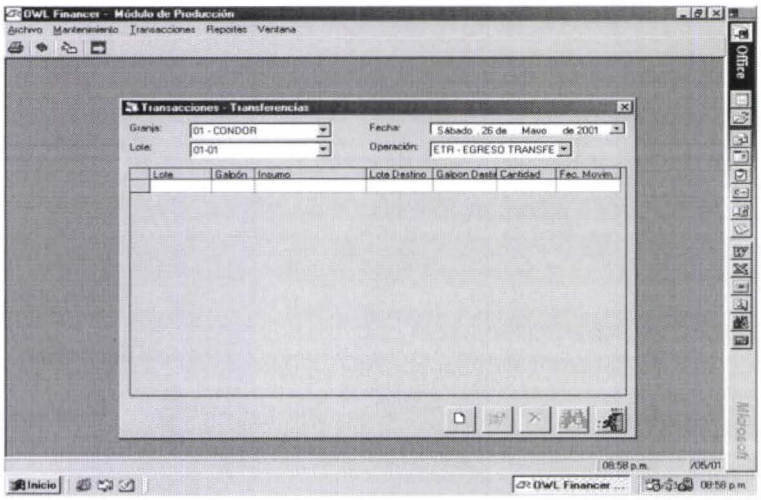


Se selecciona el tipo de operación que se va ha realizar sea un ingreso o un consumo (egreso)

En el campo Cantidad se registra cual es la cantidad que se va a afectar dentro de la operación.

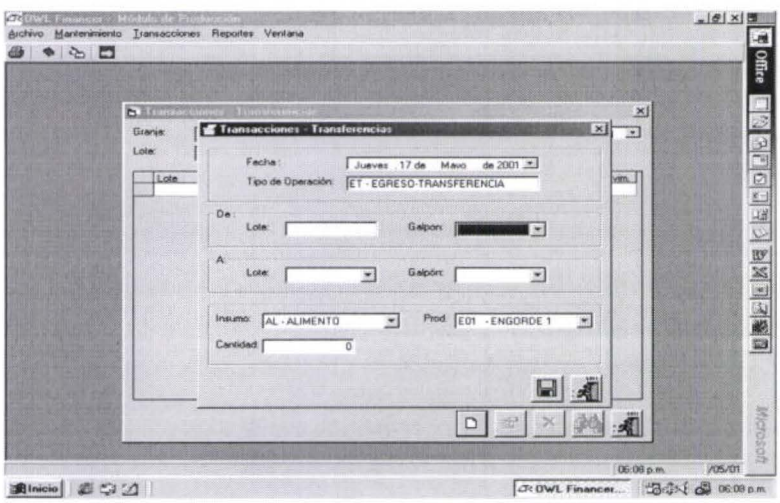
Transferencia de Insumos.- Al escoger en el Submenú de Transacciones la opción de Transferencia de Insumos se despliega la siguiente ventana:

La misma que nos permite realizar un Nuevo registro, Modificar, Eliminar si fuera necesario y buscar de acuerdo al campo seleccionado.



En el campo Granja se debe seleccionar la granja en la cual se desea realizar el movimiento,
En el campo Fecha se desplegará la del sistema por omisión, pudiendo estar ser cambiada por el usuario.
En el campo Lote se debe seleccionar el lote origen de la operación.
En el campo operación se desplegará la operación de transferencia (Egreso)

Al seleccionar nuevo se desplegará la siguiente ventana la misma que se encuentra divida en cuatro regiones.



La primera compuesta por dos campos, Fecha del movimiento y Tipo de Operación
La segunda datos de lote origen y se deberá seleccionar el galpón origen del movimiento.
La tercera datos del lote y galpón destino los cuales se deberán seleccionar de la lista.
La cuarta compuesta del tipo de insumo (alimento o medicina), el campo del producto, y la cantidad del movimiento.

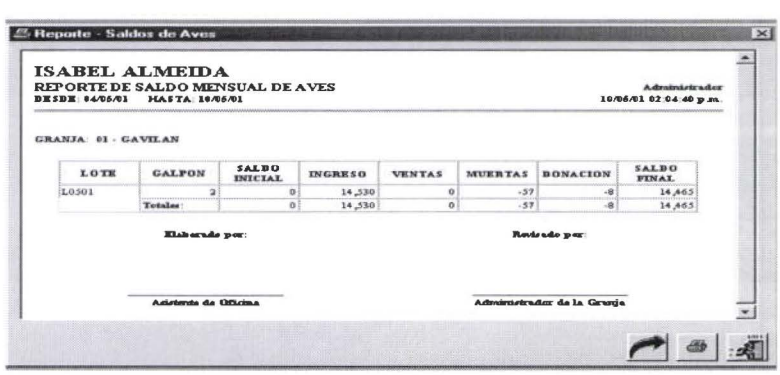
Esta operación genera dos movimientos egreso en el origen y un ingreso en el destino.

Reportes.- Al escoger del menú de Reportes se despliega un submenú con opciones de Saldos y Gráficos.

Al escoger en del submenú de Reportes la opción de saldos se despliega las opciones de Aves, Producción, Alimentos/Medicinas. Se desplegará la siguiente pantalla

En el campo Granja se deberá seleccionar la granja de la cual se va ha obtener el reporte.
En los campos Desde y Hasta se desplegará la fecha del sistema por omisión la fecha hasta no podrá ser mayor a la fecha del sistema, ni menor al campo desde.

Al dar un clip en el botón de Impresora se desplegará la siguiente ventana:



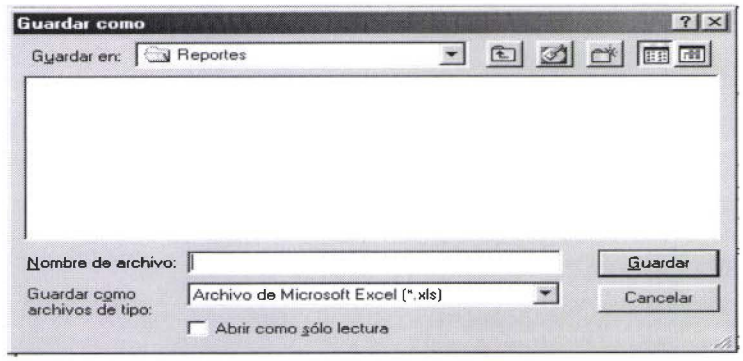
En el reporte se desplegará los siguiente datos, como cabecera del reporte.

- Nombre de la Empresa
- Titulo del reporte
- Los datos desde y Hasta así como la fecha y hora de la generación
- El nombre de la granja
- Firmas de Responsabilidad

En el detalle se tendrá los siguientes datos, Lote, Número de Galpón, Saldo Inicial, Ingresos, Egresos (muertas, ventas y donadas), Saldo final.

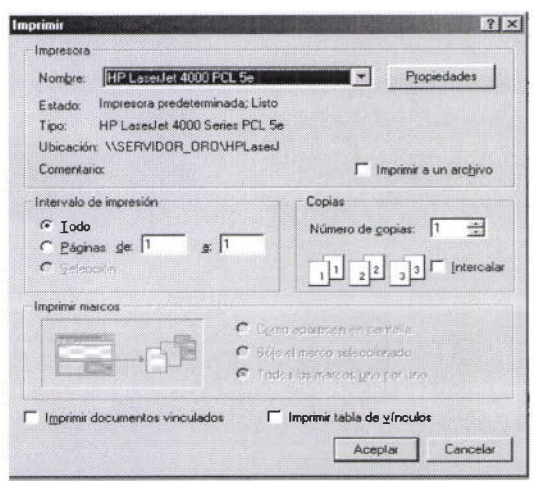
En la parte inferior derecha de la ventana se presenta tres botones el de mas de la izquierda (una flecha) sirve para exportar el reportes a excel (Guardar como), el botón del centro (una impresa) sirve para enviar el reporte a la impresora (imprimir), y el tercero para abandonar (salir).

Al dar doble clip en el botón de Guardar como se desplegará la siguiente ventana:



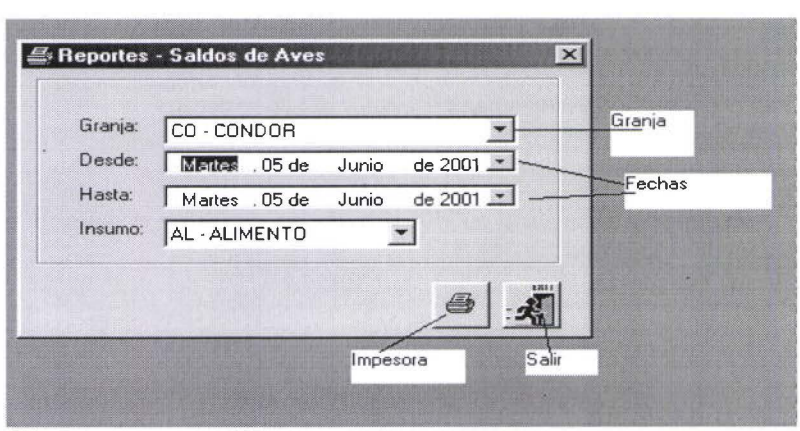
En esta pantalla nos permite definir la ubicación y el nombre del archivo exportado.

Al dar doble clip en el botón de Imprimir se desplegará la siguiente ventana:

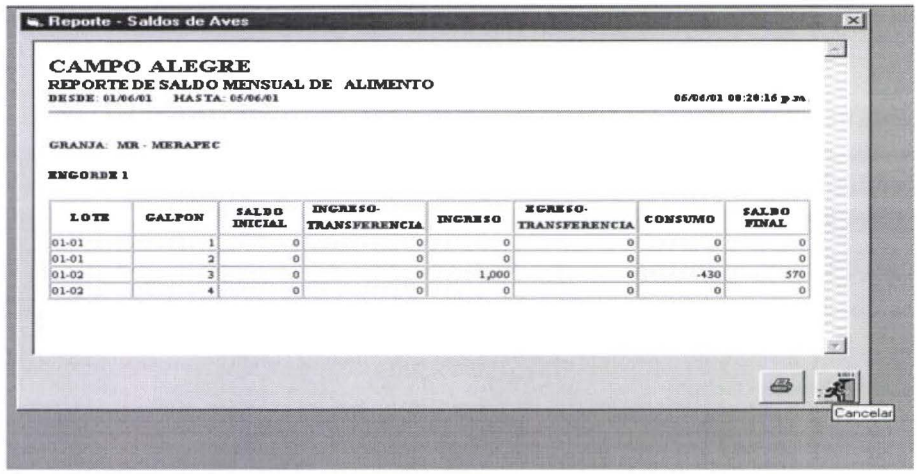


En esta pantalla se define los parámetros que va a manejar la impresora.

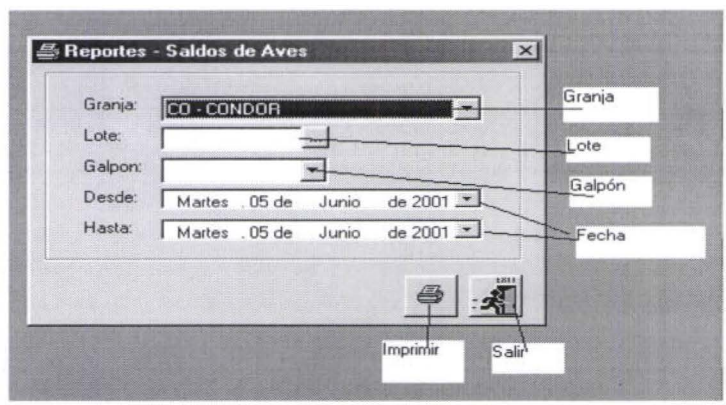
Al escoger en el submenú de Reportes y dentro de esta la opción de Alimento/Medicina. Se desplegara la siguiente ventana.



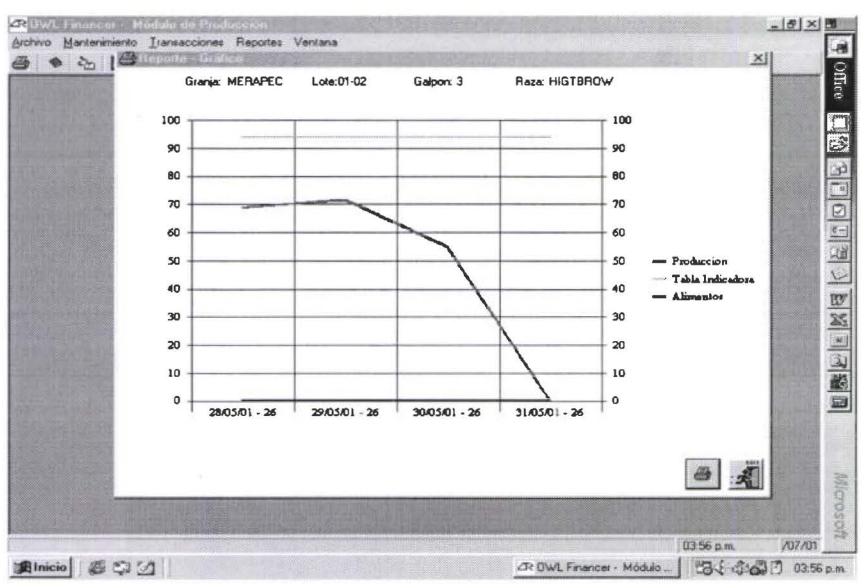
En el campo Insumo se debe seleccionar entre Alimentos y Medicina para obtener el reporte. Desplegándose la siguiente ventana.



Al escoger del submenú de Reportes la opción de Gráfico se despliega la siguiente ventana:



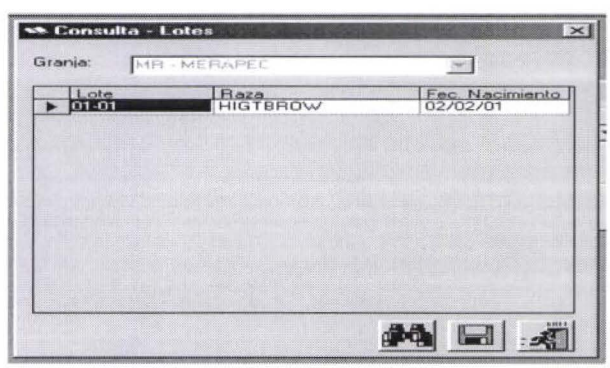
Al llenar los datos de Granja, Lote, Galpón, Fechas Desde y Hasta se desplegara la siguiente ventana.



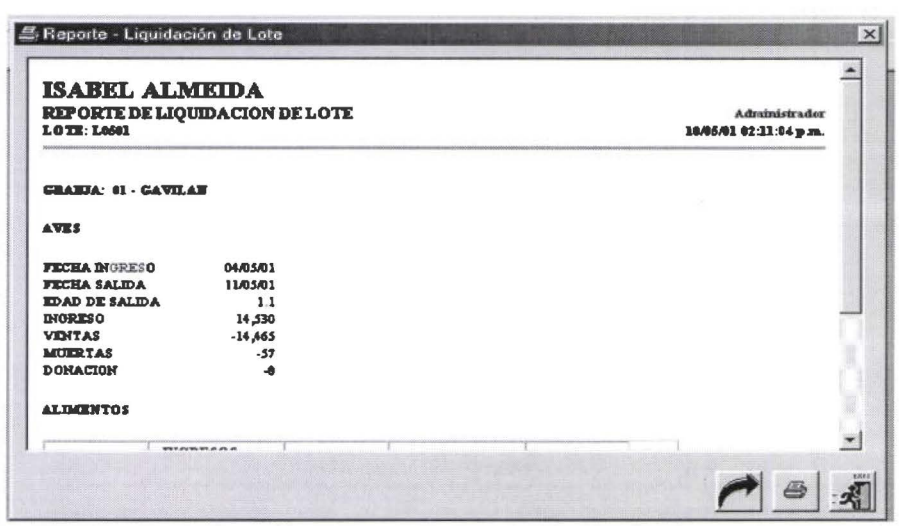
En la parte inferior derecha de la ventana se presenta dos botones el de mas de la izquierda (una impresa) sirve para enviar el reporte a la impresora (imprimir a la impresora predeterminada), y el segundo para abandonar (salir).

Al escoger del submenú de Reportes la opción de Liquidación de Lotes se despliega la siguiente ventana:

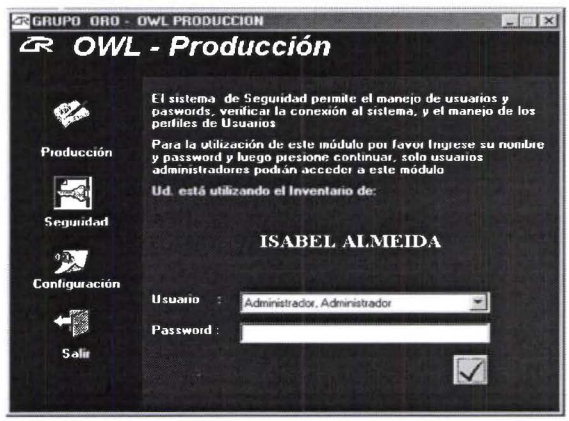
Se debe seleccionar la granja de la cual vanos a obtener el reporte. En el campo Lote se puede digitar directamente el lote o se puede seleccionar de una lista al presionar el botón de despliegue de lista mostrándose la siguiente ventana:



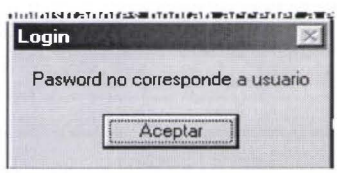
En esta ventana se desplegara solamente aquellos lotes que fueron ya dados de baja es decir que su estado esta en BA (Baja).
Una vez que se selecciona el lote regresamos a la ventana anterior en la cual se debe presionar el botón de impresora para obtener el siguiente reporte.



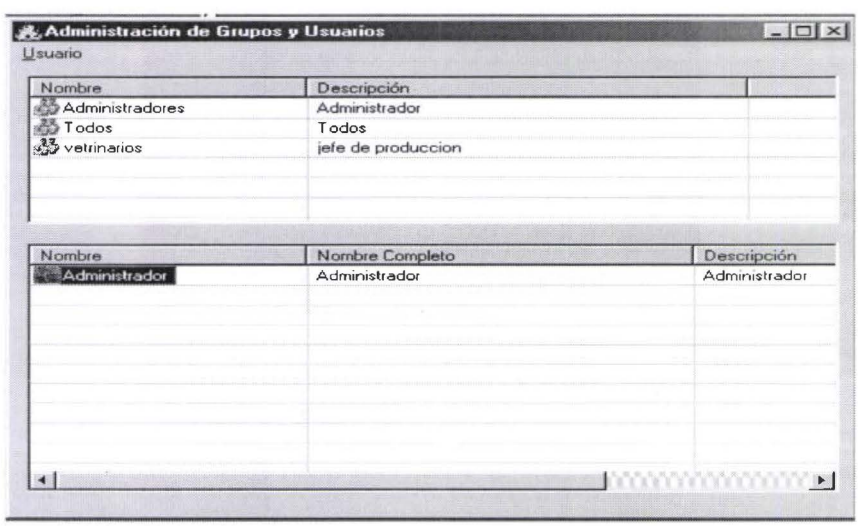
Seguridad.- Al escoger en la pantalla de ingreso la opción de Seguridad se presentará la siguiente ventana:



En esta ventana se ingresara el usuario y la clave del administrador del sistema no los usuarios administradores. Si uno de los dos esta incorrectos se desplegará la ventana de error

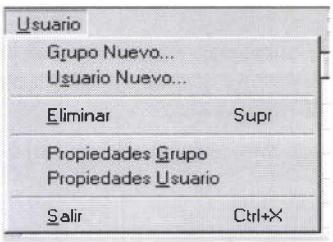


Caso contrario se desplegará la siguiente ventana:

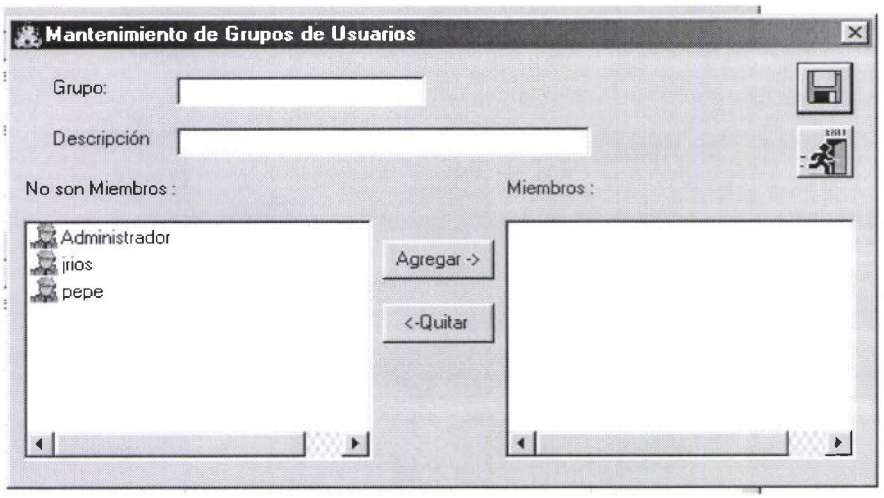


En esta ventana se visualiza los grupos de usuarios en la parte superior y en la parte inferior los usuarios de ese grupo.

Al dar un clip en el menú o la combinación de teclas "Alt+U" se desplegará la siguiente ventana:



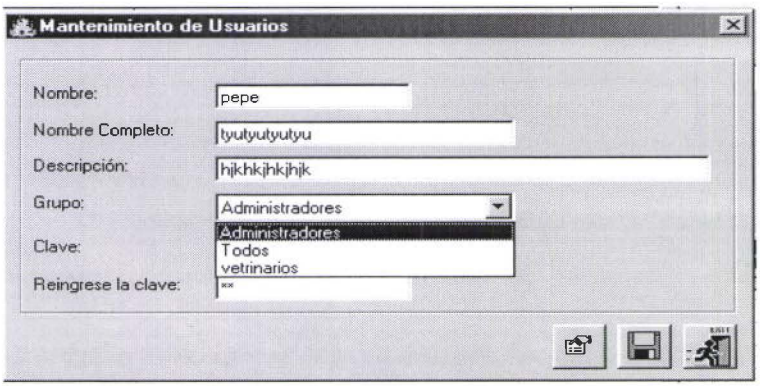
Grupo Nuevo.- Al seleccionar esta opción se crea un nuevo grupo de usuarios, y se presenta la siguiente ventana:



Grupo.- Nombre del Grupo, campo obligatorio.
Descripción.- Detallada del grupo, campo obligatorio

En la parte inferior izquierda se visualiza los usuarios que no son miembros del grupo y en parte derecha los usuarios que son miembros de ese grupo. Con el botón agregar se ingresa un usuario a ese grupo y con el botón Quitar se elimina un usuario de ese grupo.
El botón grabar graba los cambios. Y El de Salir, cierra la ventana.

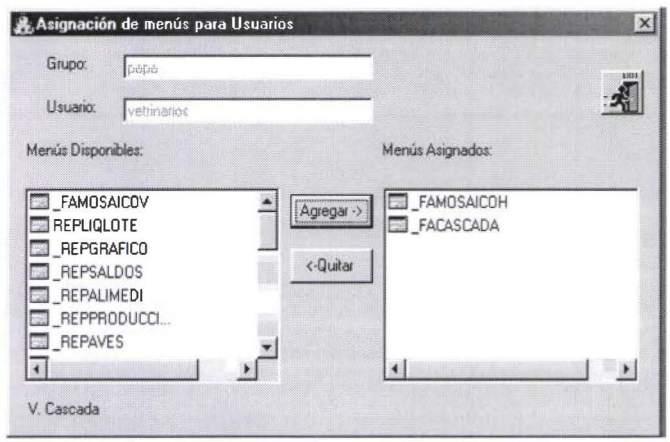
Usuarios Nuevo.- Esta opción permite crea nuevos usuarios en un grupo, se desplegará la siguiente ventana:



Nombre.- Identificación del usuario, campo obligatorio.
Nombre Completo.- Identificación del usuario, campo obligatorio.
Descripción.- Detalle descriptivo del usuario.

Grupo.- Selecciona el grupo al cual va a pertenecer el usuario.
Clave.- Ingrese la clave de usuario
Reingreso de Clave.- Ingrese nuevamente la clave del usuario para confirmar su ingreso.

Con el botón de Guardar almacena estos datos y se activa el botón de nuevo para ingresar las opciones permitidas a realizar por parte de este usuario. Entonces se desplegará la siguiente ventana:

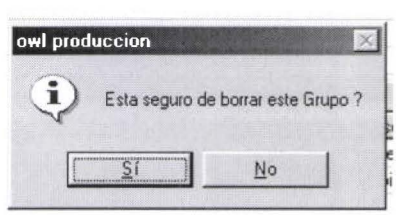


En la parte superior se desplegará el usuario y su nombre, en la parte inferior izquierda se tiene la lista los nombres de las opciones para ser seleccionadas ya sea dando doble clip en su nombre o marcándoles y luego presionando el botón de Agregar.

Para eliminar una opción a un usuario se selecciona la opción y se presiona el botón Quitar.

Al finalizar se presionar el botón de salir para abandonar y salir de esta pantalla.

Eliminar.- Para eliminar un grupo o un usuario se debe seleccionar el grupo o usuario a ser eliminado. Y confirmar o negar la interrogante desplegada, según sea el caso.

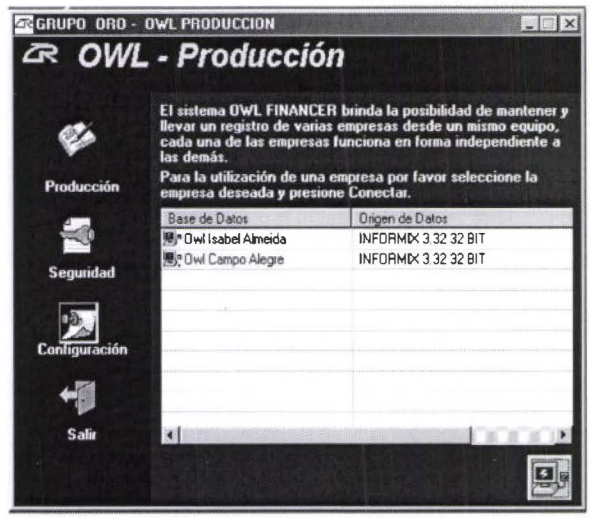


Propiedades de Grupo.- Permite agregar o quitar usuarios de un grupo en particular.

Propiedades de Usuario.- Permite cambiar las propiedades de un usuario en particular, es agrega o quita opciones a un usuario.

Salir.- Abandona el modulo de seguridad.

Configuración.- Al escoger en la ventana principal de la opción de configuración se desplegará la siguiente ventana la misma que me permite seleccionar en cual empresa se va a trabajar.



Para seleccionar la empresa en la que se va a trabajar, primero se selecciona la empresa y se presiona el botón que se encuentra en la parte inferior derecha de la ventana.