107
2EJ

Universidad Nacional Autónoma de México

Facultad de Ingeniería

Desarrollo de un sistema gráfico de captura ,
para redes de tanques y pozos
utilizando la Tecnología Orientada a Objetos

T E S I S

Que para obtener el grado de
Ingeniero en Computación

p r e s e n t a

Patricia Del Valle Morales

FALLA DE ORIGEN

Dir. Ing. Gabriel Castillo Hernández

México D. F *Enero de 1995*

TESIS CON
FALLA DE ORIGEN

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

A Dios

Por que siempre has permanecido a mi lado
Por que me has permitido llegar hasta donde he llegado, y por tantas cosas bellas que me has dado.

A Gabriel

Por tu amor
Por llenar mi vida de felicidad
Por que juntos alcanzaremos nuestras metas
Por que juntos disfrutaremos nuestros triunfos
Por que formas parte de mi vida .....

A mis padres

Por que me han dado la vida
Por que mis triunfos son sus triunfos.

A mis hermanos

Por permitirme formar parte de ustedes.

INDICE

A Dios

Por que siempre has permanecido a mi lado
Por que me has permitido llegar hasta donde he llegado, y por tantas cosas bellas que me has dado.

A Gabriel

Por tu amor
Por llenar mi vida de felicidad
Por que juntos alcanzaremos nuestras metas
Por que juntos disfrutaremos nuestros triunfos
Por que formas parte de mi vida .....

A mis padres

Por que me han dado la vida
Por que mis triunfos son sus triunfos.

A mis hermanos

Por permitirme formar parte de ustedes.

INDICE

INTRODUCCIÓN

En las últimas décadas, el desarrollo del país ha exigido un crecimiento sin precedentes en la infraestructura nacional. Entre las áreas de prioridad, se encuentra la construcción, operación y mantenimiento de sistemas de conducción y distribución de agua.

El agua es obtenida de dos fuentes principales: Las superficiales (ríos, lagos y presas) y las subterráneas (mantos freáticos). Los sistemas hidráulicos deberán de obtener el líquido y conducirlo desde el punto de extracción hasta las ciudades, que usualmente, se encuentran a una distancia que va de las decenas hasta los cientos de kilómetros. Una vez en la ciudad el agua debe ser distribuida. Los costos de explotación, conducción y distribución son altos, siendo sufragados por los usuarios, gobierno municipal, estatal y federal.

Sin embrago, los recursos hidráulicos del país son limitados, requiriendo, cada vez con mayor frecuencia, llevar el agua desde lugares más distantes. Por ello es necesario racionalizar su explotación y su distribución. Tratando de disminuir al máximo la cantidad de pérdidas por fugas.

Dentro de este contexto, es importante obtener modelos matemáticos que permitan predecir de manera más precisa el comportamiento del líquido. Un modelo matemático más preciso permitirá que el Ingeniero Civil sea capaz de diseñar sistemas hidráulicos más eficientes. Los organismos responsables podrán planear de mejor manera la forma de operar un acueducto y como distribuir el agua dentro de la ciudad.

La cantidad de información que requiere un modelo matemático es bastante elevada, por ejemplo: se requiere información precisa de la tipología del sistema de conducción y distribución, así como de cada uno de los tubos que forman el sistema, de cada una de las estructuras de control (tanques, torres de oscilación, cámaras de aire, tanques de regulación, válvulas, etc.) , y de los sistemas de extracción.

El Instituto de Ingeniería ha desarrollado desde hace años modelos numéricos que permiten diseñar y analizar el comportamiento de los sistemas de conducción y los problemas asociados a ellos. Sin embargo los archivos de datos que describen los sistemas que se desean estudiar son grandes y su elaboración a menudo requiere de gran cantidad de tiempo. Si el sistema se encuentra en la fase de

diseño, las alternativas posibles son muchas, y la cantidad de cambios que pueden requerir los archivos son también bastantes y a menudo implican la reconstrucción total de los mismos.

Se requiere por tanto, de un programa amigable en el cual se pueda capturar, actualizar o simplemente consultar la información en forma rápida y eficiente sin necesidad de navegar en la gran cantidad de números. Un sistema donde el usuario pueda pedir cualquier información que necesite de la red hidráulica en el momento que desee.

La presente tesis versa sobre el desarrollo de un sistema de captura de información para redes hidráulicas. Desde el punto de vista computacional, el sistema se basó en la aplicación de la Tecnología Orientada a Objetos.

El capítulo dos presenta los conceptos básicos de la tecnología orientada a objetos.

El capítulo tres versa sobre el análisis del sistema deseado.

El capítulo cuatro presenta las bases del Diseño orientado a objetos.

El capítulo cinco aplica los conceptos del capítulo cuatro al sistema que se desea desarrollar.
A manera de conclusión, en el capítulo seis se presentan algunos comentarios sobre el sistema desarrollado.

I. CONCEPTOS DE LA TECNOLOGÍA ORIENTADA A OBJETOS

I.1 Historia de la TOO

En el presenten capítulo se presentan los conceptos relacionados con la tecnología orientada a objetos. Actualmente no existe un estándar en tanto en los términos como en su significado, presentamos por ello los términos y significado más aceptado.

Desde el principio de los 90's, el desarrollo del software se retrasó respecto a las posibilidades ofrecidas por el hardware, algunos autores sostienen que este retrasó es de al menos dos generaciones de procesadores, y esta distancia continua aumentando. A pesar de los esfuerzos que se han hecho para construir mejores sistemas, la crisis del software sigue creciendo año con año. Han pasado más de 40 años desde que se invento la subrutina y se siguen construyendo los sistemas "a mano", es decir, una instrucción a la vez.

La velocidad a la que avanza la tecnología del hardware es sorprendente, año con año se logran avances que conducen a la construcción de computadoras más veloces, compactas y más baratas. A diferencia del software, no existe un desarrollo tan acelerado y su costo tiende a incrementarse día con día.

La construcción del software no es una tarea fácil y en muchas ocasiones los proyectos de programación sobrepasan los presupuestos de tiempo y costo. En caso de liberarse algún sistema, generalmente son ineficientes y están construidos con componentes únicos, no intercambiables y de manera rígida por lo que es casi imposible hacer modificaciones importantes. La mayoría del software corporativo es obsoleto, aun antes de ser liberado es incapaz de evolucionar. Muchas veces se invierte menos tiempo en generar un sistema nuevo que en modificar el ya existente.

Desde la década de los 70's, se han desarrollado varias metodologías para el desarrollo de sistemas, destacando los trabajos del enfoque estructurado (programación procedural) y el enfoque a datos (bases de datos).

Aunque estas metodologías han representado un gran avance, siendo efectivas durante largo tiempo,

es evidente que la velocidad de cambio que se requiere con respecto al hardware rebasan su capacidad.

Tomando las mejores ideas de estás metodologías, junto con otros nuevos conceptos, surge el paradigma orientado a objetos, con el fin de resolver estos problemas, dando una nueva visión de la tarea de análisis, diseño y programación de sistemas.

En lugar de tratar de modelar un problema en la computadora se trata ahora de acercar la computadora al problema. Es decir, modelar la realidad del problema a través de entidades independientes pero que interactúan entre sí y cuyas fronteras no estén determinadas por su instrumentación computacional sino por la naturaleza del problema. Estas entidades serán denominadas *objetos* por analogía con el concepto objeto en el mundo real.

I.2. Bases de la tecnología orientada a objetos

Para entender esta nueva tecnología, es necesario recordar como actúa la mente humana.

El hombre trata con su entorno abstrayendo de él lo que considera esencial para su entendimiento. Por ejemplo, pensemos en vehículos y en particular de bicicletas y motocicletas. En este contexto, al momento de hacer referencia a una bicicleta pensaremos probablemente en el tamaño de las llantas, si es de montaña o de carreras, si es de velocidades o no, etc. y muy rara vez nos preguntaremos por que color tendrá. En el caso de la motocicleta es posible pensar en si es de campo traviesa o de carreras, cuantos centímetros cúbico tiene de capacidad, la marca, etc. pero no pensaríamos en cual es la forma exacta del volante. Si además particularizamos hablando de las bicicletas (o de las motocicletas) de Juan, Pedro y Antonio, entonces tendremos información particular de cada una de ellas.

En los renglones anteriores hemos aplicado buena parte de los conceptos que el paradigma orientado a objetos utiliza, veamos los conceptos utilizados:

Clases, subclase y superclase. Hemos hablado de vehículos, de bicicletas y de motocicletas, podemos decir que las bicicletas y las motocicletas son subclases de vehículos, o que vehículos es la superclase (o clase superior) a la cual pertenecen tanto las motocicletas como las bicicletas. Además podemos afirmar que cada uno de los tres son clases de objetos reales o conceptuales, las cuales denotarán características esenciales que permiten al serhumano discriminar si un objeto en particular pertenece o no a una clase determinada.

Objeto (o instancia de una clase). Un objeto es un caso particular de una clase, el objeto cumple con todas las características que la clase a la que pertenece marca, dándole valores concretos a esas características, las cuales lo distinguen de otros objetos de la misma clase. Por ejemplo la bicicleta de Juan cumple con todas las características de la clase bicicletas, pero además es único (suponiendo que Juan solo tiene una bicicleta)

Encapsulamiento. El encapsulamiento permite ocultar ciertos aspectos de un objeto, que en el contexto en que es usado son irrelevantes. Por ejemplo si estamos hablando de que una bicicleta es capaz de transportar a una persona de 100 Kg en un trayecto de 10 Km, la función de transportación no se explica, porque resultaría ocioso hablar acerca de que para mover la bicicleta se tiene una mecanismo de dos engranes unidos por una cadena, uno de ellos esta sujeto a la rueda trasera y el otro a los pedales ....

Ahora se describirán los conceptos anteriores desde el punto de vista computacional, para poder

entender la orientación a objetos.

I.2.1. Modelo de un objeto

Se puede decir que existen cuatro elementos fundamentales sin los cuales un modelo no puede ser considerado orientado a objetos, estos elementos son:

. Abstracción
. Encapsulación
. Modularidad
. Jerarquía

Adicionalmente existen tres elementos deseables, pero no indispensables en un modelo orientado a objetos, estos elementos son:

. Tipificación
. Concurrencia
. Persistencia

Abstracción

La abstracción denota las características esenciales de un objeto. Esta permite distinguir un objeto en particular de otros tipos de objetos y por lo tanto proporciona fronteras conceptuales claramente definidas, relativas a la perspectiva del observador.
La abstracción se enfoca a ver al objeto desde fuera, lo que permite separar la tarea principal de un objeto de su implementación.

Encapsulamiento

La abstracción de un objeto debe preceder a las decisiones acerca de su implementación. Una vez que una implementación es seleccionada, ésta debe ser tratada como un secreto de la abstracción y debe permanecer oculta (encapsulada) para los clientes, permitiendo de esta manera que ninguna parte de un sistema complejo dependa de los detalles internos de cualquier otra parte.

"Encapsulación es el proceso de ocultar todos los detalles de un objeto que no contribuyen al entendimiento de sus características esenciales."

La abstracción y el encapsulamiento son conceptos complementarios. La abstracción permite al diseñador modelar el problema con el que se enfrenta, mientras que el encapsulado permite la realización de cambios con relativamente poco esfuerzo.

Modularidad

La modularidad es la propiedad de un sistema que ha sido descompuesto en un conjunto de módulos acoplados cohesionadamente.

Jerarquía

Jerarquizar u ordenar las abstracciones. Es posible encontrar muchas diferentes abstracciones y podemos entender solo una a la vez. Por lo que un conjunto de abstracciones a menudo forman una jerarquía, y al identificar estas jerarquías en nuestro diseño simplificamos altamente nuestro entendimiento del problema.

Existen tres tipos de jerarquizaciones (métodos de organización) que son: Identidad, parte de y tipo de.

1) *Identidad*. Identifica los objetos por medio de sus atributos y comportamiento. Cada objeto es único y diferente a cualquier otro. Por ejemplo un avión, una bicicleta, un auto, etc.

2) Relación: *"es del tipo de"*
   Describe al objeto en termino de sus semejanzas con otros. Todas las clases que son del tipo de otra clase tienen información o comportamiento común.
   Está jerarquía tiende a generar particularizaciones a partir de conceptos generales, por ejemplo un auto compacto es un tipo especial de automóvil, a su vez un VW es un tipo especial de auto compacto, etc. Esta jerarquía se le conoce como la *"estructura clase"*.

3) Relación: *"es parte de"*
   Esta relación explica al objeto en término de sus componentes. A este tipo de objeto que se forma a partir de otros objetos se le llama *"Objeto compuesto"*. Por ejemplo un coche consta de ruedas, motor, carrocería, etc. A esta jerarquía se le conoce como la *"estructura objeto"*.

Tipificación

El concepto de tipos deriva de las teorías de los tipos de datos abstractos. Un tipo es una caracterización precisa de las propiedades estructurales o funcionales, las cuales son compartidas por una colección de entidades.
Tipificación es la imposición de la clase de un objeto, tal que objetos de diferentes tipos no pueden ser intercambiados, solo podrán serlo en un sentido muy restringido.

Persistencia

La persistencia es la propiedad de un objeto por la cual éste existe transcendiendo en el tiempo (el objeto continua existiendo aun después de que su creador deja de existir) y/o el espacio (la localización del objeto se mueve de la dirección espacial en la que fue creado).

I.2.2. Programación orientada a objetos (POO)

La POO es un método de implementación en el cual los programas están organizados como colecciones cooperativas de objetos, cada uno de los cuales representan una instancia de alguna clase, y cuyas clases son todas miembro de una jerarquía unidas vía relaciones de herencia.

Dentro de la Programación orientada a objetos se manejan varios conceptos tales como clase, objeto, polimorfismo, herencia, etc. Sin embargo, resulta confuso hablar de ellos separadamente, por lo que se les agrupa en tres partes modulares:

1) Objetos
   Abstracción.
   Encapsulamiento.

2) Clases

Jerarquizar u ordenar las abstracciones. Es posible encontrar muchas diferentes abstracciones y podemos entender solo una a la vez. Por lo que un conjunto de abstracciones a menudo forman una jerarquía, y al identificar estas jerarquías en nuestro diseño simplificamos altamente nuestro entendimiento del problema.

Existen tres tipos de jerarquizaciones (métodos de organización) que son: Identidad, parte de y tipo de.

1) *Identidad.* Identifica los objetos por medio de sus atributos y comportamiento. Cada objeto es único y diferente a cualquier otro. Por ejemplo un avión, una bicicleta, un auto, etc.

2) Relación: *"es del tipo de"*
   Describe al objeto en termino de sus semejanzas con otros. Todas las clases que son del tipo de otra clase tienen información o comportamiento común.
   Está jerarquía tiende a generar particularizaciones a partir de conceptos generales, por ejemplo un auto compacto es un tipo especial de automóvil, a su vez un VW es un tipo especial de auto compacto, etc. Esta jerarquía se le conoce como la *"estructura clase"*.

3) Relación: *"es parte de"*
   Esta relación explica al objeto en término de sus componentes. A este tipo de objeto que se forma a partir de otros objetos se le llama *"Objeto compuesto"*. Por ejemplo un coche consta de ruedas, motor, carrocería, etc. A esta jerarquía se le conoce como la *"estructura objeto"*.

Tipificación

El concepto de tipos deriva de las teorías de los tipos de datos abstractos. Un tipo es una caracterización precisa de las propiedades estructurales o funcionales, las cuales son compartidas por una colección de entidades.
Tipificación es la imposición de la clase de un objeto, tal que objetos de diferentes tipos no pueden ser intercambiados, solo podrán serlo en un sentido muy restringido.

Persistencia

La persistencia es la propiedad de un objeto por la cual éste existe transcendiendo en el tiempo (el objeto continua existiendo aun después de que su creador deja de existir) y/o el espacio (la localización del objeto se mueve de la dirección espacial en la que fue creado).

I.2.2. Programación orientada a objetos (POO)

La POO es un método de implementación en el cual los programas están organizados como colecciones cooperativas de objetos, cada uno de los cuales representan una instancia de alguna clase, y cuyas clases son todas miembro de una jerarquía unidas vía relaciones de herencia.

Dentro de la Programación orientada a objetos se manejan varios conceptos tales como clase, objeto, polimorfismo, herencia, etc. Sin embargo, resulta confuso hablar de ellos separadamente, por lo que se les agrupa en tres partes modulares:

1) **Objetos**
   Abstracción.
   Encapsulamiento.

2) **Clases**

Herencia
Objetos compuestos

3) **Mensajes**
Polimorfismo
Sobrecarga de funciones
Sobrecarga de operadores
Funciones virtuales

Los conceptos abstracción y encapsulamiento ya fueron definidos con anterioridad.

I.2.3. Objeto

Los objetos representan una abstracción de algún elemento de la realidad. Los elementos de la realidad que representa el objeto puede provenir, principalmente de dos ramas:

1) Una cosa tangible o visible, tal como una mesa, un avión, un ferrocarril, etc.
2) Un concepto del pensamiento o un concepto generalizado, tal como una cola, una reacción química, un compilador, etc.

Como una definición informal podemos decir que un objeto representa un individuo, una cosa bien identificada, una unidad o una entidad ya sea real o abstracta con un papel bien definido en el dominio de un problema. El objeto consta de datos internos que representan su estado, y de métodos que representan su comportamiento. Ambos, forman parte modular del objeto y no se puede pensar en ellos en forma separada. Un objeto tiene cierta identidad porque es único y diferente a cualquier otro objeto.

En términos generales definimos un objeto como, cualquier cosa que posea una frontera ligeramente definida. Sin embargo no todo es un objeto, por ejemplo atributos tales como el tiempo, la belleza o el color no son objetos, ni tampoco las emociones tal como el amor o la ira; todas estas cosas son potencialmente propiedades de un objeto. Por lo que decir que un objeto posee una frontera definida no es suficiente.
Se sugiere la siguiente definición:

> *"Un objeto posee estados, comportamiento e identidad. La estructura y las operaciones de objetos similares son definidas en su clase común. El termino instancia y objeto son intercambiables."*

Estado de un objeto

El estado de un objeto se modela a través de un conjunto de valores variables, que representan la abstracción del objeto y sus características en el tiempo, a esto se le conoce como "datos".
En muchos casos el comportamiento de un objeto es influenciado por su historia; el orden en el que uno opera sobre él es importante, por ejemplo, en una maquina de refrescos no es lo mismo primero seleccionar y después pagar que pagar y después seleccionar, por lo tanto:

> *"El estado de un objeto incluye tanto sus propiedades (normalmente estáticas) como también los valores actuales (normalmente dinámicos) de cada una de esas propiedades."*

Los valores son normalmente dinámicos por que generalmente cambian con el tiempo, sin embargo existen casos en que esto no sucede; por ejemplo el número de serie de una máquina. Por otro lado una propiedad es una característica inherente o distintiva, rasgo, cualidad o una facción que contribuye

a hacer a un objeto único. Las propiedades tienden a ser estáticas por que sus atributos se mantienen normalmente sin cambios y son fundamentales en la naturaleza del objeto. Sin embargo bajo ciertas circunstancias las propiedades del objeto pueden cambiar, tal es el caso de los sistemas capaces de adquirir nuevos conocimientos o de ser autónomos.

Todas las propiedades tienen algún valor, esté puede ser una simple cantidad o puede denotar otro objeto. Así pues, podemos distinguir entre objetos y valores simples:

Las cantidades simples tal como el número tres son "atemporales, intercambiables y no instanciadas"; mientras que un objeto "*existe* en el tiempo, es *cambiante* (tiene un estado), es *instanciado* y puede ser *creado, destruido y compartido*"

Comportamiento de un objeto (tareas)

El comportamiento de un objeto es la forma en que un objeto actúa y reacciona en términos de sus cambios de estado y paso de mensajes; este comportamiento se modela a partir de un conjunto de funciones que llevan a cabo ciertas acciones o que modifican el estado del objeto.

Una operación es una acción que un objeto ejecuta sobre otro obteniendo una reacción.
En lenguajes tales como Samalltalk hablamos de que un objeto pasa a otro un mensaje. *Generalmente un mensaje es simplemente una operación que un objeto ejecuta sobre otro, aunque el mecanismo sea diferente*. Para nuestro propósito el término operación y mensaje son intercambiables. En los LOO las operaciones que los clientes pueden efectuar sobre un objeto son declaradas típicamente como "métodos" o "funciones miembro".

En la practica encontramos que un cliente efectúa cinco diferentes tipos de operaciones; las más comunes son:

a) Modificante
es una operación que altera el estado de un objeto.

b) Selectora
Una operación que accesa el estado de un objeto pero no lo altera; una operación lectora.

c) Iterator
Una operación que permite que todas las partes de un objeto sean accesadas en algún orden bien definido.

En lenguajes tales como Object Pascal, C + + y Samalltalk es posible declarar otras dos operaciones:

d) Constructora
Una operación que crea un objeto y/o inicializa su estado.

e) Destructora
Una operación que libera el estado de un objeto y/o lo destruye.

En lenguajes tales como Object Pascal, C + +, Clos y Ada es posible escribir operaciones como subprogramas libres, los cuales son procedimientos o subfunciones que sirven como operaciones no primitivas sobre objetos de la misma clase o de clases diferentes. Los subprogramas libres son agrupados típicamente de acuerdo a las clases sobre las cuales ellos son construidos, por lo que llamaremos a tales colecciones de subprogramas libres "*utilidades de clase*" (class utilities). Con lo que

podemos afirmar que "*todos los métodos son operaciones pero no todas las operaciones son métodos*".

Colectivamente todos los métodos y subprogramas asociados a un objeto en particular comprenden el *protocolo de ese objeto*.

Los objetos pueden ser *activos* o *pasivos*. Un objeto activo es el que comprende sus propios pasos de control, mientras que un pasivo no.

De esta manera los objetos activos del sistema sirven como las raíces de control. Si nuestro sistema implica múltiples pasos de control entonces tendemos múltiples objetos activos. (Los sistemas secuenciales solo tienen un objeto activo a la vez).

Los objetos activos son generalmente autónomos, lo cual significa que son capases de exhibir algún comportamiento sin ser operados por otro objeto. En cambio los objetos pasivos pueden tener un cambio de estado cuando se actúa explícitamente sobre ellos.

I.2.4. Clases

Mientras que un objeto es una entidad que ejecuta alguna labor sobre todo el sistema, la clase captura la estructura y comportamiento común a todos los objetos similares.
Así mismo, mientras un objeto es la entidad concreta que existe en el tiempo y el espacio, una clase representa solamente la abstracción, la "esencia" de un objeto. Podríamos decir que:

> "*Una clase permite construir una taxonomía de objetos en un nivel abstracto y conceptual. Es una especificación genérica para un número arbitrario de objetos similares. Una clase puede ser vista como un patrón para un tipo específico de objeto.*"

Las clase definen las características comunes de un grupo de objetos. *Un objeto es simplemente la instancia (caso) de una clase*. Como tal, es una combinación de las características que comparten varios objetos, expresados dentro de la clase, y de un estado único representado por los valores que tomen los datos de la instancia.

NOTA: Un objeto no es una clase, una clase representa solo su abstracción (la esencia del objeto), así pues los objetos se crean mediante la instanciación de una clase. Una analogía con los lenguajes de programación orientados a procedimientos, se dice que la variable A es de tipo entero, entonces en términos de la POO se dice que el objeto A es una instancia de la clase entero.

Cada clase esta compuesta por dos partes importantes la interfase y la implementación:

- *La interfase de la clase*. Captura solamente su vista exterior abarcando nuestra abstracción del proceso común a todas las instancias de esa clase, esta constituida por los métodos públicos de esa clase.
  Es el lugar donde aseveramos todas las suposiciones que un cliente puede hacer acerca de cualquier instancia de la clase.

- *La implementación de la clase*. Son los adentros, comprende la representación de la abstracción así como la del mecanismo para obtener el proceso deseado. La implementación de la clase encapsula detalles a cerca de los cuales ningún cliente puede hacer suposiciones.

Superclase

Una clase que puede heredar comportamiento a otra clase es llamada *"superclase"*, *"clase padre"* o *"clase base"*

Subclase

La clase que hereda comportamiento de otra clase (u otras clases) y añade además comportamiento propio para definir su tipo de objeto único es llamada *"subclase"*, *"clase hijo" o "clase derivada"*

Las clases traen orden a los objetos organizándolos en jerarquías de clase: Herencia y Objetos compuestos

I.2.5. Herencia

La herencia es el más importante tipo de jerarquía y es un elemento esencial en los sistemas orientados a objetos. Básicamente la herencia define relaciones entre clases, donde una clase comparte la estructura o función definida en una (herencia simple) o más (herencia múltiple) clases.
La herencia representa entonces una jerarquía de abstracciones en la cual una subclase hereda de una o más superclases.

Podemos decir que la herencia es una jerarquización que utiliza la organización "tipo de". Este mecanismo permite definir una clase añadiendo características propias (datos y métodos) a una clase ya existente. Dado que una subclase hereda datos y métodos de una superclase, las jerarquizaciones siempre deben ser de lo general a lo particular.

En la herencia las clases se definen como casos especiales de otras clases, formando así lo que llamamos una *"jerarquía de clases"*.

La herencia es un mecanismo muy eficiente, por que cada variable y cada método se definen una sola vez, en el nivel más general que se aplique. Es por lo cual la herencia es la base principal en la reutilización de código dentro de la POO

Herencia múltiple

Cuando una subclase hereda solo de una superclase se le llama *"herencia simple"*. No obstante, es posible que una clase herede información de más de una clase, a esto se le conoce como *"herencia múltiple"*.

I.2.6. Objetos compuestos

La jerarquización "parte de" es referida por medio de los objetos compuestos. Esta jerarquización es la que hacemos al distinguir un objeto y las partes que lo componen. La composición permite definir una nueva clase de objetos mediante la unión de un conjunto de clases ya existentes.

Gran parte del poder de los objetos consiste en que pueden contener también otros objetos. En la mayoría de los sistemas, los objetos compuestos no "contienen" literalmente otros objetos, si no que contienen datos que apuntan a la referencia de ese objeto.

Superclase

Una clase que puede heredar comportamiento a otra clase es llamada *"superclase"*, *"clase padre"* o *"clase base"*

Subclase

La clase que hereda comportamiento de otra clase (u otras clases) y añade además comportamiento propio para definir su tipo de objeto único es llamada *"subclase"*, *"clase hijo" o "clase derivada"*

Las clases traen orden a los objetos organizándolos en jerarquías de clase: Herencia y Objetos compuestos

I.2.5. Herencia

La herencia es el más importante tipo de jerarquía y es un elemento esencial en los sistemas orientados a objetos. Básicamente la herencia define relaciones entre clases, donde una clase comparte la estructura o función definida en una (herencia simple) o más (herencia múltiple) clases.
La herencia representa entonces una jerarquía de abstracciones en la cual una subclase hereda de una o más superclases.

Podemos decir que la herencia es una jerarquización que utiliza la organización "tipo de". Este mecanismo permite definir una clase añadiendo características propias (datos y métodos) a una clase ya existente. Dado que una subclase hereda datos y métodos de una superclase, las jerarquizaciones siempre deben ser de lo general a lo particular.

En la herencia las clases se definen como casos especiales de otras clases, formando así lo que llamamos una *"jerarquía de clases"*.

La herencia es un mecanismo muy eficiente, por que cada variable y cada método se definen una sola vez, en el nivel más general que se aplique. Es por lo cual la herencia es la base principal en la reutilización de código dentro de la POO

Herencia múltiple

Cuando una subclase hereda solo de una superclase se le llama *"herencia simple"*. No obstante, es posible que una clase herede información de más de una clase, a esto se le conoce como *"herencia múltiple"*.

I.2.6. Objetos compuestos

La jerarquización "parte de" es referida por medio de los objetos compuestos. Esta jerarquización es la que hacemos al distinguir un objeto y las partes que lo componen. La composición permite definir una nueva clase de objetos mediante la unión de un conjunto de clases ya existentes.

Gran parte del poder de los objetos consiste en que pueden contener también otros objetos. En la mayoría de los sistemas, los objetos compuestos no "contienen" literalmente otros objetos, si no que contienen datos que apuntan a la referencia de ese objeto.

Ventajas de los objetos compuestos:

- Los objetos contenidos cambian más fácilmente, tanto en tamaño como en composición sin afectar la estructura del objeto que lo contiene. Esto hace que el mantenimiento de los sistemas que utilizan este anidamiento sea más eficiente.

- Los objetos contenidos pueden ser usados por cualquier número de objetos compuestos, en vez de estar ligados solo a un objeto, evitando así la duplicidad.

- Los objetos contenidos en objetos compuestos pueden a su vez ser objetos compuestos, y este anidamiento puede llegar a cualquier nivel. Se pueden construir estructuras tan complejas como se deseen.

I.2.7. Mensaje

Un mensaje es, simplemente el requerimiento que hace un objeto a otro, para que el segundo ejecute uno de sus métodos.

Al objeto que envía el requerimiento se le llama "*objeto transmisor*" (cliente) y al objeto que lo recibe se le llama "*objeto receptor*" (servidor).La interfase entre el objeto transmisor y el receptor debe ser bien definida.

Estructuralmente un mensaje consta de tres partes:

1) La identidad del objeto receptor.
2) El nombre del método que se desea ejecutar.
3) El conjunto de parámetros de ese método.

I.2.8. Polimorfismo

- El polimorfismo es la habilidad de dos o más clases de objetos de responder al mismo mensaje, cada uno en su propia forma.

- El polimorfismo es el proceso por el cual se pueden acceder a diferentes implementaciones de una función con el mismo nombre.

Para que el polimorfismo exista, deberá existir más de un método con el mismo nombre en distintos objetos (esto es posible dado que los objetos son independientes entre si).

El fin del polimorfismo es permitir el uso de un nombre para especificar una clase de acción general

Existen tres formas de para conseguir el polimorfismo:

- Sobrecarga de funciones.
- Sobrecarga de operadores.
- Funciones virtuales.

Sobrecarga de funciones

- Dos o más funciones pueden compartir el mismo nombre siempre que su declaración de parámetros sea diferente. En esta situación las funciones que comparten el mismo nombre se dice que están sobrecargadas. La sobrecarga de funciones ayuda a manejar la complejidad.

- También se el conoce como sobrecarga de funciones a la redefinición de los operadores o de funciones del compilador. En esta situación las funciones u operadores que están redefinidos se les dice que están sobrecargados.

Sobrecarga de operadores

La sobrecarga de operadores de los lenguajes que son orientados a objetos se les puede dar significados especiales con relación a clases especificas. Cuando se sobrecarga un operador no se pierde ninguno de sus significados previos, simplemente significa se ha definido una nueva operación con relación a una clase especifica.
Se debe crear una clase que defina el tipo de datos sobre los que trabajará el operador sobrecargado.

Funciones virtuales.

Una función virtual, es una función que se declara en la clase base (superclase) como virtual y se redefine en sus clases derivadas (subclases).

Las funciones virtuales son especiales por que cuando se accede a una de ellas, usando un apuntador de su superclase a un objeto de la subclase, se determina en tiempo de ejecución a que función llamar, en función del tipo de objeto apuntado.

I.2.9. Análisis orientado a objetos (AOO)

El AOO es un método de análisis que examina los requerimientos desde la perspectiva de clases y objetos basados en el vocabulario del dominio del problema.
Se propone una solución del sistema, esta propuesta se hace en papel, describiendo ampliamente el bosquejo preliminar del sistema.

I.2.10. Diseño orientado a objetos (DOO)

El DOO es un método de diseñar abarcando los procesos de descomposición orientada a objetos y una notación, para representar modelos tanto lógica como físicamente, así como también estáticos como dinámicos, del sistema bajo diseño.

II. ANÁLISIS ORIENTADO A OBJETOS (AOO)

El análisis orientado a objetos es un método que examina los requerimientos desde la perspectiva de clases y se propone una solución del sistema (planteamiento de la solución), esta propuesta se hace en papel, describiendo ampliamente el bosquejo preliminar del sistema.

II.1. Planteamiento del problema

El grupo de Hidromecánica del Instituto de Ingeniería de la UNAM diseña acueductos, efectúa mediciones y pruebas, proponiendo soluciones a problemas en acueductos. Actualmente para realizar estas tareas se apoyan en los paquetes de computo que se han realizado en el mismo Instituto, pero aun no cuentan con un programa que capture la información de dichos acueductos y genere, en forma automática, los archivos de datos que requieren dichos paquetes.
Actualmente a través de un editor de textos se realiza la captura de datos numéricos, por lo que, cuando se tiene que verificar algún dato de la red hidráulica o las características de algún dispositivo hidráulico es muy engorroso, tardado y complicado realizar la búsqueda de dicho dato.
Por esta razón el grupo de Hidromecánica requiere de un programa amigable en el cual se pueda capturar, actualizar o simplemente consultar la información en forma rápida y eficiente sin necesidad de navegar en la gran cantidad de números, así pues, el usuario podrá pedir cualquier información que necesite de la red hidráulica en el momento que desee, bastará con especificárselo al programa.

Los dispositivos hidráulicos de los cuales se requiere información son:

a) Bomba. Los datos de una bomba son el nombre, la elevación, el gasto máximo valido, a_0, a_1 y a_2 que son los coeficientes del polinomio que la modela:

$$a_0 + a_1 Q + a_2 Q^2 = H(Q)$$

b) Línea de conducción. Los datos de la línea de conducción son el nombre, longitud, diámetro, fricción y celeridad.

c) Nudo. Los datos del nudo son el nombre y la elevación.

d) Tanque. Los datos del tanque son el nombre, elevación y el nivel.

El programa debe generar un archivo de datos con un formato especifico, para que posteriormente con este archivo se realice el modelado, simulación y el análisis de resultados del acueducto en otros programas de computo.

Infraestructura con la que se cuenta

a) PC's 386 y 486:
. Monitor super VGA color
. 33 MH y 66 MH en disco duro
. 4MB y 8MB en RAM

b) Lenguajes de desarrollo disponibles:
. Borland Pascal 7
. Borland C 3.1

II.2. Planteamiento de la solución

La interfase gráfica para la captura de datos de sistemas hidráulicos será un programa mediante el cual el usuario pueda realizar la captura de información para los distintos dispositivos hidráulicos (tanques, bombas, líneas de conducción, nudos, etc.) que conforman una red hidráulica y almacenarlos en un archivo de datos. Este programa consistirá de una pantalla de trabajo, un menú de comandos, un menú de colores, un menú de íconos y una zona donde se despliegan las coordenadas de la pantalla de trabajo. El programa será desarrollado con tecnología orientada a objetos en Borland Pascal 7.
La presentación de estos elementos se muestra en la figura[1] II.1.

Para la interacción con el usuario se usará el ratón y cajas de diálogos con botones de lectura, botones de título, botones con imágenes, botones de mensajes, etc.

La mecánica para seleccionar cualquier botón es:

1. El usuario deberá colocar el ratón dentro del botón a seleccionar.
2. Presionar una vez el botón izquierdo del ratón.

Para cancelar la ejecución de cualquier proceso deberá presionar el botón derecho del ratón o la tecla de escape "ESC".

▸Pantalla de trabajo

La pantalla de trabajo es la zona en donde se podrá dibujar la red hidráulica, para poder realizar esto, se cuenta con un menú de íconos que representan los dispositivos hidráulicos; el usuario podrá seleccionar un ícono a través del ratón y arrastrarlo hasta la pantalla de trabajo para ir formando dicha red, la mecánica para realizarlo se describe a continuación:

---

1 Las gráficas aquí presentadas son las desarrolladas finalmente como parte del sistema. Obviamente en un inicio fueron simplemente bosquejos de lo que se deseaba.

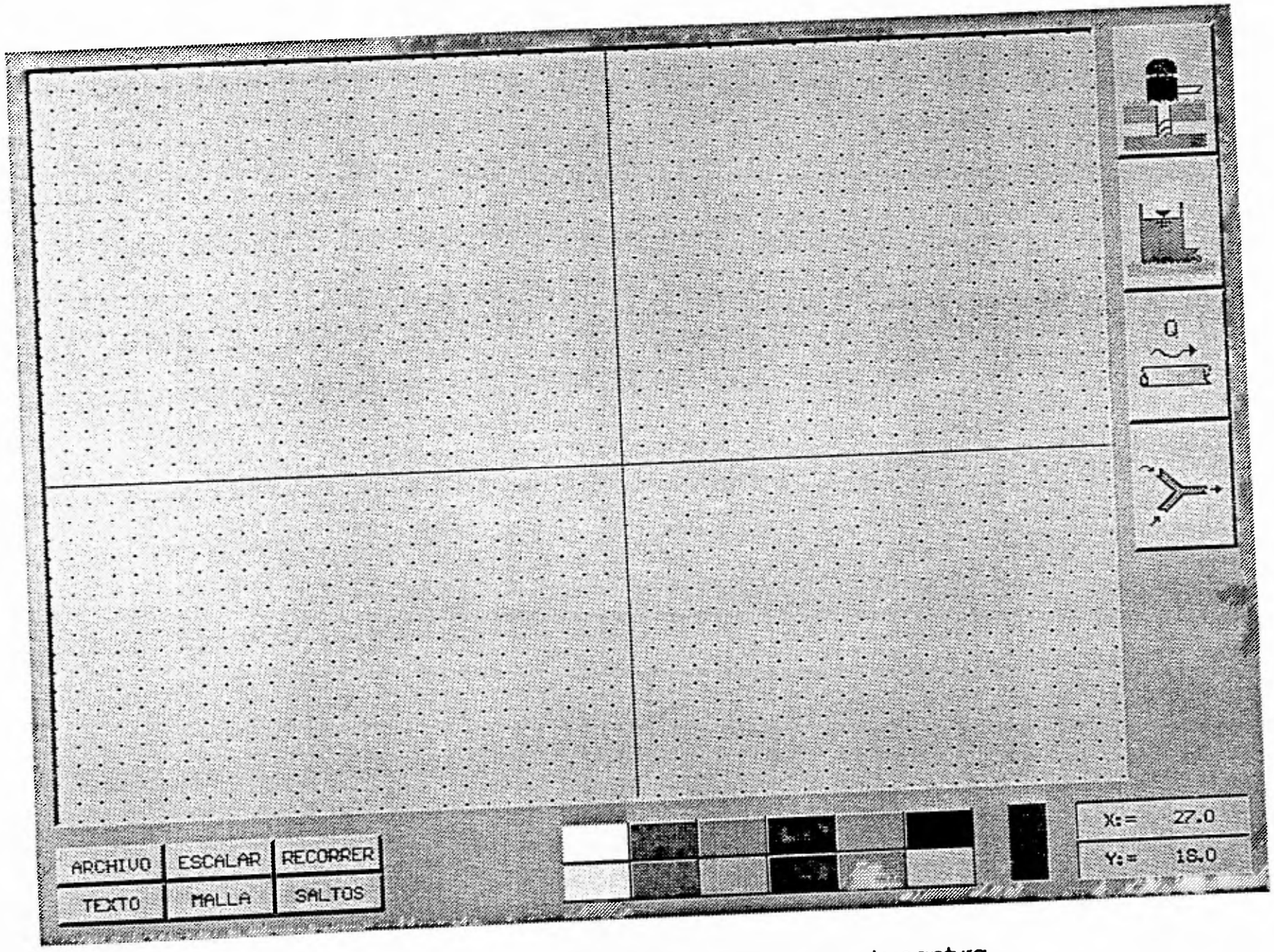


FIGURA II.1. Pantalla principal del sistema de captura

1. Seleccionar el botón con el ícono del dispositivo deseado.
2. Sin dejar de presionar el botón izquierdo del ratón arrastrar el dispositivo hasta la posición deseada en la pantalla de trabajo.
3. Soltar el botón izquierdo del ratón para que se dibuje el dispositivo en la posición seleccionada.

▸Coordenadas de la pantalla de trabajo

Las coordenadas de la pantalla de se despliegan en la esquina inferior derecha, así pues, cada vez que el ratón entra a la zona de trabajo las coordenadas se van actualizando tomando nuevos valores dependiendo de la posición del ratón.

▸ Cajas de diálogo y tablas de datos

Cada dispositivo tendrá una o varias cajas de diálogo (figura II.2) y una o varias tablas de datos (figura II.3), una vez que se ha dibujado un nuevo dispositivo en la pantalla de trabajo el usuario proporcionará la información correspondiente. Los datos de un dispositivo se podrán consultar y actualizar. Para realizar esto se seleccionará, con el ratón, el dispositivo del cual se desea consultar su información, presionando dos veces el botón izquierdo del ratón. Se abrirán sus cajas de diálogo y sus tablas de datos en la pantalla; de esta manera el usuario podrá moverse a través de ellas para actualizar la información o simplemente consultarla.

Para modificar la información de una tabla, el usuario deberá mover el ratón al renglón y columna donde se quiere registrar los datos y presionar el botón izquierdo del ratón.
Todas las tablas tendrán un encabezado de columna, un número de renglón (opcional), un menú de comandos con las siguientes funciones: adicionar, insertar, borrar, mover y continuar. Así como también tendrán cuatro funciones de desplazamiento representadas por dos barras verticales y dos flechas respectivamente, que son: avance de página, regreso de página, avanzar un renglón y regresar un renglón; estas funciones permitirán al usuario desplazarse a lo largo de la tabla. El usuario podrá moverse con el ratón solamente en la zona visible de la tabla de datos.

Comandos de la tabla de datos:

Adicionar: Adiciona un nuevo renglón al final de la tabla. Para ello el usuario deberá seleccionar el botón "adicionar".

Borrar: Borra un renglón de la tabla, actualizandola automáticamente. El usuario deberá seleccionar el botón "borrar", seleccionar el renglón que se desea borrar presionando el botón izquierdo del ratón dos veces.

Insertar: Inserta un nuevo renglón en la tabla, actualizandola automáticamente. El usuario deberá seleccionar el botón "insertar", seleccionar el renglón en donde se quiere insertar presionando dos veces el botón izquierdo del ratón (el usuario podrá desplazarse a lo largo de la tabla con los botones de desplazamiento de la tabla o bien moviendo el ratón en la zona visible de la tabla, para buscar el renglón en donde se quiere insertar el nuevo renglón)

Mover: Mueve un renglón a otra posición de la tabla, actualizandola automáticamente. El usuario deberá seleccionar el botón "mover", seleccionar el renglón que se desea mover presionando dos veces el botón izquierdo del ratón, seleccionar el renglón donde se desea insertar presionando dos veces el botón izquierdo del ratón.

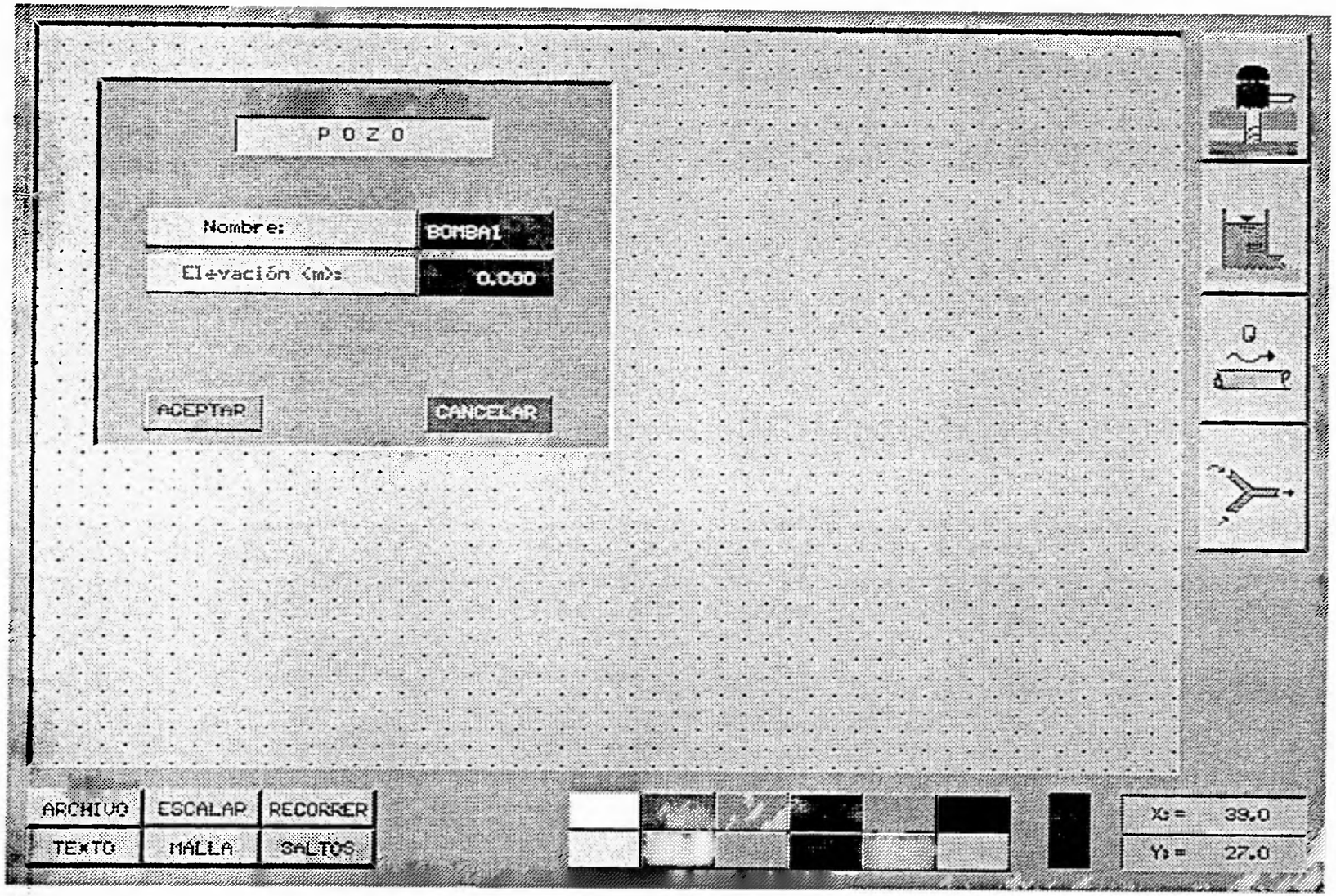


FIGURA II.2. Caja de diálogo

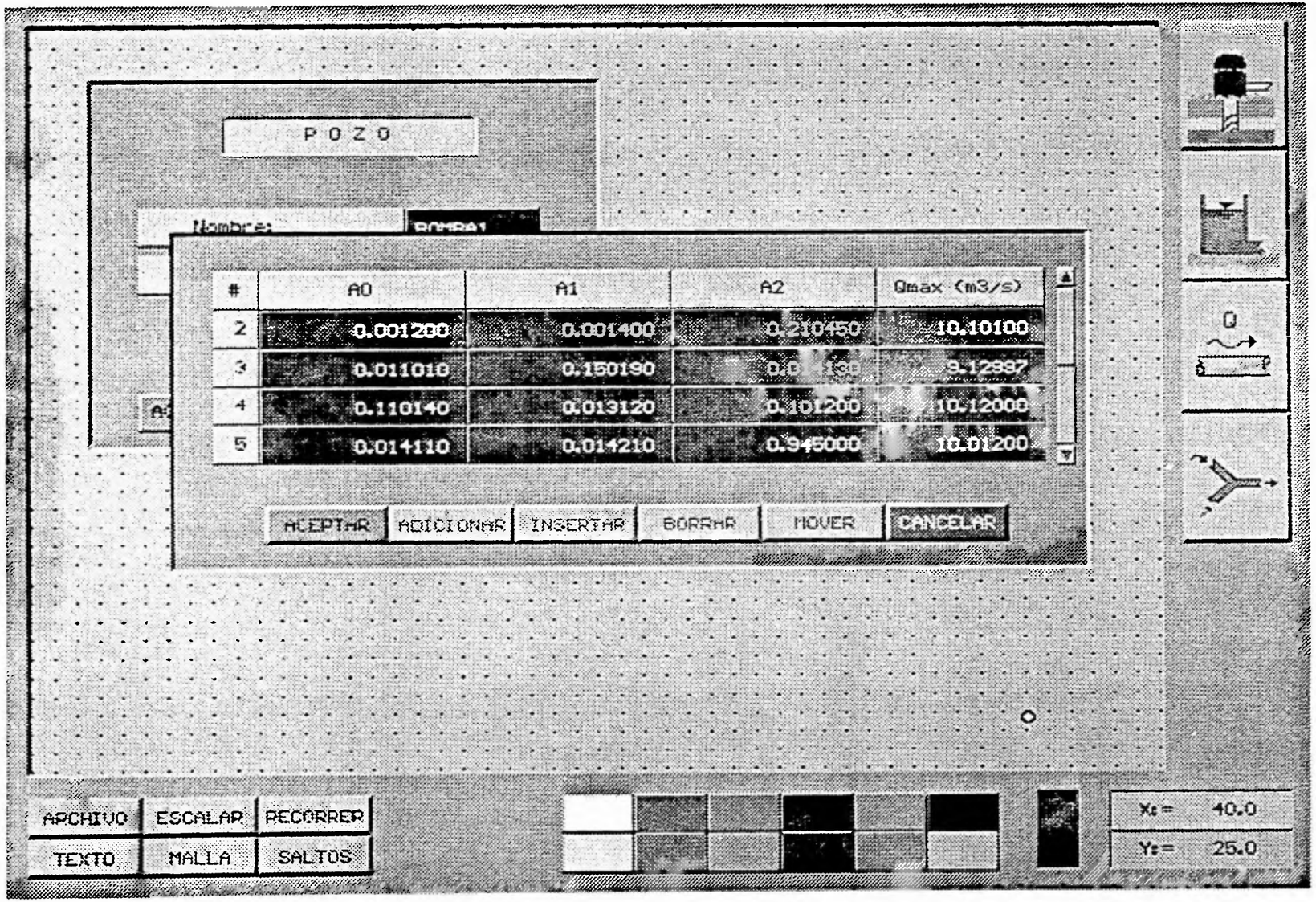


FIGURA II.3. TABLA DE DATOS.

Terminar: salir de la tabla.

Para interrumpir el proceso de alguno de estos comandos el usuario deberá presionar el botón derecho del ratón.

A diferencia de una tabla, una forma tiene botones donde se captura la información que de alguna manera no debe agruparse como una tabla, tal es el caso de los tanques. Un tanque se define con base en su elevación y tirante de agua respecto al nivel del piso, siendo estos datos únicos, (en el sentido de que un tanque no tiene varias elevaciones o inicialmente no se definen varios tirantes) La manera más adecuada de capturar esta información es lo que hemos llamado una caja de diálogo.

▸Menú de colores

El menú de colores permitirá al usuario cambiar el color de dibujo en el momento que desee, para hacer esto, el usuario deberá seleccionar el color que desea utilizar. A la derecha del menú de colores se encuentra un botón de color, ese botón indicará al usuario cual es el color actual de dibujo, este botón cambiará su color cada vez que se seleccione un color de este menú.

▸Menú de comandos

El menú de comandos permitirá al usuario realizar operaciones en la pantalla de trabajo y en la red hidráulica. Estos comandos son: archivo, borrar, copiar, mover, escalar, texto, ayuda, corrimiento.

Archivo: Proporcionará un menú con las siguientes opciones:

a) Crear. Creará un nuevo archivo de datos.
b) Recuperar. Recuperará un archivo de datos.
c) Salvar. Grabará el archivo de datos en uso.
d) Salvar como. Grabará el archivo de datos en uso con otro nombre.
e) Salir del sistema. Terminará con la ejecución del sistema.

Escalar: Hará un acercamiento y/o alejamiento de la pantalla modificando la escala de todo lo que se encuentre en la pantalla de trabajo. Cuando el usuario elige esta opción se abre un menú de escalamiento con los siguientes factores de escalación:

a) x2 y x5:
Estas opciones hace un acercamiento de 2:1 y de 5:1 respectivamente.

b) ÷2 y ÷5:
Estas dos opciones hacen un alejamiento de 1:2 y 1:5 respectivamente.

Para realizar un escalamiento, el usuario deberá seleccionar el botón "escalar" y seleccionar alguna de las opciones de escalamiento.

Malla: El botón "malla" dibujará una malla de puntos equiespaciados con un incremento de (1,1), esta malla es una referencia para la localización del dibujo. El usuario podrá activar la malla simplemente seleccionando el botón con el ratón y para desactivarla deberá presionar (seleccionar) el botón nuevamente.

Recorrer: Permitirá al usuario recorrer el dibujo a otro punto de la pantalla de trabajo. El usuario deberá seleccionar el botón "recorrer" y teclear la nueva posición (x,y) donde se

quiere recorrer la imagen.

Saltos: El botón "saltos" forzará a que el cursor haga movimientos discretos dentro de la pantalla de trabajo, es decir cada vez que el usuario mueva el ratón dentro de la pantalla de trabajo, este se moverá en saltos unitarios relativos a la escala del dibujo. El usuario podrá activar los saltos simplemente seleccionando el botón con el ratón y para desactivarlos deberá presionar (seleccionar) el botón nuevamente.

Texto: Permitirá poner texto en la pantalla de trabajo. Para realizarlo el usuario deberá:

a) Seleccionar el botón "texto".
b) Se abrirá una ventana donde el usuario tecleará el texto.
c) Al terminar de escribir el texto deberá de presionar la tecla "enter".
d) Seleccionará con el ratón la posición en donde se desea colocar el texto.

III. DISEÑO ORIENTADO A OBJETOS

III.1 ¿Que es el diseño orientado a objetos?

El diseño orientado a objetos es un método de diseñar abarcando los procesos de descomposición orientada a objetos para representar modelos tanto lógica como físicamente, así como también, estáticos como dinámicos del sistema bajo diseño.

Durante el proceso de diseño las especificaciones de requerimientos son llevadas a una especificación detallada de los objetos que intervienen, esta incluye una descripción completa de las responsabilidades y papel de cada objeto, así como de como se comunica cada objeto con el resto de los objetos.

Sobre el proceso de diseño:

- El resultado de este proceso no es un producto final. Aún después de que una aplicación, armazón o componente sea implementada, probada y vendida; estos pueden volverse a revisar una y otra vez, incluso antes de la implementación los diseñadores revisan reiteradamente decisiones anteriores y rediseñan parte del trabajo.

- El proceso de diseño nunca puede ser rígido, ni se puede considerar constituidos por reglas de aplicación estrictamente obligatorias. Aunque el diseño requiere rigor y disciplina, siempre queda un lugar para la inspiración, para el "arte" y no debemos perder el sentido estético como una guía.

III.2. PROCESO DE DISEÑO

1. Encontrar las clases del sistema.

   El objetivo es crear clases de objetos que modelen el dominio de la aplicación.
   Procedimiento para la búsqueda de clases:

   - Leer cuidadosamente las especificaciones de los requerimientos, buscando las frases

nominales, cambiando los plurales a singulares y construyendo una lista con ellos.

- En la lista aparecen clases obvias, otras frases que claramente no constituyen clases y frases sobre las que no podemos asegurar nada. Se desechan las frases que obviamente no son clases y se dejan las otras dos categorías como candidatas a ser clases.

- No todas las frases candidatas serán clases. Algunas candidatas a clases serán desechadas y aparecerán clases que no se encontraban entre las candidatas.

- Para seleccionar las clases existen además del sentido común que fundamentalmente trata de buscarle un sentido u objetivo a la clase candidata. Algunos lineamientos guías para la selección se muestran a continuación:

  . Modelos de objetos físicos.
  . Modelos de entidades conceptuales.
  . Si mas de una palabra se utiliza para un mismo concepto se selecciona la más significativa para el resto del sistema.
  . Ser prudente con los calificativos, ya que un calificativo puede denotar un objeto diferente, un uso diferente de un mismo objeto o bien ser irrelevante.
  . Ser prudente con los sujetos que no forman parte el sistema y con las oraciones en voz pasiva que tiene sujetos implicados.
  . Modelos de categorías de clases.
  . Modelos de interfases escondidas.
  . Modelos de valores de atributos de los objetos, pero no de los propios atributos.

Búsqueda de clases abstractas

- Se identifican las clases abstractas que deriven superclases de otras clases, para ayudar a identificar la estructura de la aplicación.

- Las clases abstractas se obtienen de clases que comparten atributos útiles, es decir que tienen asociado comportamiento compartido por ellas. El comportamiento compartido se captura en la superclase abstracta, entonces las subclases heredan dicho comportamiento. El objetivo es encontrar tantas clases abstractas como sea posible, buscando atributos comunes de las clases según se describa en la especificación de los requerimientos.

Grupos de clases

Identificar candidatas para superclases abstractas agrupando clases relacionadas. Una clase puede aparecer en más de un grupo. Cuando se ha identificado un grupo se nombra la superclase que lo debe representar con un simple nombre o con una frase nominal.

Análisis de relaciones entre clases

- Relación: "*es del tipo de*"
  Es frecuente un signo de la relación subclase-superclase entre ellas.
  Todas las clases que son del tipo de otra clase tienen información o comportamiento común. Encontrar estas relaciones nos permite asignar responsabilidades compartidas a superclases, de donde son heredadas por las subclases.

- Relación: "*es parte de*"
  Un objeto se compone de muchas instancias de otras clases, pero no se comporta como ellas. Una clara distinción entre objeto y partes nos ayuda en determinar a quién asignar las responsabilidades. Un objeto es responsable también por conocer los objetos que lo componen.

- Relación: *"es análoga a"*
  Cuando varias clases parecen análogas, ello es frecuente un signo de que comparten una superclase común. Ello nos permite identificar superclases no encontradas aún y asignar las responsabilidades comunes a dichas superclases.

Finalmente se crea una jerarquía de clases, así mismo se crean tarjetas (registro) de clases registrando el nombre de la clase, su superclase y sus subclases, y al reverso de cada tarjeta se describe en forma brevemente el objetivo de dicha clase.

2. Determinar que operaciones son responsabilidad de cada clase y que conocimiento deben tener para ello.

Las responsabilidades transmiten el sentido del propósito de un objeto y su lugar en el sistema. Las responsabilidades incluyen:

- El conocimiento que un objeto mantiene.
- Las acciones que un objeto puede manejar.

Las responsabilidades representan solo servicios disponibles públicamente. En este nivel las responsabilidades se definen independientes de la implementación (lo importante es que se hace y no como se hace).

Identificación de responsabilidades

- Leer cuidadosamente las especificaciones de requerimientos, buscando los verbos e informaciones que aparecen y determinando cuales representan acciones o conocimientos que pertenecen claramente a un objeto.

- Recorrido del sistema: partiendo de como el sistema es invocado y pasado por varios escenarios probando tantas capacidades del sistema como sea posible.

- A partir de las clases: Al identificar una clase se le detecto al menos un propósito o responsabilidad. El nombre de la clase sugiere esa responsabilidad y la oración que expresa el propósito de cada clase puede ayudar a encontrar otras responsabilidades.

Asignación de responsabilidades

Cuando no es evidente a que clase pertenecen las responsabilidades encontradas se pueden tener en cuenta las siguientes guías:

- Distribuir uniformemente la inteligencia del sistema.
  Cuando se minimiza el número de clases inteligentes en caso extremo a una sola, se obtiene un sistema con el control centralizado en un objeto inteligente, comportandose similar a un sistema procedimental, donde el objeto inteligente realiza la misma tarea que el módulo

principal de un programa procedimental y los otros objetos se comportan como estructuras de datos tradicionales.
Cuando la inteligencia del sistema se distribuye lo más uniformemente posible, se necesita más estudio para comprender el comportamiento del sistema, pero permite que cada objeto tenga que conocer sobre menos asuntos, produciendo un sistema más flexible, más fácil de modificar y que requiere menos objetos para su implementación.

- Expresar las responsabilidades lo más generalmente posible. Expresando las responsabilidades en términos generales, podemos encontrar más fácilmente responsabilidades compartidas por varias clases y llevar estas a superclases,de donde son heredadas por las otras clases.

- Mantener unidos el comportamiento y la información relacionada. Si un objeto es responsable por mantener cierta información, él debe tener asignada también la responsabilidad de las operaciones que se realizan sobre ella.
  Si un objeto requiere cierta información para realizar una operación que es su responsabilidad, debe de asignarsele a él la responsabilidad de mantener dicha información.

- Mantener la información especifica a algo en un solo lugar. La información nunca debe ser duplicada para evitar inconsistencias. Si más de un objeto necesita conocer la misma información para realizar una acción, hay tres posibles soluciones:

  . Crear un nuevo objeto depositario de dicha información.
  . Reasignar todas las acciones que requieren dicha información al objeto que más operaciones realiza con la misma.
  . Unificar todos los objetos que realizan operaciones con dicha información en un solo objeto.

- Compartir responsabilidades. Descomponer responsabilidades en otras más pequeñas, responsabilidades especificas y asignar cada una de estas a las clases más adecuadas.

Dificultades comunes

- Clases perdidas. Añadir una nueva clase par encapsular un conjunto de responsabilidades no asignadas y relacionadas, puede ser una buena decisión en el diseño.

- Asignaciones arbitrarias. Cuando una responsabilidad puede ser asignada a más de una clase hay que ver cada posibilidad que significa desde el punto de vista de la funcionalidad del sistema y de su apariencia y comportamiento. Puede necesitarse para decidir recorrer partes del sistema en los diferentes casos analizando su comportamiento.

Registro de responsabilidades

En la tarjeta creada para cada clase se registran las responsabilidades asignadas a ella. Las responsabilidades deben registrarse lo más resumidamente posible, una frase para cada responsabilidad.

3. **Determinar el modo en que cada objeto colabora con el resto para cumplir sus responsabilidades.**

El diseño orientado a objetos modela al mundo en términos de colaboración entre objetos para cumplir sus responsabilidades. Así pues, un objeto realizando una solicitud es llamado cliente y el

objeto que recibe la solicitud y ejecuta el servicio es llamado servidor. el cliente y el servidor interactuan a través de un contrato.
Los objetos en un sistema pueden ser clientes o servidores según el papel que desempeñen en un momento dado.

Una colaboración es una solicitud de un cliente a un servidor para cumplir una responsabilidad del cliente, es decir, una colaboración es la forma de determinar que clientes y que servidores están unidos por contratos. Desde el punto de vista del cliente, cada una de sus colaboraciones está asociada con una responsabilidad implementada por un servidor.

Un contrato es una lista de solicitudes que el cliente puede hacer al servidor para realizar sus responsabilidades, o bien, es un grupo de responsabilidades de una clase relacionadas de alguna manera.

Cada clase puede tener cualquier número de contratos, cada responsabilidad de ella puede pertenecer a lo sumo a un contrato, las responsabilidades de las clases pueden crear contratos entre ellas, pero las responsabilidades privadas no pertenecen a ningún contrato de la clase.

La definición de contratos comienza en el tope de una jerarquía y se añaden nuevos contratos a las subclases que agregan nuevas funciones a usar por diferentes clientes. Si en una subclase no se añaden nuevas funciones o si no se especifican funciones de la superclase no se crean nuevos contratos y se mantiene el contrato heredado.

Guías para determinar la pertenencia de responsabilidades a contratos

- Agrupar responsabilidades requeridas por los mismos clientes.
- Maximizar la coherencia de las clases.
- Minimizar el número de contratos.

La identificación de colaboraciones permite:

- Encontrar caminos de comunicación entre clases y en última instancia identificar subsistemas de clases que colaboran.
- Determinar que clase hace el papel de cliente y que clase el papel de servidor en cada contrato, y utilizando esta información encontrar responsabilidades omitidas y responsabilidades incorrectamente asignadas.

Búsquedas de colaboraciones

- realizar las siguientes preguntas para cada responsabilidad de cada clase:

  1. ¿Es la clase capaz de cumplir la responsabilidad ella sola?
  2. ¿Que necesita entonces para poder cumplir la responsabilidad?
  3. ¿De que otras clases puede obtener lo que ella necesita?

  Similarmente para cada clase debe preguntarse:

  1. ¿Que hace o que conoce la clase?
  2. ¿Que otras clases necesitan sus resultados o su información?
  3. Si una clase no tiene interacciones con otras clases, entonces dicha clase debe ser

eliminada.

Nota: Una clase no tiene interacciones con otras clases si ella no colabora con otras clases ni otras clases colaboran con ella.

. Utilización de relaciones entre objetos:

- Relación *"es parte de"*
- Relación *"es análoga a"*
- Relación *"es tipo de"*

. Recorrido de escenarios a partir de diferentes entradas tantas veces como sea necesario para explorar completamente el funcionamiento de cada objeto.

. Hacer un registro de colaboraciones de clases.

IV. DISEÑO DEL SISTEMA DE CAPTURA DE SISTEMAS HIDRÁULICOS

El proceso de diseño antes mencionado se aplicará a continuación en el diseño del sistema de captura de datos para acueductos.

IV.1 Búsqueda de clases

Análisis de la solución:

La interfase gráfica para la captura de datos de sistemas hidráulicos será un programa mediante el cual el usuario pueda realizar la captura de información para los distintos dispositivos hidráulicos (tanques, bombas, líneas de conducción, nudos, etc.) que conforman una red hidráulica y almacenarlos en un archivo de datos. Este programa consistirá de una pantalla de trabajo, un menú de comandos, un menú de colores, un menú de íconos y una zona donde se despliegan las coordenadas de la pantalla de trabajo. El programa será desarrollado con tecnología orientada a objetos en Borland Pascal 7.

Para la interacción con el usuario se usará el ratón y cajas de diálogos con botones de lectura, botones de título, botones con imágenes, botones de mensajes, etc.

La mecánica para seleccionar cualquier botón es:

1. El usuario deberá colocar el ratón dentro del botón a seleccionar.
2. Presionar una vez el botón izquierdo del ratón.

Para cancelar la ejecución de cualquier proceso deberá presionar el botón derecho del ratón o la tecla de escape "ESC".

▸Pantalla de trabajo

La pantalla de trabajo es la zona en donde se podrá dibujar la red hidráulica, para poder realizar esto, se cuenta con un menú de íconos que representan los dispositivos hidráulicos; el usuario podrá

seleccionar un ícono a través del ratón y arrastrarlo hasta la pantalla de trabajo para ir formando dicha red, la mecánica para realizarlo se describe a continuación:

1. Seleccionar el botón con el ícono del dispositivo deseado.
2. Sin dejar de presionar el botón izquierdo del ratón arrastrar el dispositivo hasta la posición deseada en la pantalla de trabajo.
3. Soltar el botón izquierdo del ratón para que se dibuje el dispositivo en la posición seleccionada.

▸Coordenadas de la pantalla de trabajo

Las coordenadas de la pantalla de se despliegan en la esquina inferior derecha, así pues, cada vez que el ratón entra a la zona de trabajo las coordenadas se van actualizando tomando nuevos valores dependiendo de la posición del ratón.

▸ Cajas de diálogo y tablas de datos

Cada dispositivo tendrá una o varias cajas de diálogo y una o varias tablas de datos, una vez que se ha dibujado un nuevo dispositivo en la pantalla de trabajo el usuario proporcionará la información correspondiente. Los datos de un dispositivo se podrán consultar y actualizar. Para realizar esto se seleccionará, con el ratón, el dispositivo del cual se desea consultar su información, presionando dos veces el botón izquierdo del ratón. Se abrirán sus cajas de diálogo y sus tablas de datos en la pantalla; de esta manera el usuario podrá moverse a través de ellas para actualizar la información o simplemente consultarla.

Para modificar la información de una tabla, el usuario deberá mover el ratón al renglón y columna donde se quiere registrar los datos y presionar el botón izquierdo del ratón.
Todas las tablas tendrán un encabezado de columna, un número de renglón (opcional), un menú de comandos con las siguientes funciones: adicionar, insertar, borrar, mover y continuar. Así como también tendrán cuatro funciones de desplazamiento representadas por dos barras verticales y dos flechas respectivamente, que son: avance de página, regreso de página, avanzar un renglón y regresar un renglón; estas funciones permitirán al usuario desplazarse a lo largo de la tabla. El usuario podrá moverse con el ratón solamente en la zona visible de la tabla de datos.

Comandos de la tabla de datos:

Adicionar: Adiciona un nuevo renglón al final de la tabla. Para ello el usuario deberá seleccionar el botón "adicionar".

Borrar: Borra un renglón de la tabla, actualizandola automáticamente. El usuario deberá seleccionar el botón "borrar", seleccionar el renglón que se desea borrar presionando el botón izquierdo del ratón dos veces.

Insertar: Inserta un nuevo renglón en la tabla, actualizandola automáticamente. El usuario deberá seleccionar el botón "insertar", seleccionar el renglón en donde se quiere insertar presionando dos veces el botón izquierdo del ratón (el usuario podrá desplazarse a lo largo de la tabla con los botones de desplazamiento de la tabla o bien moviendo el ratón en la zona visible de la tabla, para buscar el renglón en donde se quiere insertar el nuevo renglón)

Mover: Mueve un renglón a otra posición de la tabla, actualizandola automáticamente. El usuario deberá seleccionar el botón "mover", seleccionar el renglón que se desea mover presionando dos veces el botón izquierdo del ratón, seleccionar el renglón donde se

desea insertar presionando dos veces el botón izquierdo del ratón.

Terminar: salir de la tabla.

Para interrumpir el proceso de alguno de estos comandos el usuario deberá presionar el botón derecho del ratón.

A diferencia de una tabla, una forma tiene botones donde se captura la información que de alguna manera no debe agruparse como una tabla, tal es el caso de los tanques. Un tanque se define con base en su elevación y tirante de agua respecto al nivel del piso, siendo estos datos únicos, (en el sentido de que un tanque no tiene varias elevaciones o inicialmente no se definen varios tirantes) La manera más adecuada de capturar esta información es lo que hemos llamado una caja de diálogo.

▸Menú de colores

El menú de colores permitirá al usuario cambiar el color de dibujo en el momento que desee, para hacer esto, el usuario deberá seleccionar el color que desea utilizar. A la derecha del menú de colores se encuentra un botón de color, ese botón indicará al usuario cual es el color actual de dibujo, este botón cambiará su color cada vez que se seleccione un color de este menú.

▸Menú de comandos

El menú de comandos permitirá al usuario realizar operaciones en la pantalla de trabajo y en la red hidráulica. Estos comandos son: archivo, borrar, copiar, mover, escalar, texto, ayuda, corrimiento.

Archivo: Proporcionará un menú con las siguientes opciones:

a) Crear. Creará un nuevo archivo de datos.
b) Recuperar. Recuperará un archivo de datos.
c) Salvar. Grabará el archivo de datos en uso.
d) Salvar como. Grabará el archivo de datos en uso con otro nombre.

Escalar: Hará un acercamiento y/o alejamiento de la pantalla modificando la escala de todo lo que se encuentre en la pantalla de trabajo. Cuando el usuario elige esta opción se abre un menú de escalamiento con los siguientes factores de escalación:

a) x2 y x5:
Estas opciones hace un acercamiento de 2:1 y de 5:1 respectivamente.

b) ÷2 y ÷5:
Estas dos opciones hacen un alejamiento de 1:2 y 1:5 respectivamente.

Para realizar un escalamiento, el usuario deberá seleccionar el botón "escalar" y seleccionar alguna de las opciones de escalamiento.

Malla: El botón "malla" dibujará una malla de puntos equiespaciados con un incremento de (1,1), esta malla es una referencia para la localización del dibujo. El usuario podrá activar la malla simplemente seleccionando el botón con el ratón y para desactivarla deberá presionar (seleccionar) el botón nuevamente.

Recorrer: Permitirá al usuario recorrer el dibujo a otro punto de la pantalla de trabajo. El usuario deberá seleccionar el botón "recorrer" y teclear la nueva posición (x,y) donde se quiere recorrer la imagen.

Saltos: El botón "saltos" forzará a que el cursor haga movimientos discretos dentro de la pantalla de trabajo, es decir cada vez que el usuario mueva el ratón dentro de la pantalla de trabajo, este se moverá en saltos unitarios relativos a la escala del dibujo. El usuario podrá activar los saltos simplemente seleccionándolos con el ratón y para desactivarlos deberá presionar (seleccionar) el botón nuevamente.

Texto: Permitirá poner texto en la pantalla de trabajo. Para realizarlo el usuario deberá:

a) Seleccionar el botón "texto".
b) Se abrirá una ventana donde el usuario tecleará el texto.
c) Al terminar de escribir el texto deberá de presionar la tecla "enter".
d) Seleccionará con el ratón la posición en donde se desea colocar el texto.

Lista de frases nominales que aparecen en la especificaciones de requerimientos.

acercamiento
alejamiento
archivo
archivo de datos
barra vertical
bomba
botón de color
botón de malla
botón de mensaje
botón del ícono
botón de saltos
botón de título
botón adicionar
botón borrar
botón con imagen
botón recorrer
botón de desplazamiento
botón derecho del ratón
botón escalar
botón insertar
botón izquierdo del ratón
botón mover
botón texto
botón de lectura
botón
caja de diálogo
captura de información
captura de datos
color
color de dibujo
color actual de dibujo
columna
coordenadas de la pantalla
coordenadas
cursor
datos del dispositivo
datos
dibujo
dispositivo
dispositivos hidráulicos
elemento
elevación
encabezado de columna
escala del dibujo
escala
escalamiento
esquina inferior derecha
factores de escalación
flecha
forma
función
función de desplazamiento
ícono
imagen
información
interfase gráfica
línea de conducción
malla
malla de puntos
mecánica
menú de escalamiento
menú de comandos
menú
menú de colores

menú de íconos
nombre
nudo
nuevo dispositivo
nuevo renglón
número de renglón
opción
opción de escalamiento
pantalla de trabajo
pantalla
posición del ratón
posición seleccionada
posición
proceso
programa
punto de la pantalla de trabajo
ratón
red hidráulica
red
renglón
renglón de la tabla
salto unitario relativo
sistema de captura
sistema hidráulico
tabla de dispositivos
tabla
tabla de datos
tanque
tecla escape "ESC"
tecla enter
texto
tirante del agua
usuario
nuevo valor
zona visible de la tabla
zona

Selección de una frase para expresar mejor un concepto

archivo de datos	(por archivo)
barra de desplazamiento	(por barra vertical y función de desplazamiento)
botón con imagen	(por botón del ícono)
botón de color actual	(por botón de color)
botón lateral	(por botón de desplazamiento)
dispositivo hidráulico	(por dispositivo, nuevo dispositivo)
coordenada	(por coordenadas de la pantalla)
conjunto de colores	(por menú de colores)
conjunto de comandos	(por menú de comandos)
conjunto de íconos	(por menú de íconos)
menú de escalamiento	(por escalamiento, factores de escalación)
menú de archivo de datos	(por menú)
pantalla de trabajo	(por pantalla)
renglón de la tabla	(por renglón y nuevo renglón)

sistema hidráulico	(por red, red hidráulica)
tabla de datos	(por tabla y tabla de dispositivos)

Quitar todas las frases que no constituyen clases

Atributos:

color
color actual de dibujo
color de dibujo
elevación
escala
escala del dibujo
nombre
número de renglón
posición
posición seleccionada
posición del ratón
punto de la pantalla de trabajo
tirante del agua
nuevo valor

Frases nominales que pueden ser eliminadas analizando su significado en el contexto en que aparecen:

acercamiento
alejamiento
captura de información
captura de datos
columna
cursor
datos del dispositivo
datos
elemento
encabezado de columna
flechas
forma
función
ícono
imagen
interfase gráfica
información
malla de puntos
mecánica
opción
opción de escalamiento
proceso
programa
salto unitario relativo
sistema de captura
zona
zona visible

Modelo de objetos físicos

archivo de datos
botón izquierdo del ratón
botón derecho del ratón
ratón
tecla enter
tecla esc

Eliminación de frases nominales que denotan objetos externos al sistema

. esquina inferior derecha
. usuario

. Las frases nominales "tecla enter" y "tecla esc" serán contenidas en la frase nominal "teclas".

. La frase "archivo de datos" es un resultado de la red hidráulica almacenada físicamente, pero ese medio físico debe ser modelado como un objeto que conozca como almacenarse, como recupera esa información en tablas, como interpretarla para crear gráficamente la red en pantalla, etc. por lo que, ésta frase nominal será modelada como una entidad conceptual.

Modelo de entidades conceptuales

archivo de datos
barra de desplazamiento
bomba
botón
botón escalar
botón lateral
botón recorrer
botón texto
botón con imagen
botón de malla
botón de saltos
botón mover
botón de mensaje
botón de título
botón de lectura
botón adicionar
botón borrar
botón de color actual
botón insertar
caja de diálogo
coordenada
conjunto de colores
conjunto de comandos
conjunto de funciones
conjunto de íconos
dibujo
dispositivo hidráulico

línea de conducción
menú de archivo de datos
menú de escalamiento
nudo
pantalla de trabajo
ratón
renglón de la tabla
sistema hidráulico
tabla de datos
tanque
teclas
texto

La frase nominal dibujo quedará implícita en la frase nominal sistema hidráulico.

La frase nominal texto quedará implícita en la frase nominal botón texto

Frases nominales candidatas a clases

archivo de datos
barra de desplazamiento
bomba
botón
botón de lectura
botón de título
botón de mensaje
botón mover
botón adicionar
botón borrar
botón insertar
botón de color actual
botón de saltos
botón lateral
botón recorrer
botón escalar
botón texto
botón malla
botón con imagen
caja de diálogo
coordenada
conjunto de colores
conjunto de comandos
conjunto de funciones
conjunto de íconos
dispositivo hidráulico
línea de conducción
menú de archivo de datos
menú de escalamiento
nudo
pantalla de trabajo
ratón
renglón de la tabla

sistema hidráulico
tabla de datos
tanque
teclas

Búsqueda de clases abstractas

Se agrupan todas las clases que tengan comportamiento común, para posteriormente generar una jerarquía de clases:

. Dispositivo hidráulico
    Bomba
    Línea de conducción
    Nudo
    Tanque
. Botón
    Botón de coordenada
    Botón con imagen
    Botón de título
    Botón de color
        Botón de color actual
    Botón de mensaje
        Botón adicionar
        Botón borrar
        Botón con imagen
        Botón recorrer
        Botón escalar
        Botón insertar
        Botón malla
        Botón mover
        Botón saltos
        Botón texto

    Botón de lectura
        LeeEnteros
        LeeReales
        LeeCaracteres
    Pantalla de trabajo

. Colecciones de objetos
    Caja de diálogo
    Renglón de la tabla
    Sistema hidráulico
    Tabla de datos
    Botones
        Conjunto de colores
        Conjunto de íconos
        Conjunto de comandos
    Menú
        Menú de archivo de datos
        Menú de escalamiento

Las frases nominales Botón adicionar, Botón borrar, Botón con imagen, Botón recorrer, Botón escalar, Botón insertar, Botón mover, Botón texto, Botón de malla y Botón de saltos, son un mismo tipo de objeto (instancias del mismo objeto) pero mandan distintos mensajes; por lo que es necesario crear una superclase que contenga sus características comunes, esta clase se llamará Botón de función.

La frase nominal Botones es un conjunto de objetos de tipo botón; por lo que Conjunto de colores, Conjunto de íconos y Conjunto de comandos quedarán representados por el objeto de tipo Botones.

La Barra de desplazamiento tiene como objetivo desplazar información en la tabla de datos por lo que es necesario crear dos subclases de la clase Botón lateral con el nombre de Botón de avance y Botón de retroceso.

Frases nominales candidatas a clases del sistema

Archivo de datos

Barra de desplazamiento

Ratón

Teclas

Dispositivo hidráulico
- Bomba
- Línea de conducción
- Nudo
- Tanque

Botón
- Botón de coordenada
- Botón con imagen
- Botón de color
  - Botón de color actual
- Botón lateral
  - Botón de avance
  - Botón de retroceso
- Botón de mensaje
  - Botón de función
  - Botón de título
- Botón de lectura
  - LeeEnteros
  - LeeReales
  - LeeCaracteres
- Pantalla de trabajo

Colección de objetos
- Caja de diálogo
- Renglón de la tabla

Sistema hidráulico
Tabla de datos
Botones
Menú
    Menú de archivo de datos
    Menú de escalamiento

Los diagrámas de clases se presentan en las figuras IV.1 y IV.2

Objetivos de las clases

1. Archivo de datos.
   Almacenar la red hidráulica en dispositivos magnéticos y así mismo, cargar dicha red al sistema de captura.

2. Barra de desplazamiento.
   Contener los botones de desplazamiento como un solo objeto.

3. Botón de coordenada.
   Desplegar las coordenadas del ratón cuando se encuentra dentro de la pantalla de trabajo.

4. Ratón.
   Enviar y recibir eventos, es decir generar mensajes al sistema y recibir mensajes del sistema.

5. Dispositivo hidráulico.
   Representar a los dispositivos con los cuales se podrá formar el sistema hidráulico.

6. La Bomba, la Línea de conducción, el Nudo y el Tanque.
   Representar al dispositivo propio.

7. Botón.
   Permitir la interacción con el usuario, haciendo el programa más amigable.

8. Botón de color.
   Permitir la selección de color de dibujo en cualquier momento (solo se cambiará el color de dibujo en el momento en que se solicite, pero no cambiará el color de la red ya dibujada).

9. Botón de color actual
   Actualizar el color de dibujo.

10. Botón con imagen.
    Presentar un botón con un dispositivo en forma de ícono; la imagen se leerá de un archivo.

11. Botón lateral.
    Desplegar un botón sin texto y de un tamaño fijo. Hacer el recorrido de la tabla hacia adelante o hacia a tras página por página.

12. Botón de avance.
    Desplegar un botón de tamaño fijo con un símbolo dibujado, este símbolo se creará dibujando pixel por pixel y hacer el recorrido de la tabla hacia adelante renglón por renglón.

Sistema hidráulico
Tabla de datos
Botones
Menú
Menú de archivo de datos
Menú de escalamiento

Los diagrámas de clases se presentan en las figuras IV.1 y IV.2

Objetivos de las clases

1. Archivo de datos.
   Almacenar la red hidráulica en dispositivos magnéticos y así mismo, cargar dicha red al sistema de captura.

2. Barra de desplazamiento.
   Contener los botones de desplazamiento como un solo objeto.

3. Botón de coordenada.
   Desplegar las coordenadas del ratón cuando se encuentra dentro de la pantalla de trabajo.

4. Ratón.
   Enviar y recibir eventos, es decir generar mensajes al sistema y recibir mensajes del sistema.

5. Dispositivo hidráulico.
   Representar a los dispositivos con los cuales se podrá formar el sistema hidráulico.

6. La Bomba, la Línea de conducción, el Nudo y el Tanque.
   Representar al dispositivo propio.

7. Botón.
   Permitir la interacción con el usuario, haciendo el programa más amigable.

8. Botón de color.
   Permitir la selección de color de dibujo en cualquier momento (solo se cambiará el color de dibujo en el momento en que se solicite, pero no cambiará el color de la red ya dibujada).

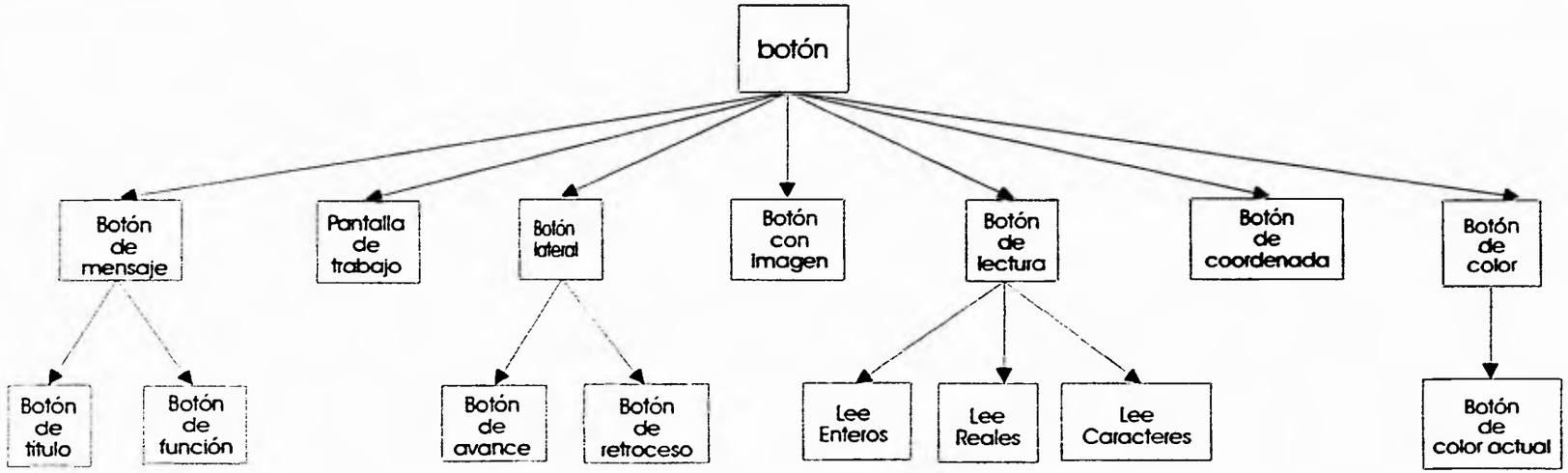
9. Botón de color actual
   Actualizar el color de dibujo.

10. Botón con imagen.
    Presentar un botón con un dispositivo en forma de ícono; la imagen se leerá de un archivo.

11. Botón lateral.
    Desplegar un botón sin texto y de un tamaño fijo. Hacer el recorrido de la tabla hacia adelante o hacia a tras página por página.

12. Botón de avance.
    Desplegar un botón de tamaño fijo con un símbolo dibujado, este símbolo se creará dibujando pixel por pixel y hacer el recorrido de la tabla hacia adelante renglón por renglón.

Jerarquía del botón:



Jerarquía de dispositivo:

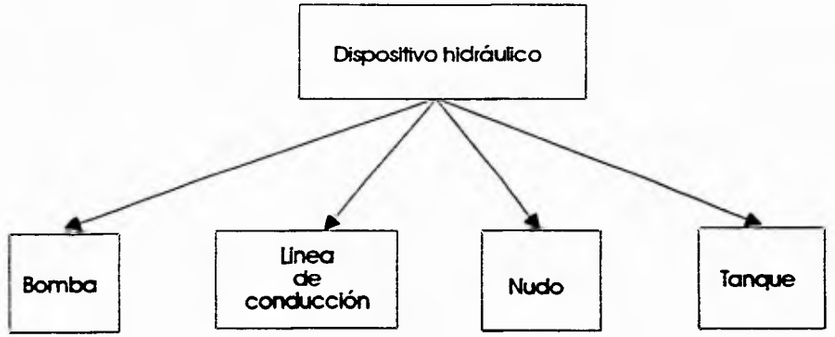
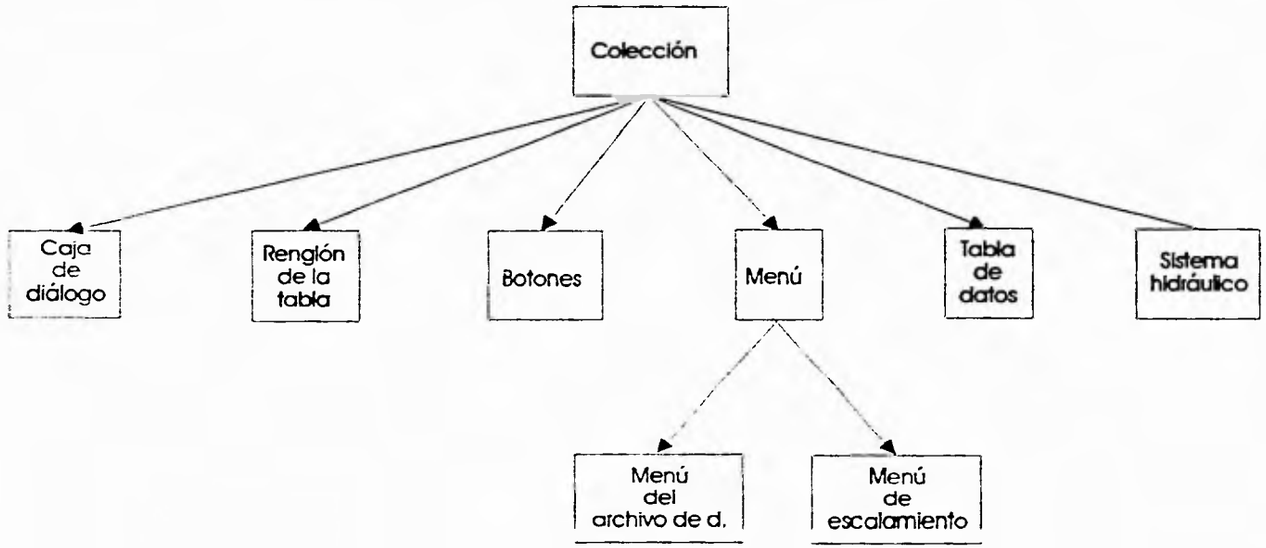


Figura IV.1 jerarquía de la clase botón y de la clase dispositivo

Jerarquía de colección:



Jerarquía de Barra de desplazamiento:

Jerarquía de Teclas:

Jerarquía de Ratón:

Jerarquía de Archivo de datos:

Figura IV.2 Jerarquías de las clases colección, barra de desplazamiento, teclas, ratón y archivo de datos.

13. Botón de retroceso.
    Desplegar un botón de tamaño fijo con un símbolo dibujado, este símbolo se creará dibujando pixel por pixel y Hacer el recorrido de la tabla hacia a tras renglón por renglón.

14. Botón de mensaje.
    Desplegar un botón con texto permitiendo simular la acción de un botón presionado y la de un botón no presionado.

15. Botón de función.
    Desplegar un botón con texto permitiendo simular la acción de un botón presionado y la de un botón no presionado; al ser presionado este botón enviará un mensaje al sistema.

16. Botón de título.
    Desplegar un botón con texto sin tener función alguna.

17. Botón de lectura.
    Capturar información del teclado.

18. LeeEnteros.
    Capturar y validar información entera numérica del teclado.

19. LeeReales.
    Capturar y validar números reales del teclado.

20. LeeCaracteres.
    Capturar y validar datos tipo carácter del teclado.

21. Pantalla de trabajo.
    Representar el plano de trabajo en la pantalla de video, permitiendo almacenar y modificar en la pantalla el sistema hidráulico.

22. Colección.
    Permitir formar un conjunto de objetos. (Este objeto es definido por Borland Pascal 7).

23. Caja de diálogo.
    Permitirá una interacción amigable con el usuario, conteniendo un conjunto de objetos para su interacción.

24. Renglón de la tabla.
    Capturar diversos tipos de datos por cada uno de sus campos, es decir en un campo capturar un número real, en otro un entero, etc.

25. Sistema hidráulico.
    Almacenar la red hidráulica en pantalla y en un archivo de datos.

26. Botones.
    Formar un conjunto de botones de la misma clase.

27. Menú.
    Formar un menú con una base, título, y un conjunto de botones (opciones).

28. Menú de archivo de datos.

    Proporcionar y ejecutar opciones para los archivos de datos.
29. Menú de escalamiento.
    Proporcionar opciones de escalamiento en la pantalla de trabajo.

30. Tabla de datos.
    Permitir consultar y/o actualizar información de un dispositivo en particular.
    Hacer un "scroll" (viajar a lo largo de la tabla) siempre y cuando halla más renglones en la tabla que los que se muestran por omisión, así también capturar, consultar o modificar información de cada uno de los dispositivos hidráulicos.

31. Teclas.
    Representar el teclado físico en el sistema

IV.2 Asignación de responsabilidades para el sistema de captura

Identificación de responsabilidades

Para encontrar las acciones realizadas por el sistema se tomarán las frases verbales del sistema propuesto y se construirá una lista con estas frases:

- Construye (dibuja) la red hidráulica en la pantalla de trabajo.
- Almacena la red hidráulica en un archivo de datos.
- Usa el ratón.
- Dibuja el ratón (cursor) en el video.
- Cambia la representación del cursor.
- Selecciona opciones de cualquier botón seleccionable.
- Representa dispositivos hidráulicos a través de íconos.
- Toma un dispositivo del menú de íconos y lo copia a la pantalla de trabajo.
- Dibuja dispositivos hidráulicos en la pantalla de trabajo.
- Despliega las coordenadas del ratón, es decir, despliega la posición del ratón cuando éste se encuentra en la pantalla de trabajo.
- Actualiza la posición del ratón.
- Abre (presenta) y cierra cajas de diálogos.
- Abre (presenta) y cierra tablas de datos.
- Captura datos del teclado.
- Permite la consulta y actualización de la información.
- Permite al usuario viajar a lo largo de la información almacenada en tablas.
- Realiza operaciones de adición, borrado, inserción y movimiento de renglones en la tabla de datos.
- Permite la interrupción de cualquier comando de la tabla de datos.
- Indica el color actual de dibujo.
- Cambia el color actual de dibujo.
- Modifica características de la pantalla de trabajo.
- Dibuja una malla de puntos.
- Cambia el movimiento del cursor a movimientos unitarios discretos.
- Activa y desactiva funciones.
- Realiza operaciones sobre la red hidráulica.
- Borra dispositivos de la red hidráulica.
- Actualiza la red hidráulica.

- Realiza el duplicado de dispositivos en la pantalla de trabajo.
- Realiza el duplicado de datos de un dispositivo a otro.
- Mueve la red hidráulica a otro punto de la pantalla de trabajo.
- Presenta un menú de escalamiento.
- Hace escalamientos de la pantalla de trabajo.
- Modifica la escala de los dispositivos hidráulicos que se encuentran en la pantalla de trabajo.
- Mueve los dispositivos hidráulicos a otra posición de la pantalla de trabajo.
- Permite desplegar texto sobre la pantalla de trabajo.
- Captura texto del teclado.
- Crea archivos de datos.
- Recupera archivos de datos del disco.
- Graba el archivo de datos en uso.
- Graba el archivo de datos en uso con otro nombre.
- Ofrece ayuda al usuario sobre el funcionamiento del sistema de captura.

Posteriormente se hace un análisis general del comportamiento interno del sistema propuesto, pasándolo por varios escenarios y probando sus capacidades (ver figura IV.3).

Para la asignación de responsabilidades se tomarán encuenta los siguientes puntos:

. El objetivo de cada una de las clases.
. El comportamiento interno del sistema propuesto
. Las frases verbales tomadas de las especificaciones del sistema propuesto.

Haciendo un análisis completo de los tres puntos anteriores, se realiza la asignación de responsabilidades a cada una de las clases. En esta etapa de diseño algunas candidatas a clases serán desechadas y aparecerán clases que no se encontraban dentro de las candidatas modificando la jerarquía de clases existente.

Asignación de responsabilidades

1. Archivo de datos.

   - Presentar el menú de archivo de datos.
   - Solicitar el almacenamiento de la red hidráulica.
   - Solicitar la recuperación de un archivo de datos.
   - Cambiar el nombre del archivo de datos y solicita el almacenamiento de la red hidráulica con otro nombre.
   - Solicitar la terminación del sistema.

2. Barra de desplazamiento.

   - Contener los botones de avance y retroceso de la tabla de datos.
   - Notificar a todos sus elementos el cambio de sus posiciones.
   - Notificar a todos sus elementos el mensaje de dibujarse.
   - Enviar replicas del mensaje a cada uno de sus elementos.

3. Botón.

   - Cambiar su estado de botón presionado a botón no presionado y viceversa.
   - Cambiar de posición.
   - Dibujarse.

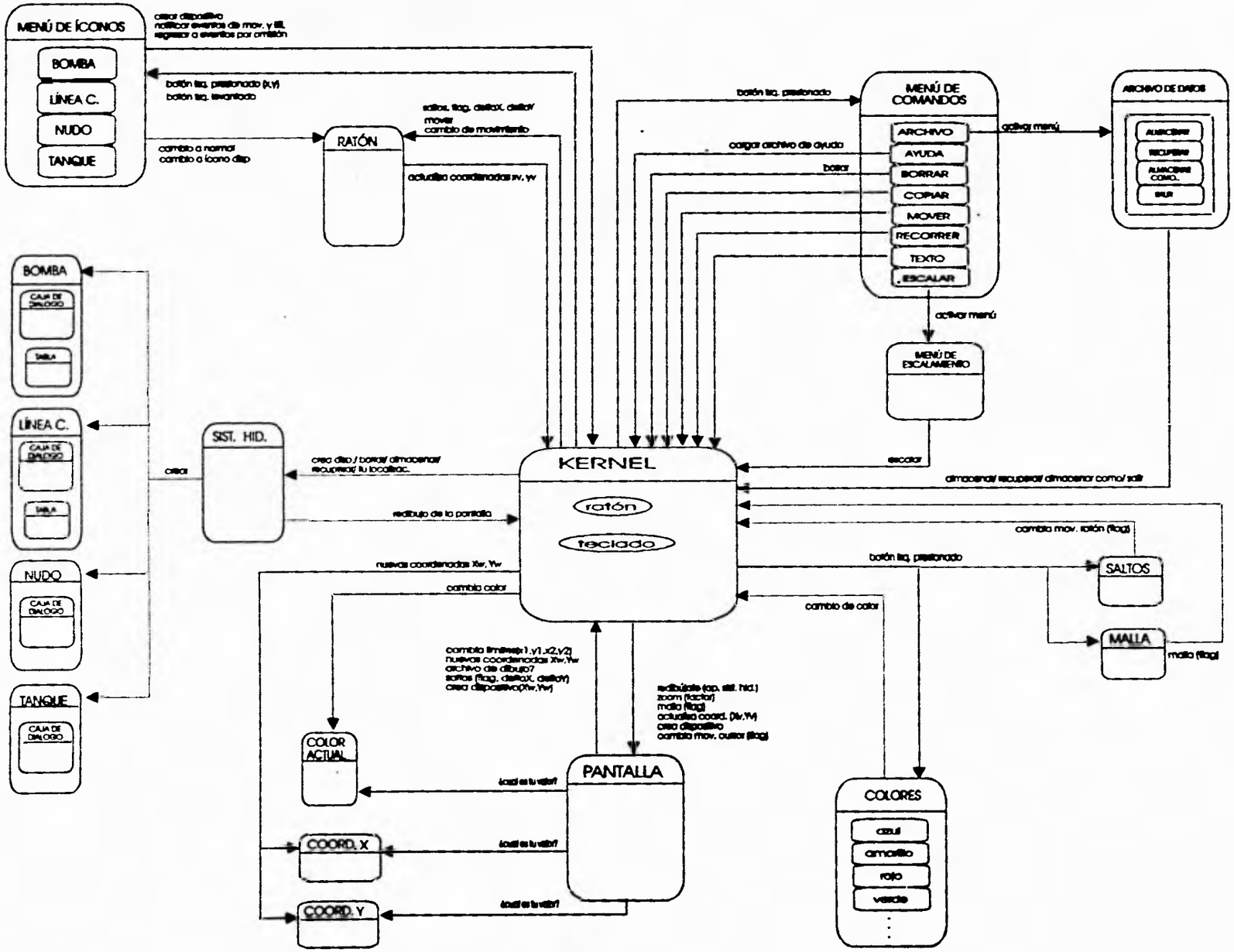


Figura IV.3 Esquema general de los objetos que intervienen en el sistema.

4. Botón con imagen.

   - Representar un dispositivo hidráulico a través de íconos.
   - Cambiar su estado de botón presionado a botón no presionado y viceversa.
   - Dibujarse.
   - Solicitar el cambio de representación gráfica del ratón de acuerdo al ícono seleccionado.
   - Solicita al kernel notificación por los eventos de botón levantado y de movimiento del ratón.
   - Crear en memoria un nuevo dispositivo hidráulico.
   - Solicitar la creación de un nuevo dispositivo hidráulico proporcionando el apuntador del nuevo dispositivo.
   - Solicitar el cambio de representación gráfica del ratón a modo normal.

5. Botón Bomba

   - Representar el dispositivo hidráulico "bomba" a través del ícono bomba.
   - Crear en memoria el dispositivo hidráulico bomba.
   - Solicitar la creación del dispositivo hidráulico "bomba" proporcionando su apuntador.

6. Botón Línea de conducción

   - Representar el dispositivo hidráulico "línea de conducción" a través del ícono línea de conducción.
   - Crear en memoria el dispositivo hidráulico "línea de conducción".
   - Solicitar la creación del dispositivo hidráulico "línea de conducción" proporcionando su apuntador.

7. Botón Nudo

   - Representar el dispositivo hidráulico "nudo" a través del ícono nudo
   - Crear en memoria el dispositivo hidráulico "nudo".
   - Solicitar la creación del dispositivo hidráulico "nudo" proporcionando su apuntador.

8. Botón Tanque

   - Representar el dispositivo hidráulico "tanque" a través del ícono tanque.
   - Crear en memoria el dispositivo hidráulico tanque.
   - Solicitar la creación del dispositivo hidráulico "tanque" proporcionando su apuntador.

9. Botón lateral.

   - Desplegar un botón sin texto y de un tamaño fijo.
   - Cambiar su estado de botón presionado a botón no presionado y viceversa.
   - Solicitar el avance o retroceso de página.

10. Botón de avance.

    - Desplegar un botón de tamaño fijo con un símbolo dibujado.
    - Cambiar su estado de botón presionado a botón no presionado y viceversa.
    - Solicitar el avance de un renglón.

11. Botón de retroceso.

- Desplegar un botón de tamaño fijo con un símbolo dibujado.
- Cambiar su estado de botón presionado a botón no presionado y viceversa.
- Solicitar el retroceso de un renglón.

12. Botón de título.

    - Desplegar un botón con texto sin tener función alguna.

13. Botón de mensaje.

    - Desplegar un botón con texto.
    - Cambiar su estado de botón presionado a botón no presionado y viceversa.

14. Botón de coordenada.

    - Desplegar un botón con texto constante y con texto variable (desplegando distintos valores de acuerdo al mensaje).
    - Cambiar el valor del texto variable y desplegarlo.
    - Desplegarse.
    - Proporcionar la coordenada.

15. Botones.

    - Formar un conjunto de botones de la misma clase.

16. Botón de color.

    - Solicitar el cambio del color de dibujo.
    - Cambiar su estado de botón presionado a botón no presionado y viceversa.

17. Botón de color actual.

    - Proporcionar al cliente el color actual de dibujo.
    - Cambiar el color actual de dibujo.

18. Botón de función.

    - Desplegar un botón con texto.
    - Cambiar su estado de botón presionado a botón no presionado y viceversa.
    - Solicitar la realización de dicha función de acuerdo al mensaje enviado.

19. Botón de función con bandera.

    - Desplegar un botón con texto.
    - Cambiar el estado actual de su bandera.
    - Cambiar su estado de botón presionado a botón no presionado y viceversa.
    - Solicitar la realización de dicha función de acuerdo a la estado de la bandera enviada.

20. Botón de lectura de caracteres.

    - Capturar información del teclado (el número de caracteres es determinado).
    - Desplegar los caracteres conforme se van capturando.

- Desplegar información
- Desplegarse.
- Cambiar su estado de botón presionado a botón no presionado y viceversa.
- Solicitar el cambio de representación gráfica del ratón a modo texto.
- Proporcionar al cliente el tipo de dato capturado.
- Leer un tipo de dato proporcionado por el cliente.
- Validar los datos capturados para un tipo especifico de dato.
- Realizar la conversión de una cadena de caracteres a un tipo de datos en particular.
- Realizar la conversión de un tipo de datos en particular a una cadena de caracteres.

21. LeeEnteros.

- Proporcionar al cliente el dato capturado en el formato de número entero.
- Leer un entero proporcionado por el cliente.
- Validar los datos capturados para números enteros.
- Convertir números enteros a cadenas de caracteres.
- Convertir los datos capturados a datos enteros.
- Cambiar su estado de botón presionado a botón no presionado y viceversa.

22. LeeReales.

- Proporcionar al cliente el dato capturado en el formato de número real.
- Leer un número real proporcionado por el cliente.
- Validar los datos capturados para números reales.
- Convertir números reales a cadenas de caracteres.
- Convertir los datos capturados a números reales.
- Cambiar su estado de botón presionado a botón no presionado y viceversa.

23. Caja de diálogo.

- Permitir una interacción amigable con el usuario.
- Contener un conjunto de objetos para su interacción.
- Consultar y actualizar información de los dispositivos hidráulicos que no se puede representar en tablas.
- Dibujarse.

24. Caja de diálogo de la bomba.

- Contener información propia de la bomba.

25. Caja de diálogo de la línea de conducción.

- Contener información propia de la línea de conducción.

26. Caja de diálogo del nudo.

- Contener información propia del nudo.

26. Caja de diálogo del tanque.

- Contener información propia del tanque.

24. Colección.

- Permitir formar un conjunto de objetos de la misma jerarquía de clase. (Este objeto es definido por Borland Pascal 7).

25. Dispositivo hidráulico.

- Crear su forma (caja de diálogo).
- Crear su tabla de datos.
- Almacenar su información.
- Recuperar su información de un archivo de datos.
- Proporcionar al cliente el nombre de su archivo de dibujo
- Cambiar su posición.

26. La Bomba, la Línea de conducción, el Nudo y el Tanque.

- Representar gráficamente al dispositivo propio.

27. Kernel (generador de eventos).

- Interpretar los eventos del ratón y del teclado.
- Recibir y Enviar eventos a los objetos, tales como:
  . Nueva coordenada x.
  . Nueva coordenada y.
  . Botón izquierdo presionado.
  . Botón izquierdo levantado.
  . Crear un nuevo dispositivo.
  . Movimiento del ratón.
  . Cambio de color.
  . Cambio de movimiento del ratón.
  . Dibujo de la malla de puntos.
  . Borrar un dispositivo.
  . Redibujar la pantalla de trabajo.
  . Activar el menú del archivo de datos.
  . Almacenar el sistema hidráulico.
  . Recuperar el sistema hidráulico.
  . Escalar la pantalla de trabajo.
  . Salir del sistema de captura.
  . Actualizar la posición del ratón.
  .
  .
  . Etc.

Por omisión siempre se detectarán los eventos de presión de algunos de los botones del ratón. Por otro lado, los eventos de movimiento del ratón y levantar (soltar) un botón del ratón se notificarán al sistema bajo solicitud expresa.

28. Ratón.

- Dibujarse.
- Solicita la actualización de las coordenadas x,y
- Cambiar su posición.

- Cambiar la representación gráfica a:
  . Modo normal
  . Modo de cruz
  . Modo de texto
  . Modo de ícono
    . Bomba
    . Línea de conducción
    . Nudo
    . Tanque

29. Menú.

- Formar un menú con una base, título, y un conjunto de botones (opciones).

30. Menú de archivo de datos.

- Presentar un menú de archivo de datos.
- Permitir la elección de alguno de sus elementos.

31. Menú de escalamiento.

- Presentar un menú de escalamiento.
- Solicitar escalamientos de la pantalla de trabajo de acuerdo al factor de escalación seleccionado.

32. Pantalla de trabajo.

- Notificar al kernel sus limites actuales.
- Solicitar la coordenada x.
- Solicitar la coordenada y.
- Actualizar las coordenadas (x,y).
- Notificar al sistema las nuevas coordenadas (x,y).
- Solicitar color actual.
- Solicitar el nombre del archivo de dibujo de los dispositivos hidráulicos.
- Dibujar el dispositivo en cuestión.
- Solicitar el apuntador al sistema hidráulico.
- Dibujar el sistema hidráulico.
- Calcular los incrementos (deltaX, deltaY) del movimiento del ratón.
- Enviar los incrementos calculados al kernel junto con la bandera de salto.
- Realizar escalamientos en el plano.
- Dibujarse
- Desplegar una malla de puntos.
- Borrar la malla de puntos.

33. Renglón de la tabla.

- Formar un conjunto de botones de lectura de diversos tipos de datos.
- Dibujarse

34. Sistema hidráulico.

- Contener la relación y la dependencia de la red hidráulica entre si.

- Construir la red hidráulica como una colección de dispositivos (aumentar un nuevo dispositivo a la colección).
- Construirse a través de un archivo de datos solicitando la recuperación de cada uno de los dispositivos hidráulicos que lo componen.
- Solicitar un redibujo del sistema hidráulico en la pantalla de trabajo.
- Borrar un nuevo dispositivo a la colección y destruirlo.
- Solicitar el almacenamiento del dispositivo en cuestión.
- Proporciona al cliente su apuntador.
- Dibujarse.

35. Tabla de datos.

- Permitir consultar y actualizar la información de los dispositivos hidráulicos.
- Realizar operaciones de adición, borrado, inserción y movimiento de renglones en la tabla de datos.
- Permitir la interrupción de cualquier comando de la tabla de datos.
- Realizar el duplicado de datos de un dispositivo a otro.
- Hacer un "scroll" (viajar a lo largo de la información almacenada en tablas) siempre y cuando halla más renglones en la tabla que los que se muestran por default:
  . Hacer el recorrido página por página de la tabla hacia adelante o hacia a tras.
  . Hacer el recorrido de la tabla hacia adelante renglón por renglón.
  . Hacer el recorrido de la tabla hacia a tras renglón por renglón.
- Dibujarse.
- Activar y desactivar la barra de desplazamiento

36. Tabla de datos de la bomba.

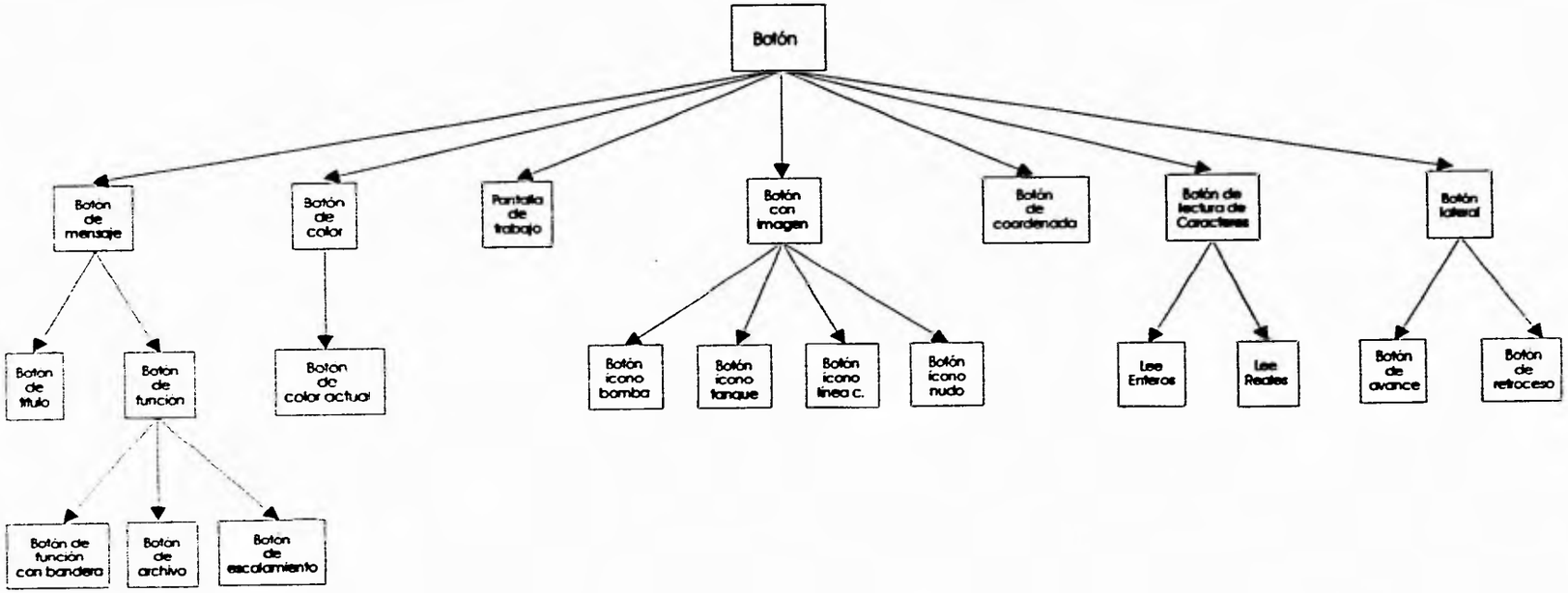
- Crear una tabla de datos con información propia del dispositivo hidráulico bomba.

36. Tabla de datos de la línea de conducción.

- Crear una tabla de datos con información propia del dispositivo hidráulico línea de conducción.

Después de haber analizado las acciones del sistema en sus distintas etapas y haber asignado responsabilidades a las clases, agrupando clases con responsabilidades comunes y creando nuevas clases o desechando clases existentes; se presentan finalmente las jerarquías de clases del sistema de captura gráfico para redes hidráulicas en las figuras IV.4 y IV.5.

Jerarquía del botón:



Jerarquía de dispositivo:

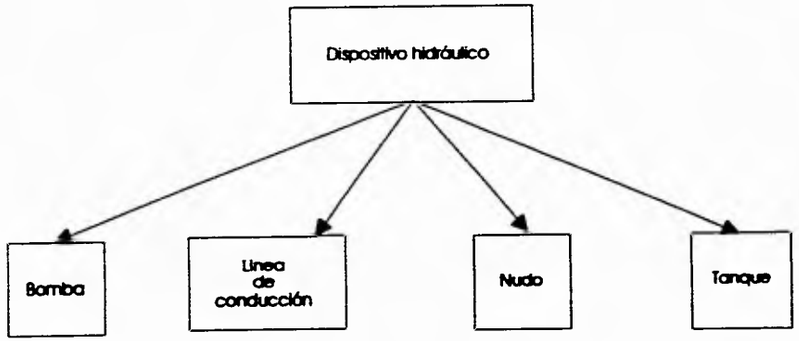


Figura IV.4 Jerarquías de la clase botón y de la clase dispositivo.

ESTA TESIS NO DEBE
SALIR DE LA BIBLIOTECA

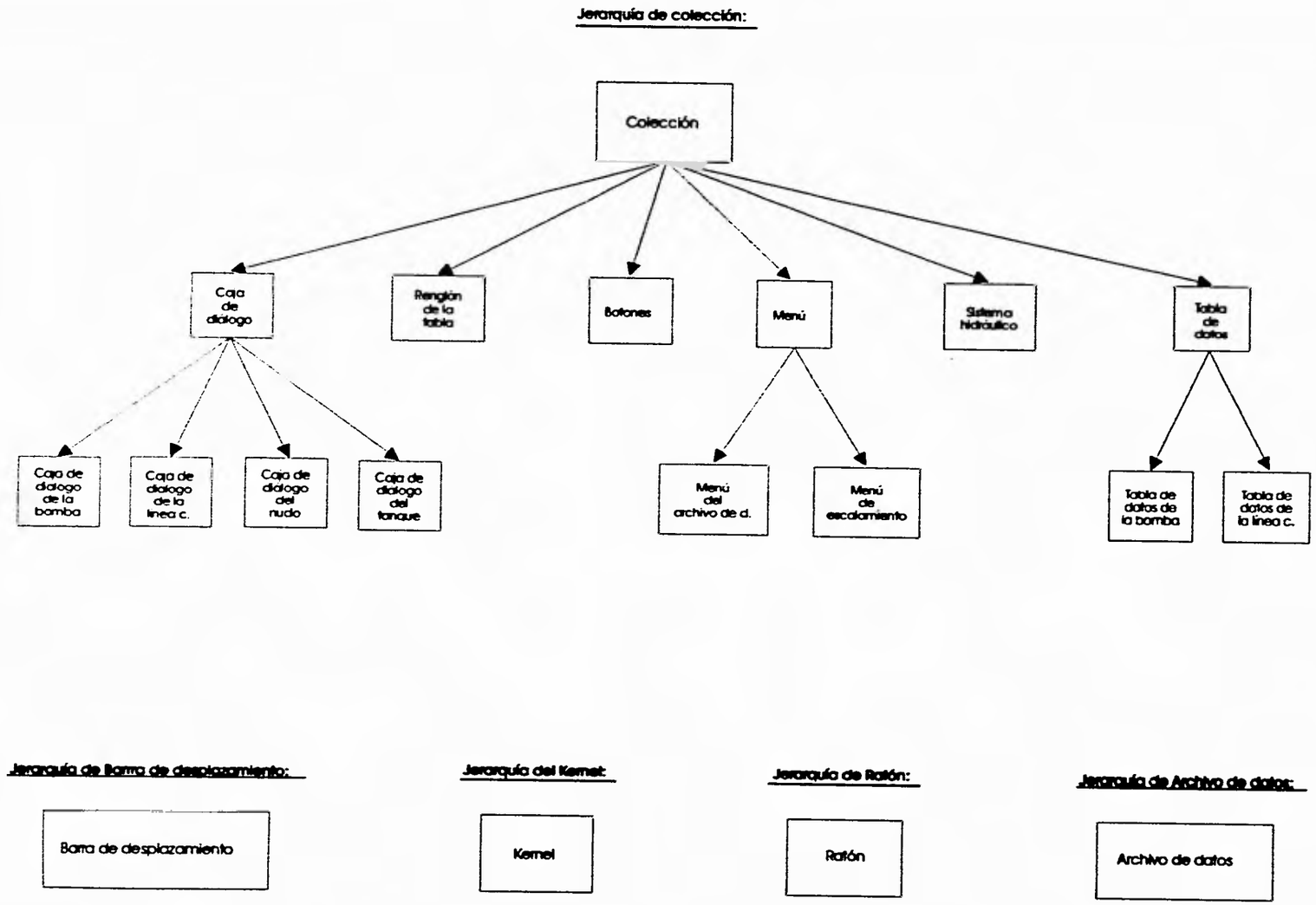


Figura IV.5 Jerarquías de las clases colección, barra de desplazamiento, kernel, ratón y archivo de datos

V. IMPLEMENTACIÓN.

La implementación del sistema se llevó a cabo en Borland Pascal 7.0. Como datos diremos que el código tiene aproximadamente 9,000 líneas y el programa ejecutable tiene un tamaño total de 117,632 bytes. Las rutinas de manejo del ratón se escribieron en ensamblador, con el objeto de mejorar el rendimiento del sistema. Se han omitido en el trabajo escrito los listados del sistema, los cuales están disposición del interesado. Incluimos, solo como referencia el directorio de archivos que conforman el sistema, así como el listado del programa principal comentado.

Archivos que conforman el sistema de captura de información:

```
ACUEDUCT PAS          7.530 07/03/94   11:39
APLICACI PAS          4.332 21/11/94   14:45
BMP_IMG  PAS          4.884 29/09/94   12:00
BOMBA    IMG          1.568 09/11/93   12:50
BOMBA    WRD            480 08/11/94   16:17
BOTONES  PAS          4.842 17/11/94   17:49
CAMPO    PAS          4.141 21/11/94   13:04
CLTMOUSE PAS          1.306 21/11/94   15:59
COORDS   INC          3.529 21/11/94   16:50
DISPFORM PAS          6.837 21/11/94   19:39
DISPOSIT PAS          9.784 07/03/94   11:26
DISPTABL PAS          5.844 21/11/94   21:11
DRVBOTON PAS         37.737 18/11/94   17:34
DRVCLIEN PAS          1.955 02/11/94   15:33
DRVCONST PAS            839 16/11/94   17:30
DRVDISP  PAS         18.793 21/11/94   20:11
DRVKERNE PAS         12.772 15/11/94   17:01
DRVMENU  PAS          6.853 21/11/94   16:02
DRVNODO  PAS            172 06/09/94   17:36
DRVPLANO PAS         14.292 22/11/94    8:37
DRVSIST  PAS         14.303 24/11/94   12:58
EVENTOS  PAS          2.634 16/11/94   13:58
FORMA    PAS          5.845 21/11/94   21:48
HANDLER  ASM          3.214 03/10/94   12:31
HDUMMY   ASM            114 29/09/94   14:14
```

```
LINEA     IMG          1.544 26/10/94   16:41
LISTACOO  PAS            373 05/11/94   11:09
LISTAS    PAS          8.692 10/11/94   11:13
MISC      PAS          2.355 18/11/94   17:20
MISCBOT   PAS          5.383 21/11/94   20:22
MODELO    PAS          1.871 21/11/94   17:48
MOUSEBOX  PAS         15.061 21/11/94   17:02
NUDO      IMG          1.544 26/10/94   17:27
NUDO      WRD            136 14/11/94   11:24
OBJ       PAS            501 02/11/94    9:13
OBJMOUSE  PAS          5.513 18/11/94   17:10
OKEY      INC          6.057 21/11/94   18:55
PATCHGR   PAS            544 24/09/94   10:16
RENGLON   PAS          8.070 18/11/94   17:22
TABLA     PAS         23.742 21/11/94   13:05
TANQUE    IMG          1.544 26/10/94   16:44
TANQUE    WRD            222 08/11/94   16:50
```

Listado del archivo correspondiente al programa principal

```
uses
    Objects,ObjMouse,MouseBox,
    DrvKernel,DrvCliente,
    Eventos,Misc,MiscBot,
    DrvBoton,Botones,DrvMenu,Forma,DrvPlano,DrvDisp,DrvSist;

{ OBJETOS BASICOS RVIENE EN EL SISTEMA }

Var
  MouseB            : PMouseBox;
  PantallaTrab      : PPlano;
  Icono             : PBotonIcono;
  BotonesIcono      : PBotones;
  BotonComando      : PBoton;
  Comandos          : PBotones;
  Paleta            : PPaletaColores;
  BColorAct         : PBotonColorActual;
  BCoordX           : PBotonCoord;
  BCoordY           : PBotonCoord;
  SistemaHidraulico : PSistema;
  ZoomMenu          : PMenuZoom;
  FileMenu          : PMenuArchivo;

{CREACION DE LOS ICONOS DE LOS DISPOSITIVOS A MODELAR }

Procedure CreaIconos;
begin
BotonesIcono := New(PBotones,Init(LightCyan,Black,577,10,53,70,1,DePress,DoubleLine));
Icono := New(PBotonIcono,Init(LightGray,0,0,0,0,0,0,Press, SingleLine,'Bomba',SolCrearBomba));
BotonesIcono^.AddBoton(PBoton(Icono));
Icono := New(PBotonIcono,Init(LightGray,0,0,0,0,0,0,Press, SingleLine,'Tanque',SolCrearTanque));
BotonesIcono^.AddBoton(PBoton(Icono));
Icono := New(PBotonIcono,Init(LightGray,0,0,0,0,0,0,Press, SingleLine,'Linea',SolCrearLinea));
BotonesIcono^.AddBoton(PBoton(Icono));
Icono := New(PBotonIcono,Init(LightGray,0,0,0,0,0,0,Press, SingleLine,'Nudo',SolCrearNudo));
BotonesIcono^.AddBoton(PBoton(Icono));
end;

{CREACION DEL MENÚ DE COMANDOS DEL SISTEMA}

Procedure CreaComandos;
begin
Comandos :=New(PBotones,Init(LightCyan,Black,10,428,55,20,3,DePress,DoubleLine));
BotonComando := New(PBotonFun,Init(LightGray,0,0,Blue,0,0,0,0,False,False,'ARCHIVO',ActivaArchivo));
Comandos^.AddBoton(PBoton(BotonComando));
BotonComando := New(PBotonFun,Init(LightGray,0,0,Blue,0,0,0,0,False,False,'ESCALAR',Bzoom));
```

```
Comandos^.AddBoton(PBoton(BotonComando));
BotonComando := New(PBotPan,Init(LightGray,0,0,Blue,0,0,0,0,False,False,'RECORRER',Bcorrimiento));
Comandos^.AddBoton(PBoton(BotonComando));
BotonComando := New(PBotonFun,Init(LightGray,0,0,Blue,0,0,0,0,False,False,'TEXTO',Btexto));
Comandos^.AddBoton(PBoton(BotonComando));
BotonComando:= New(PBotonFunFlag,Init(LightGray,0,0,blue,0,0,0,0,Press,DoubleLine,'MALLA',malla,true));
Comandos^.AddBoton(PBoton(BotonComando));
BotonComando:= New(PBotonFunFlag,Init(LightGray,0,0,blue,0,0,0,0,Press,DoubleLine,'SALTOS',salto,True));
Comandos^.AddBoton(PBoton(BotonComando));
end;

{ CREACION DE LOS MENUS POP-UP                                    }
{ QUE SON ACTIVADOS MEDIANTE LA SELECCION DE ALGUNA OPCION}

Procedure CrearMenus;
begin
ZoomMenu := New(PMenuZoom,Init(LightBlue,LightCyan,black,LightCyan,white,50,50,150,Press,SingleLine,
    'ESCALAMIENTO',BZoom));
FileMenu := New(PMenuArchivo,Init(LightBlue,LightCyan,black,LightCyan,white,50,50,150,Press,SingleLine
    ,'ARCHIVO',ActivaArchivo));
end;


{ PROGRAMA PRINCIPAL }

begin

{ENTRADA A MODO GRÁFICO}

ModoGrafico;

{ CREACION DEL OBJETO MOUSE}

MouseB := New(PMouseBox,Init);
MouseB^.SetBoxLimits(11,11,569,415);

{INICIALIZACION DEL MANEJADOR DE EVENTOS}

Kernel.Init;
Kernel.SetMouse(PMouse(MouseB),True);
Kernel.SetMouseEvent(EMLP+EMRP+EMCP);


{CREACIÓN DE OBJETOS QUE COMPONEN LA APLICACIÓN}

CreaIconos;
CreaComandos;
CrearMenus;

SistemaHidraulico := New(PSistema,Init);

PantallaTrab := New(PPlano,Init(10,10,557,406));
PantallaTrab^.ChangeW(0,0,50,40);
PantallaTrab^.Desplegar;

BotonesIcono^.Desplegar;
Comandos^.Desplegar;

Paleta := New(PPaletaColores,Init(270,428));
Paleta^.Crear;
Paleta^.Desplegar;

BColorAct := New(PBotonColorActual,Init(Blue,LightCyan,Black,505,427,20,41,DePress,SingleLine));
BColorAct^.Desplegar;

BCoordX := New(PBotonCoord,Init(LightGray,LightCyan,Black,black,blue,540,427,90,20,Depress, SingleLine,
            'X:= ','0.0',ActualizaCoordWordX));
BCoordX^.Desplegar;
BCoordY := New(PBotonCoord,Init(LightGray,LightCyan,Black,black,blue,540,448,90,20,Depress, SingleLine,
            'Y:= ','0.0',ActualizaCoordWordY));

BCoordY^.Desplegar;
```

```
{LOS OBJETOS SE DAN DE ALTA EN EL KERNEL PARA RECIBIR MENSAJES}

Kernel.AddCliente(PCliente(PantallaTrab));
Kernel.AddCliente(PCliente(SistemaHidraulico));
Kernel.AddCliente(PCliente(BotonesIcono));
Kernel.AddCliente(PCliente(BColorAct));
Kernel.AddCliente(PCliente(Paleta));
Kernel.AddCliente(PCliente(Comandos));
Kernel.AddCliente(PCliente(BCoordX));
Kernel.AddCliente(PCliente(BCoordY));
Kernel.AddCliente(PCliente(ZoomMenu));
Kernel.AddCliente(PCliente(FileMenu));
Kernel.AddCliente(PCliente(MouseB));

{EL KERNEL TOMA EL CONTROL DEL SISTEMA RECIBIENDO Y ENVIANDO MENSAJES}
{HASTA RECIBIR EL MENSAJE DE TERMINACIÓN DE APLICACION              }

Kernel.Run;

{DESTRUCCIÓN DE LOS OBJETOS}

Kernel.Done;

{SALIDA DE MODO GRAFICO}

ExitModoGrafico;
end.
```

VI. CONCLUSIONES

A continuación se presenta a manera de conclusión algunos comentarios resultantes de la experiencia del desarrollo de la presente tesis.

1. Al inicio del desarrollo de la presente tesis, se tenía que decidir si se desarrollaba basado en Tecnología Orientada a Objetos (TOO) o, como tradicionalmente, en forma estructurada. La base de la decisión era: ¿Realmente la Tecnología Orientada a Objetos ofrece ventajas con respecto al Enfoque Estructurado (EE) ?, de ser así , en la práctica ¿cuáles son?. Para contestar esta pregunta decidimos desarrollar una pequeña parte del sistema en ambos enfoques. Se seleccionó la interfase gráfica correspondiente al manejo de botones : Crear un botón, mostrarlo, borrarlo, presionarlo con el mouse, derivar con base en un botón básico otros. Los resultados de este desarrollo mostraron lo siguiente:

   Los programas basados en el EE requirieron aproximadamente el 50% más de código que los basados en TOO. Tómese en cuenta cuando se desarrolló en TOO mis conocimientos en esta área eran básicos, por lo que es posible que este porcentaje bajé aún más.

   La velocidad de operación de los mismos era de aproximadamente 10% al 20% mayor en el caso de TOO (evaluación hecha por el director de la presente tesis mediante la herramienta Turbo Profiler de Borland Pascal)

   En términos de extensibilidad la TOO demostró una mayor flexibilidad al tratar de generar nuevos tipos de botones. Los mecanismos de herencia y polimorfismo ofrecidos en TOO permitieron modificar solo aquellas partes que fueran diferentes de los botones ya existentes.

   Respecto a la depuración y mantenimiento la TOO demostró que los errores se propagaban a las clases derivadas, sin embargo, muchas veces corregir un problema en la clase base corregía muchos otros en las clases derivadas. Lo que nos mostró que antes de liberar una clase era necesario haberla probado suficientemente.

2. En el caso de los dispositivos simulados, agregar otro es muy sencillo, porque el comportamiento de ellos se heredaría a los nuevos dispositivos.

3. Para desarrollar el sistema fue necesario crear un mecanismo de comunicación entre los distintos objetos. Este mecanismo se encarga de recibir mensajes provenientes del usuario (presión de una tecla, movimiento del mouse, etc...) o de otros objetos. En este sentido se decidió desarrollar un manejador de eventos. Su desarrollo facilitó en mucho el diseño e implementación del sistema. En la etapa de prueba este mecanismo permitió probar fácilmente en forma aislada el comportamiento de los objetos.

4. Sin embargo todo este capital, obtenido del desarrollo orientado a objetos, tubo un costo. Para desarrollar e implementar este sistema me fue muy difícil al principio entender el paradigma orientado a objetos ya que es otra forma de trabajar muy diferente al paradigma estructural. Es sin lugar a dudas, otra forma de pensar, en orientación a objetos es necesario pensar en que cosa hacen los objetos y como funcionan internamente; y no pensar en forma estructurada, basados en que operaciones son necesarias aplicar a los datos para obtener el fin deseado.

5. La autora desarrolló el sistema con base en un curso de posgrado donde tuvo por primera vez contacto con los métodos de análisis y diseño orientados a objeto, sin embargo este no es el único método y mucho menos el mejor, al paso del tiempo y con más estudios no damos cuenta que el sistema es perfectible y que tal vez el diseño del mismo no es ni con mucho el mejor.

6. Finalmente es opinión de la autora que la TOO no es una moda, es en realidad un tendencia con 25 años de desarrollo. Sin embargo, la gente debe entender que TOO no es simplemente aprender una nueva sintaxis de un nuevo lenguaje o modificaciones a uno ya existente. TOO es mucho más, es cambiar la forma de análisis, diseño e implementación de sistemas, donde una gran cantidad del tiempo es invertida en análisis y diseño más que en implementación. La gente debe entender que bajo este enfoque no es posible sentarse frente a la computadora sin antes haber desarrollado completamente el diseño del sistema.

7. Por último, el Enfoque Estructurado sigue vigente, al menos mientras la TOO no este madura, se dispongan de sistemas operativos, compiladores y ambientes 100% orientados a objetos. O mientras el tipo de programas que se deseé hacer no sean grandes.

BIBLIOGRAFIA

Object-Oriented Analysis and Design.
Primera edición
Grady Booch
Rational
Santa Clara, California.

Hierarchical Object-Oriented Desing.
Peter J. Robinson
Prentice Hall.

Borland Pascal With Objects
Programmer's reference
Borland International, 1992

ObjectWindows
Programming guide
Borland International, 1992

Hidráulica general
Volumen 1 Fundamentos
Gilberto Sotelo Avila
Ed. Limusa