DISEÑO DE UN SOFTWARE PARA EL DISEÑO Y COMPROBACIÓN DE SISTEMAS DE CONTRAINCENDIO CON AGUA DE MAR Y ACHIQUE PARA TODO TIPO DE EMBARCACIÓN MARÍTIMA Y FLUVIAL

Osorio del Valle, Cristina

Pardo Alvarez, Raul Antonio

Quintana Alvarez, Moises

Director

Universidad Tecnológica de Bolívar

Ingeniería de Sistemas y Computación

Cartagena de Indias

2002

CONTENIDO

INDICE DE TABLAS

Pág.

INDICE DE FIGURAS

Pág.

AGRADECIMIENTOS

A mis Padres: Por todo el apoyo incondicional en mi formación como ser humano, por la confianza que me ha brindado en todo aquello que hago, por sus consejos y su cariño incondicional. **Gracias.**

Hernán Osorio Camacho , Martha Lucía De Osorio

A mi Hermanito: Por olvidar que somos personas muy diferentes y brindarme su cariño . **Gracias.**

Hernán Osorio Del Valle

A Mario: Por ser mi amigo , por creer en mi, por apoyarme en los momentos difíciles y hacer alegres los ratitos que compartimos juntos. **Gracias.**

A Rau: Por ser mi compañero en la realización de este proyecto; por acompañarme y ser mi apoyo ante todos los problemas y dificultades que se presentaron en el camino. Gracias

Raul Antonio Pardo

A mi famia: Por brindarme su apoyo incondicional y su cariño.

Gracias

Raul Pardo Padilla, Rosario Alvarez, Raul Jaime Pardo, Raul Nicanor Pardo

A Luisa Maria : Por iluminar el camino. **Gracias.**

A Cristina: Por su paciencia y colaboración. **Gracias**

Cristina Osorio Del Valle

A Nuestro director: Por ser nuestro guía en este camino.

Gracias.

Moisés Quintana Alvarez

Al personal de la dirección de proyectos de Cotecmar: Por brindarnos la información y asesoría necesaria para realizar este proyecto. **Gracias**

TN. Oscar Tascon, TN Ricardo Lugo, Ingeniero Gaster Pacheco, TN Victor Jimenez

GLOSARIO

ACHICAR: Extraer el agua de sentinas.

BABOR: Lado izquierdo de un buque.

COLECTOR PRINCIPAL: Tubo madre (principal) de un sistema de tuberías.

COMPARTIMIENTO: Subdivisiones por mamparos del casco de la nave.

ESLORA: Medida del largo del buque (de popa a proa)

ESTRIBOR: Lado derecho de un buque.

MANGA: Medida del ancho del buque (de estribor a babor)

MAMPARO: termino aplicado a las paredes de división verticales las cuales subdividen en interior del barco en compartimentos o cuartos.

POPA: Parte trasera de un buque.

PROA: Parte delantera de un buque

SENTINA: Parte baja de los mamparos, tanques o espacios de maquinaria donde puede acumularse agua.

SISTEMA DE ACHIQUE: Sistema de tuberías localizado en sentinas y conectado a bombas. Este sistema es usado para bombear acumulaciones de agua hacia fuera de un buque.

CRISTINA OSORIO DEL VALLE

RAUL ANTONIO PARDO ALVAREZ

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR INSTITUCIÓN

UNIVERSITARIA

FACULTAD DE INGENIERÍA DE SISTEMAS

CARTAGENA D.T. Y C.

2002

DISEÑO DE UN SOFTWARE PARA EL DISEÑO Y COMPROBACIÓN DE SISTEMAS DE CONTRAINCENDIO CON AGUA DE MAR Y ACHIQUE PARA TODO TIPO DE EMBARCACIÓN MARÍTIMA Y FLUVIAL

CRISTINA OSORIO DEL VALLE

RAUL ANTONIO PARDO ALVAREZ

Proyecto de grado presentado como requisito para optar al titulo de ingeniero de sistema

Director

MOISÉS QUINTANA ALVAREZ
Magíster en Informática Aplicada

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR INSTITUCIÓN

UNIVERSITARIA

FACULTAD DE INGENIERÍA DE SISTEMAS

CARTAGENA D.T. Y C.

2002

Cartagena de Indias, 16 de mayo del 2002

Señores
COMITÉ PROYECTO DE GRADO
Facultad de Ingeniería de Sistemas
La ciudad

Distinguidos señores.

Por medio de la presente me permito informales que el proyecto de grado titulado **"DISEÑO DE UN SOFTWARE PARA EL DISEÑO Y COMPROBACIÓN DE SISTEMAS DE CONTRAINCENDIO CON AGUA DE MAR Y ACHIQUE PARA TODO TIPO DE EMBARCACIÓN MARÍTIMA Y FLUVIAL"** ha sido desarrollado de acuerdo a los objetivos establecidos.

Como director del proyecto considero que el trabajo es satisfactorio y amerita ser presentado por sus autores.

Cordialmente,

MOISÉS QUINTANA ALVAREZ

Cartagena de Indias, 16 de mayo del 2002

Señores
COMITÉ PROYECTO DE GRADO
Facultad de Ingeniería de Sistemas

La ciudad

Distinguidos señores.

Por medio de la presente me permito informales que el proyecto de grado titulado **“DISEÑO DE UN SOFTWARE PARA EL DISEÑO Y COMPROBACIÓN DE SISTEMAS DE CONTRAINCENDIO CON AGUA DE MAR Y ACHIQUE PARA TODO TIPO DE EMBARCACIÓN MARÍTIMA Y FLUVIAL”** ha sido desarrollado de acuerdo a los objetivos establecidos.

Como autores del proyecto consideramos que el trabajo es satisfactorio y amerita ser presentado ante ustedes.

Cordialmente,

CRISTINA OSORIO DEL VALLE **RAUL ANTONIO PARDO**

Articulo 107.

La Universidad Tecnológica de Bolívar, se reserva el derecho de propiedad intelectual de todos los trabajos de grado aprobados y no pueden ser explotados comercialmente sin su autorización.

NOTA DE ACEPTACIÓN

_____

_____

Presidente del Jurado

Jurado

Jurado

Cartagena de Indias D. T. y C. 29 de abril de 2002

BIBLIOGRAFÍA

GIARRATANO & RILEY. Sistemas Expertos - Principios y Programación -. 3º Edición. Madrid: Editorial Int. Thomson, 2000. 356 p.

SILBERSCHATZ, Abraham. KORTH, Henry F. SUDARSHAN, S. Fundamentos de bases de datos. 3º Edición. Madrid : Editorial Mc Graw Hill, 1998. 641 p.

WHITTEN, Jeffrey L. BENTLEY, Lonnie D. BARLOW, Victor M. Análisis y Diseño de sistemas de información. . 3º Edición. Madrid : Editorial Mc Graw Hill, 1998. 907 p.

http://www.fortunecity.com/skyscraper/romrow/207/se/Portada.html

http://www.gsi.dit.upm.es/~cif/cursos/ssii/clipshtml/clips-index.html

http://www.medinlab.ehu.es/ehsis/Espanol.shtml

http://www.ghg.net/clips/CLIPS.html

REGLAMENTACIONES:

American Bureau of Shipping, rules for building and classing steel vessels under 61 meters in length (1983).

Bureau Veritas. Rules and regulations for the classification of ships. (1996).

Marine Engineering, society of naval architects and marine engineerig. (1992), editor Roy L. Harrington, Newport new shipbuilding.

Normas y Standards ASTM.

SOLAS. Edición refundida del convenio internacional para la seguridad de la vida humana en el mar 1974 y su protocolo de 1978: artículos, anexos y certificados. Incorpora todas las enmiendas hasta 1990 inclusive. Edición 1992. Publicada por la organización marítima internacional.

INTRODUCCIÓN

Las organizaciones han reconocido, desde hace mucho, la importancia de administrar recursos principales tales como la mano de obra y las materias primas. La información se ha colocado en lugar adecuado como recurso principal. Los tomadores de decisiones están comenzando a comprender que la información no solo es un subproducto de la conducción, sino que a la vez

alimenta a los negocios y puede ser el factor crítico para la determinación del éxito o fracaso de éstos.

Para maximizar la utilidad de la información, en un negocio, se debe manejar correctamente tal como maneja los demás recursos. Los administradores necesitan comprender que hay costos asociados con la producción, distribución, seguridad, almacenamiento y recuperación de toda información. Aunque la info rmación se encuentra a nuestro alrededor ésta no es gratis, y su uso es estratégico para posicionar la competitividad de un negocio.

La fácil disponibilidad de computadoras ha creado una explosión de información a través de la sociedad en general y de los negocios en particular. El manejo de información generada por computadoras difiere en forma significativa del manejo de los datos producidos manualmente.

Por otro lado es común encontrar empresas que no emplean las oportunidades que la tecnología de la computación ofrece. Las ventajas que se ofrecen en ella, son por ejemplo, el tener información guardada en una base de datos para un mejor manejo de la misma, utilizar una red de computadoras para la transferencia de información, emplear sistemas de información para poder acceder de manera adecuada a la información y tomar así una mejor decisión.

Todas estas facilidades que nos brindan la computación y la administración de la información por medio de ella; sirven para permitir que el manejo de información se realice con facilidad y rapidez, ahorrando tiempo y dinero, para

apoyar al ejecutivo en la toma de decisiones estratégicas y competitivas para el desarrollo satisfactorio de una organización productiva.

RESUMEN

La Corporación de Ciencia y Tecnología Para el Desarrollo de la Industrial Naval, Marítima y Fluvial (COTECMAR) le ha permitido al país recuperar la ventaja estratégica que implica poder realizar, con autonomía el mantenimiento de sus buques (incluso los de mayor tamaño), sin tener que trasladarlos a otros países, además de aportar la capacidad tecnológica para prestar este servicio a todas la embarcaciones extranjeras que ingresen a nuestras costas. Según el análisis realizado por la máxima dirección de COTECMAR, para lograr el objetivo antes planteado es necesario favorecer que el proceso productivo de la empresa se realice empleando los métodos más eficaces y sustentados sobre bases científicas y tecnológicas actuales.

La forma en que se realiza en la actualidad el diseño del sistema contraincendio y de achique de los buques, requiere de un cambio tecnológico, pues el proceso se realiza de forma manual, con el riesgo que la imprecisión humana pueda retrasar o dañar el producto esperado. Además, el tiempo que se invierte por el personal altamente calificado en el análisis bibliográfico de las normas y el ulterior ajuste de los datos a las características de estas normas y del resto del diseño del buque, puede ser realizado por un software apropiado que será manipulado por una persona que no requiera de tan alta calificación.

Al momento de verificar los cumplimientos de las normas de diseño, los ingenieros se basan en documentos y libros que tienen fácilmente accesibles, pero resulta muy tedioso y cuesta mucho tiempo revisar entre tantos detalles tan complejos que implica un diseño de tuberías como lo son materiales de tuberías, de accesorios (bridas o válvulas por poner un ejemplo), o especificaciones de diseños establecidas por casas de estándares como la ANSI y en la mayoría de los casos estas especificaciones son complejas y difíciles de memorizar, luego requeriría de demasiada experiencia para hacer un diseño sin consultar las especificaciones y reglas para el mismo. Incluso por mucha experiencia que se tenga no estaría de más una consulta a dichas reglamentaciones considerando que se trata de la seguridad de una embarcación entera y en consecuencia de la tripulación y los pasajeros mismos. Todo este tiempo representa una demora en la elaboración de las embarcaciones y como toda demora representa falta de ingresos en un corto plazo.

Este trabajo busca diseñar un software que permita la optimización del proceso de diseño, en términos de disminución de tiempo de respuesta al cliente y ejecución del mismo, con el fin de lograr un proceso de construcción o reparación de buques con claridad y rapidez.

1. DESCRIPCIÓN DEL PROYECTO

En este capítulo se describe: el objetivo general de la investigación, los objetivos específicos, el alcance y las limitaciones del sistema, así como el software y el hardware a utilizar para el desarrollo del mismo.

1.1 DEFINICIÓN DEL PROBLEMA Y SU AMBIENTE.

La Corporación de Ciencia y Tecnología para el Desarrollo de la Industria Naval, Marítima y Fluvial (COTECMAR) le ha permitido al país recuperar la ventaja estratégica que implica poder realizar, con autonomía de sus buques (incluso los de mayor tamaño), sin tener que trasladarlos a otros países, además de aportar la capacidad tecnológica para prestar este servicio a todas las embarcaciones extranjeras que ingresen a nuestras costas. Pero más allá de las funciones operativas y de su objeto de abrir las puertas al mercado internacional al servicio de mantenimiento de buques, COTECMAR cumplirá una tarea fundamental para el progreso de la industria naval del país: promoverá la investigación científica y el desarrollo tecnológico en esta área.

Según el análisis realizado por la máxima dirección de COTECMAR, para lograr el objetivo antes planteado es necesario favorecer que el proceso productivo de la empresa se realice empleando los métodos más eficaces y sustentados sobre bases científicas y tecnológicas actuales.

La forma en que se realiza en la actualidad el diseño del sistema de contraincendio con agua de mar y de achique de los buques, requiere un cambio tecnológico, pues el proceso se realiza de forma manual, con el riesgo que la imprecisión humana puede retrasar o dañar el producto esperado. Además, el tiempo que se invierte por el personal altamente calificado en el análisis bibliográfico de las normas y el ulterior ajuste de los datos a las características de estas normas y del resto del diseño del buque, puede ser realizado por un software apropiado que será manipulado por una persona que no requiera de tan alta calificación. Sin duda alguna, esto traerá una disminución en el costo de mano de obra y en una mejora en cuanto a tiempo de respuesta al cliente se refiere.

Al momento de verificar los cumplimientos de las normas de diseño, los ingenieros se basan en documentos y libros que tienen fácilmente accesibles, pero resulta muy tedioso y cuesta mucho tiempo revisar entre tantos detalles tan complejos que implica un diseño de tuberías (como lo son los materiales de tuberías), o de accesorios y en la mayoría de los casos estas especificaciones son complejas y difíciles de memorizar, luego requeriría de demasiada experiencia para hacer un diseño sin consultar las especificaciones y reglas para el mismo, incluso por mucha experiencia que se tenga no estaría de más una consulta a dichas reglamentaciones considerando que se trata de la seguridad de una embarcación entera y en consecuencia de la tripulación y los pasajeros mismos. Todo este tiempo representa una demora en la elaboración del buque y como toda demora, representa falta de ingresos en un corto plazo.

Dada la importancia que tiene este procedimiento y los beneficios que traerá su optimización, esta investigación está orientada a realizar al análisis y diseño de una herramienta informática que, condensando normas y criterios de Casa Clasificadoras aprobadas por la asociación

Internacional de Sociedades de Clasificación , para el diseño y comprobación de sistemas de contraincendio mediante agua de mar y sistemas de achique, en todo tipo de embarcación.

1.2 OBJETIVO GENERAL

Diseñar un Software que permita la optimización del proceso del diseño, en términos de disminución de tiempo de respuesta al cliente y ejecución del mismo, con el fin de lograr un proceso de construcción o reparación de buques con calidad y rapidez.

1.3 OBJETIVOS ESPECÍFICOS

- Profundizar en el conocimiento del personal de COTECMAR y del grupo de investigación en lo relacionado con los equipos y servicios del buque.

- Suministrar una fuente de información alterna que indique las variables y los criterios, en que se basan las casas clasificadoras y reglamentaciones, para utilizar en el diseño de la aplicación.

- Diseñar e implementar una base de datos óptima, que abarque todas las normas existentes para sistemas de contraincendio y achique, y que sea suficientemente flexible par modificaciones.

- Diseñar una herramienta muy sencilla, fácil de usar y que minimice el tiempo de comprobación de normas en los sistemas de contraincendio con agua de mar y de achique, y de diseño de los mismos en embarcaciones nuevas.

1.4 JUSTIFICACIÓN DE LA INVESTIGACIÓN

Esta investigación está orientada al desarrollo de una herramienta informática que, condensando normas y criterios de la Organización Marítima internacional y otras casas clasificadoras aprobadas por la Asociación Internacional de Sociedades de Clasificación (IACS) para el diseño de sistemas de contraincendio mediante agua de mar y sistemas de achique, en todo tipo de embarcaciones, permite ágilmente:

- Obtener la cantidad, calidad, dimensiones y disposición de todos los componentes asociados con los sistemas.
- Ajustar el diseño preliminar en construcciones nuevas.
- Comprobar diseños existentes y optimizarlos mediante alteraciones y/o modificaciones.
- Obtener la información preliminar requerida para el costeo.

Con el desarrollo de esta herramienta informática se pretende incrementar la productividad y disminución de costos en el proceso de construcción o reparación de embarcaciones al reducir considerablemente el tiempo de diseño del proceso, así como preparar las condiciones para el desarrollo de otros proyectos en esta dirección, haciendo que la empresa sea más competitiva con respecto a sus similares en otros países.

2. PROCEDIMIENTO DE DISEÑO Y COMPROBACIÓN DE SISTEMAS DE CONTRAINCENDIO CON AGUA DE MAR Y ACHIQUE

2.1 GENERALIDADES DE COTECMAR.

La Corporación de Ciencia y Tecnología para el Desarrollo de la Industria Naval, Marítima y Fluvial (Cotecmar), es una empresa de reciente creación dedicada a la construcción y reparación de buques . La experiencia de los empleados y directivos de esta empresa provienen del trabajo desarrollado en el antiguo Astillero Naval de la Armada Nacional.

El Astillero Naval se disuelve como empresa estatal y con el apoyo de un grupo de socios fundadores se decide crear una nueva empresa privada, que además de realizar la actividad productiva antes mencionada, se convirtiera en un centro de investigación y desarrollo de proyectos de construcción y reparación naval. En estas circunstancias surge Cotecmar.

Cotecmar propende por la investigación científica e innovación tecnológica para el desarrollo de la industria naval colombiana y conexas, mejorando su productividad y competitividad, comprometida con una cultura de calidad que respeta el medio ambiente, propicia la prestación de servicios a precios justos y el desarrollo personal y profesional de los integrantes de la

Organización, garantizando el soporte técnico que el país requiere para el mantenimiento y fortalecimiento de su flota naval, marítima y fluvial.

Cotecmar busca ser la organización líder en la investigación científica e innovación tecnológica, comprometida con el desarrollo del poder marítimo nacional, en el campo de la industria naval, marítima y fluvial.

2.2 DESCRIPCIÓN DEL PROCEDIMIENTO.

Existen criterios o normas de diseño que debe cumplir todo buque para garantizar la seguridad de la estructura (el buque como tal) y de su tripulación.

Al momento de verificar el cumplimiento de estos criterios los diseñadores se basan en documentos y libros que poseen en su biblioteca (fácilmente accesibles) pero resulta muy tedioso y cuesta mucho tiempo revisar todos y cada uno de los detalles.

Cuando los diseñadores tienen las dimensiones del buque se realizar un chequeo preliminar del tamaño y la localización relativa de los espacios para que el recorrido de las tuberías (ya sea del sistema de contraincendio o de achique) sea acomodado.

Para el caso de sistema de contraincendio con agua de mar, el diseñador debe identificar la longitud de la eslora entre perpendiculares, la manga de la nave y el puntal trazado del buque medido hasta la cubierta de cierre. Además, identificar el tipo de buque (buque de pasajeros o buque de carga) y cual es su arqueo bruto. Para el caso de sistemas de achique, el diseñador identifica en que compartimentos se instalará bocas de succión y de estos se toma su eslora, además de las medidas que se especificaron para el diseño de sistemas de contraincendio con agua de mar.

Basándose en las normas, pueden dar un primer concepto de que dimensiones tendrá el sistema (de contraincendio o achique), cuantas y en que lugares se ubicarán las diferentes bombas, que accesorios deben utilizarse, que velocidad debe alcanzar que circulará por dicho sistema, etc. Cada norma da un resultado diferentes (en algunos casos coinciden con ciertos criterios) y queda a juicio de los diseñadores que norma aplicar. En la selección de los componentes del sistema (bombas, mangueras, hidrantes, etc.) influye la misión del buque (si en de pasajeros o de carga, por ejemplo), tamaño, perfil de operación (comercial, por ejemplo), y otros factores (peso, arqueo bruto, etc.). En algunas ocasiones no es posible satisfacer todos los requerimientos exactos pero se debe seleccionar los suficientes para que el diseño sea óptimo. Un ejemplo de los resultado que da la casa clasificadora SOLAS para el diseño de sistemas de Contraincendio con agua de mar y la solución que da BUREAU VERITAS para sistemas de achique:

Ejemplo de SOLAS

Buque de Pasajeros

NÚMERO DE BOMBAS

Buques con arqueo bruto >= 4000 toneladas	Mínimo 3
Buques con arqueo bruto < 4000 toneladas	Mínimo 2
Buques con arqueo bruto >= 1000 toneladas	1 bomba fija *

* Bomba adicional de accionamiento independiente cuya capacidad será mayor o igual al 40% de la capacidad total de las bombas.

CAPACIDAD DE LAS BOMBAS

La capacidad de las bombas de contraincendio debe ser 2/3 del caudal de la bomba de sentina. Cada una de las bombas tendrá una capacidad no inferior al 80% de la capacidad total exigida

dividida entre el número mínimo de bombas. En todo caso no será menor a 25 m^3/h.

PRESIÓN DE BOMBAS CONTRAINCENDIO

Buques con arqueo bruto >= 4000 toneladas 0.31 N/mm^2

Buques con arqueo bruto >= 1000 y < 4000 toneladas 0.27 N/mm^2

Buques de carga

NÚMERO DE BOMBAS

Buques con arqueo bruto >= 1000 toneladas Mínimo 2

Buques con arqueo bruto < 1000 toneladas Juicio Diseñador

Buques con arqueo bruto >= 2 000 toneladas 1 bomba fija*

*Bomba adicional de accionamiento independiente cuya capacidad será mayor o igual al 40% de la capacidad total de las bombas.

CAPACIDAD DE LAS BOMBAS

La capacidad de las bombas de contraincendio debe exceder al menos en 1/3 al caudal de la bomba de sentina. Cada una de las bombas tendrá una capacidad no inferior al 80% de la capacidad total exigida dividida entre el número mínimo de bombas, en todo caso no será menor a 25 m^3/h.

PRESIÓN DE BOMBAS CONTRAINCENDIO

Buques con arqueo bruto >= 6000 toneladas 0.27 N/mm^2

Buques con arqueo bruto >1000 <=6000 toneladas 0.25 N/mm^2

Ejemplo de BUREAU VERITAS

Buques de pasajeros

DIÁMETRO DE TUBERIAS

Diámetro del colector principal (en mm) .d = 25 + 1.68 $\sqrt{L(B+D)}$ (se toma el valor comercial más próximo a la formula)

Diámetro de ramales (en mm)

Este diámetro debe ser mayor a 50 mm y menor de 100 mm.

.d = 25 + 2.16 $\sqrt{c(B+D)}$ (se toma el valor comercial más próximo a la formula)

VELOCIDADES DE FLUIDO

Velocidad mínima del fluido en el colector principal debe ser 2 m/s para esloras mayor a 35m y de 1.22 m/s para esloras menor a 35m.

NUMERO DE BOMBAS

Debe haber mínimo 3 bombas de achique y si el buque tiene coeficiente de criterio (Cs) mayor o igual a 30 debe haber además

una bomba independiente.

Si la eslora es mayor o igual a 91.5 m o Cs mayor a 30 debe haber por lo menos una bomba motorizada.

CAPACIDAD DE BOMAS

La capacidad de la bomba de achique debe ser igual a 0.345 d^2

Buque de carga

DIÁMETRO DE TUBERIAS

Diámetro del colector principal (en mm) .d = 25 + 1.68 $\sqrt{L(B+D)}$ (se toma el valor comercial más próximo a la formula)

Diámetro de ramales (en mm)

Este diámetro debe ser mayor a 50 mm y menor de 100 mm.

.d = 25 + 2.16 $\sqrt{c(B+D)}$ (se toma el valor comercial más próximo a la formula)

VELOCIDADES DE FLUIDO

Velocidad mínima del fluido en el colector principal debe ser 2 m/s para esloras mayor a 35m y de 1.22 m/s para esloras menor a 35m.

NUMERO DE BOMBAS

Debe haber mínimo 2 bombas de achique y si el buque tiene coeficiente de criterio (Cs) mayor o igual a 30 debe haber además una bomba independiente.

CAPACIDAD DE BOMBAS

La capacidad de la bomba de achique debe ser igual a 0.345 d^2 .

Si la capacidad de una o más bombas es menor a la capacidad que indica la regla debe compensarse con un aumento en la capacidad de otra(s) bomba(s). La deficiencia no debe exceder el 30% de la capacidad de la regla.

Entonces se elabora un diseño preliminar del tubo principal y sus componentes obteniendo estimados preliminares de presión, flujo y temperatura. Los parámetros del sistema pueden cambiar varias veces durante la fase de este diseño. Durante este diseño se desarrolla detalles adicionales para cada sistema.

Debe especificarse los requerimientos mínimos de presión, temperatura, flujo, etc., para cada sistema; el número, capacidad y localización de los componentes; materiales para las tuberías

y sus especificaciones. Para la elección de estos materiales se tienen en cuentas otras normas (ASTM, ANSI, etc.) que los clasifican de acuerdo al fluido, el sistema (para nuestro caso, contraincendio o achique) y el componente. Por ejemplo, si se elige ASTM para escoger el material de un sistema de contrai ncendio con agua de mar y el accesorio (o componente) es *tubería sin costura* , el resultado es el siguiente:

Norma utilizada: ASTM

Sistema de tubería: Contraincendio

Fluido: Agua de mar

Accesorio: Tubería sin costura

Material: CNA 90:10

Especificación de material: ASME SB466

Especificación de diseño: ASME SB466

Con el diseño preliminar del sistema y los materiales para cada uno de los componentes se calcula las pérdidas con una herramienta informática (Flow of Fluids). Si las pérdidas son muy altas se juega con el diseño (hasta donde lo permita los requerimiento mínimos que exigen las normas) hasta que las pérdidas sean aceptables.

2.2.1 Procedimiento de achique: el procedimiento es el siguiente;

1. Tomar geometría del buque : Para el diseño del sistema de achique debemos conocer las siguientes medidas del buque:

- Eslora entre perpendiculares (en metros)
- Manga de la nave (en metros)

- Puntal trazado del buque medido hasta la cubierta de cierre (en metros)
- Eslora de los compartimentos a achicar

2. Identificar el tipo de buque y su arqueo bruto: Se debe identificar si el diseño del sistema de achique se hará para un buque de pasajeros o para un buque de carga. Además conocer cual es el arqueo bruto del buque.

3. Elegir norma : Existen diferentes normas (Solas, Bureau Veritas, U.S. Navy, etc.) para obtener diámetros de tuberías (Colector principal y ramales), No. De Bombas y su ubicación, caudal de bombas, etc. Se debe escoger una de estas normas para hacer el diseño, si la norma no especifica para ciertos ítem (No. De mangueras, por ejemplo) este campo quedará a juicio del diseñador.

RESULTADOS:

SOLAS

Buque de pasajeros:

Mínimo 3 bombas de sentina en el colector principal

Una de las bombas exigidas debe ser de tipo sumergible.

Coeficiente de criterio (Cs) Mayor o igual a 30 además una bomba motorizada independiente.

Cada bomba debe estar ubicada de tal forma que una sea capaz de acudir en cualquier tipo de inundación.

Eslora mayor o igual a 91.5m por lo menos una bomba motorizada que quepa utilizar en todas las condiciones de inundación que el buque deba poder afrontar.

Si las máquinas propulsoras, máquinas auxiliares y calderas se encuentran ubicadas en diferentes compartimientos se debe tratar de repartir las bombas por estos compartimientos.

Se instalará conductos laterales de aspiración salvo en compartimientos estrechos situados en los extremos en los extremos del buque, en los que cabrá considerar que basta con un solo conducto de aspiración.

Diámetro mínimo del colector principal (en mm) .d = 25 + 1.68 $\sqrt{L(B+D)}$

Diámetros mínimos de ramales (en mm) .d = 25 + 2.16 $\sqrt{c(B+D)}$

Velocidad mínima del fluido en el colector principal 2 m/s.

Buque de carga

Mínimo 2 bombas de sentina en el colector p rincipal

Diámetro del colector principal .d = 25 + 1.68 $\sqrt{L(B+D)}$

Diámetros de ramales .d = 25 + 2.16 $\sqrt{c(B+D)}$

Velocidad mínima del fluido en el colector principal 2 m/s.

LLOYDS REGISTER

Buque de pasajeros

Mínimo 3 bombas de las cuales una debe ir conectada al colector principal. Si Cs es > 30 debe haber una bomba independiente.

Capacidad de bomba $Q = 5.75/10^3 \ d^2$

Diámetro mínimo del colector principal (en mm) $d = 25 + 1.68\sqrt{L(B+D)}$. Se podrá aproximar máximo 5mm.

Diámetros mínimos de ramales (en mm) $d = 25 + 2.16\sqrt{c(B+D)}$. Se podrá aproximar 5 mm.

Velocidad mínima del fluido en el colector principal 122/60 m/s

No debe haber más de dos succiones en cada espacio. Cada succión en cada lado del espacio.

Buque de carga

Mínimo 2 bombas de achique.

Si la eslora del buque es mayor a 90m una de las bombas debe trabajar independientemente.

Capacidad de bomba $Q = 5.75/10^3 \ d^2$

Diámetro mínimo del colector principal (en mm) .d= 25 + 168 $\sqrt{L(B+D)}$. Se podrá aproximar máximo 5mm.

Diámetros mínimos de ramales (en mm) $d = 25 + 2.15\sqrt{c(B+D)}$

Se podrá aproximar 5 mm.

Velocidad mínima del fluido en el colector principal 122/60 m/s. Esta velocidad puede disminuir hasta en un 30% si se compensa por un exceso de capacidad en otro equipo.

BUERAU VERITAS

Buque de pasajeros

Mínimo 3 bombas de achique

No debe haber más de 2 succiones en un mismo espacio.

Coeficiente de criterio (Cs) Mayor o igual a 30 además una bomba independiente.

Cada bomba debe estar ubicada de tal forma que una sea capaz de acudir en cualquier tipo de inundación.

Eslora mayor o igual a 91.5m o Cs > 30 por lo menos una bomba motorizada que quepa utilizar en todas las condiciones de inundación que el buque deba poder afrontar.

Si las máquinas propulsoras, máquinas auxiliares y calderas se encuentran ubicadas en diferentes compartimientos se debe tratar de repartir las bombas por estos compartimientos.

Capacidad de bomba $Q = 0.345\ d^2$

Diámetro del colector principal (en mm) $.d = 25 + 1.68 \sqrt{L(B+D)}$

Se toma el valor comercial más próximo a la formula.

Diámetros de ramales (en mm) $.d = 25 + 2.16 \sqrt{c(B+D)}$

Se toma el valor comercial más próximo a la formula.

Debe ser mayor a 50mm y menor de 100mm

Velocidad mínima del fluido en el colector principal 2 m/s para eslora > 35m.

Velocidad mínima del fluido en el colector principal 1.22 m/s para eslora < 35m .

Buque de carga

Mínimo 2 Bombas de achique.

Si Cs > 30 debe haber una bomba adicional.

Capacidad de bomba $Q = 0.345\ d^2$

Si la capacidad de una o más bombas es menor que la capacidad que indica la regla debe compensarse con un aumento en la capacidad de otra (s) bomba(s). La deficiencia no debe exceder el 30% de la capacidad de la regla.

Diámetro del colector principal (en mm) .$d = 25 + 1.68\sqrt{L(B+D)}$

Se toma el valor comercial más próximo a la formula.

Diámetros de ramales (en mm) .$d = 25 + 2.16\sqrt{c(B+D)}$

Se toma el valor comercial más próximo a la formula.

Debe ser mayor a 50mm y menor de 100mm

Velocidad mínima del fluido en el colector principal 1.22 m/s para eslora < 35m.

Velocidad mínima del fluido en el colector principal 2 m/s para eslora > 35m. Esta velocidad puede disminuir hasta en un 20% si se compensa por un exceso de capacidad en otro equipo.

2.2.2 Procedimiento para calcular sistemas de contraincendio con agua de mar: Estos son los pasos a seguir;

1. Tomar geometría del buque : Para el diseño del sistema de achique debemos conocer las siguientes medidas del buque:

- Eslora entre perpendiculares (en metros)
- Manga de la nave (en metros)
- Puntal trazado del buque medido hasta la cubierta de cierre (en metros)

2. Identificar el tipo de buque y su arqueo bruto: Se debe identificar si el diseño del sistema de achique se hará para un buque de pasajeros o para un buque de carga. Además conocer cual es el arqueo bruto del buque.

3. Elegir norma : Existen diferentes normas (Solas, Bureau Veritas, U.S. Navy, etc.) para obtener diámetros de tuberías , No. De Bombas y su ubicación, caudal de bombas, etc. Se debe escoger una de estas normas para hacer el diseño, si la norma no especifica para ciertos ítem (No . De mangueras, por ejemplo) este campo quedará a juicio del diseñador.

RESULTADOS:

SOLAS

Buque de pasajeros

Para buques con arqueo bruto mayor o igual a 4000 toneladas debe tener como mín. 3 Bombas.

Para buques con arqueo bruto menor a 4000 toneladas debe tener como mín. 2 Bombas.

Para buques con arqueo bruto mayor o igual a 1000 toneladas debe tener además una bomba fija de emergencia de accionamiento independiente cuya capacidad será mayor o igual al 40% de la capacidad total de las bombas.

CAPACIDAD DE LAS BOMBAS

La capacidad de las bombas de contraincendio debe ser 2/3 del caudal de la bomba de sentina.

Cada una de las bombas tendrá una capacidad no inferior al 80% de la capacidad total exigida dividida entre el No. Mín de bombas. En todo caso la capacidad no será menor a 25 m^3 / h.

PRESION DE BOMBAS CONTRAINCENDIO

Para buques con arqueo bruto mayor o igual a 4000 toneladas la presión debe ser 0.31 N/mm^2

Para buques con arqueo bruto mayor o igual a 1000 toneladas y menor a 4000 toneladas debe ser 0.27 $N/mm^{2.}$

Buque de cargas

Para buques con arqueo bruto mayor o igual a 1000 toneladas debe tener como mín. 2 Bombas.

Para buques con arqueo bruto menor a 1000 toneladas según juzgue la administración.
La capacidad de las bombas de contraincendio en este tipo de buques debe ser menor o igual a 180 m^3/h.

Para buques con arqueo bruto mayor o igual a 2000 toneladas debe tener además una bomba fija de emergencia de accionamiento independiente cuya capacidad será mayor o igual al 40% de la capacidad total de las bombas.

CAPACIDAD DE LAS BOMBAS

La capacidad de las bombas de contraincendio debe exceder al menos en 1/3 al caudal de la bomba de sentina.

Cada una de las bombas tendrá una capacidad no inferior al 80% de la capacidad total exigida dividida entre el No. Mín de bombas. En todo caso la capacidad no será menor a 25 m^3 / h.

PRESION DE BOMBAS CONTRAINCENDIO

Para buques con arqueo bruto mayor o igual a 6000 toneladas la presión debe ser 0.27 N/mm^2

Para buques con arqueo bruto mayor o igual a 1000 toneladas y menor a 6000 toneladas debe ser 0.25 $N/mm^{2.}$

NUMERO DE MANGUERAS

Para un buque con arqueo bruto mayor o igual a 1000 toneladas debe haber 1 manguera por cada 30 metros de eslora que en todo caso no será menos de 5 mangueras.

MARINE ENGINEERING

La velocidad mínima que debe alcanzar el agua de mar en tuberías debe ser 3 fps.
Las velocidades especificadas por esta reglamentación se ven en la Tabla 1.

Tabla 1. Velocidades Marine Engineering

Fluido	Velocidad	Velocidad Máxima
Agua de mar (succión)	$3\sqrt{d}$	**12 fps (9 fps cuando las tuberías son de acero galvanizado)**
Agua de mar (descarga)	$5\sqrt{d}$	**12 fps (9 fps cuando las tuberías son de acero galvanizado)**

MATERIALES

El material para los sistemas de contraincendio seco que sugiere esta norma es carbon steel.

El material para los sistemas de contraincendio cooling que sugiere esta norma es 90-10 cooper-niquel o GRP .

U.S NAVY

El número de bombas contra incendio a ser instalado es determinado dividiendo el total de capacidad requerida entre la capacidad de una bomba candidata, y redondeando al entero más próximo.

El número de bombas resultantes es multiplicado por un factor (generalmente 1.33) y redondeado otra vez para proveer bombas extras. Dichas bombas extras pueden ser utilizadas en casos de daños en batallas o cuando se hagan mantenimientos, para mantener una presión de agua continua en el sistema.

El número y capacidad de bombas puede ser ajustado para acomodarse a un número de bombas convenientes de acuerdo a la disponibilidad de locaciones en el barco.

Al menos dos bombas contra incendios deben estar en una embarcación de la armada de los Estados Unidos; combatientes, tales como cruceros típicos requieren cinco o más, y los más grandes porta aviones usan más de veinte.

CAPACIDAD DE LAS BOMBAS

Las bombas para estos sistemas están primariamente manejadas por motor eléctrico; sin embargo, en las embarcaciones de maquinaria con plantas de vapor, las bombas funcionan con turbinas de vapor. Las bombas de motor tienen un rango de capacidad de 100 a 1000 gpm (galones por minuto), siendo 1000 gpm una talla preferida. La capacidad de las turbinas a vapor es usualmente de 2000 gpm. En promedio las bombas están en un rango de 125 a 175 psi, siendo las presiones más altas usadas en embarcaciones más grandes.

Las bombas están distribuidas en compartimientos separados a lo largo de la embarcación para minimizar el efecto de un accidente pequeño. Cada bomba debe tener una caja de mar independiente.

La capacidad total de las bombas contraincendio deberá ser suficiente para el suministro simultaneo de todos los equipos para combatir el fuego en una situación de caso extremo, además de enfriamiento vital, servicio y chorros suficientes. Debido a que las naves norteamericanas usan eductores de drenaje interactuando con el sistema contra incendios, el flujo de agua utilizado en un combate contra fuego debe ser controlado por el drenaje para mantener la estabilidad de la embarcación. La carga total en el peor de los casos es calculada analizando las bajas que se podría tener de acuerdo a la misión del buque (De pronto un helicóptero que se estrelle contra la nave), o daños en combate en una zona de alto riesgo

(como cerca de un depósito de misiles) para determinar los sistemas rociadores. Para algunas naves, una carga de no-bajas, puede gobernar la capacidad del sistema.

Debido a que las bombas en las naves norteamericanas pueden operar a baja o cero carga por períodos extensos, cada bomba debería tener una línea de recirculación estimada en un 3% del flujo promedio para prevenir sobrecalentamiento.

CONFIGURACION GENERAL DEL SISTEMA

Los sistemas contraincendio en las naves Estadounidenses están distribuidos de tal manera que puedan sobrevivir daños en combate en su máximo extensión. En embarcaciones más pequeñas, un sistema principal sencillo debería estar instalado aproximadamente cerca de la línea de centro para tomar ventaja de la estructura del buque para protección. Cuando el tamaño del buque es suficiente para proveer separación efectiva dos o más tuberías principales deberían ser proveídas y separadas bien sea horizontal o verticalmente y tan lejos como sea posible. Estas tuberías se conectarán formando anillos de acuerdo a las configuraciones escritas al principio de este capitulo. El sistema estará provisto de válvulas que permitan que se opere en un rango de zonas independientes, cada una suministrada por una o más bombas , y así permitir el aislamiento de zonas comprometidas. Los sistemas vitales de rociamiento deberían estar configurados para suministrar servicio a dos zonas diferentes.

El control remoto de bombas contra incendio y la segregación clave de válvulas es proveído para permitir rápida respuesta en una emergencia.

2.3 NORMAS TÉCNICAS Y/O REGLAMENTARIAS

Actualmente la dirección de proyectos cuenta con las siguientes normas y reglamentaciones para hacer los diseños y comprobaciones de los sistemas de contraincendio con agua de mar y achique. Esta documentación se encuentra en la biblioteca de la dirección de proyectos.

- Solas.
- Bureau Veritas.
- Marine Engineering
- Lloyd´s Register
- ABS
- U.S Navy
- Reglamentaciones y estándares ASTM

2.4 PROCEDIMIENTOS ANTERIORES Y POSTERIORES.

El procedimiento anterior al descrito en este capitulo es la identificación de la geometría del buque. Se pueden presentar dos casos para el diseño de sistemas de contraincendio con agua de mar y achique:

1. Cuando el sistema se diseña para una embarcación nueva.
2. Cuando el sistema se diseña para una embarcación ya construida.

Para el primer caso, la geometría del buque es tomada de los planos elaborados en herramientas informáticas (CAD) que están a disposición de los diseñadores.

Para embarcaciones ya construidas, se debe identificar los componentes del sistema y del recorrido de tuberías existentes.

El procedimiento que se realiza posterior al diseño de sistemas de contraincendio y achique (y en general, de sistemas de tuberías) es la construcción del sistema sobre la estructura del buque.

El cargo responsable de que el procedimiento que estudiamos en este capitulo sea llevado a cabo con calidad es el jefe del departamento de maquinaria naval y propulsión.

3. TEORIA SOBRE SISTEMAS EXPERTOS

3.1 DEFINICIÓN DE SISTEMA EXPERTO.

Un Sistema Experto se puede definir como un sistema computacional interactivo que permite la creación de bases de conocimiento, las cuales una vez cargadas responden a preguntas, despejan dudas y sugieren cursos de acción emulando / simulando el proceso de razonamiento de un experto para resolver problemas en un área específica del conocimiento humano.

De esta definición se desprenden las dos habilidades fundamentales que poseen los Sistemas Expertos:

- Habilidad de aprendizaje.

- Habilidad para simular el proceso de razonamiento humano.

La habilidad de aprendizaje requiere la interacción de un experto en alguna rama específica del saber y un ingeniero de conocimiento, que se encarga de traducir este conocimiento del experto a reglas heurísticas para formar la base de conocimiento.

La habilidad para imitar el razonamiento que posee el Sistemas Experto se desprende de " caminar " a lo largo de las reglas heurísticas introducidas o enseñadas al sistema por un experto, a través del proceso de aprendizaje durante la carga o generación de las bases del conocimiento.

En una situación ideal, un sistema experto se comporta en la misma forma que lo haría un experto humano en la materia sobre la que se ha construido el sistema, presentando ciertas ventajas respecto al humano: su experiencia es permanente, el conocimiento que contiene es fácil de actualizar, es capaz de utilizar grandes cantidades de conocimiento, son fáciles de duplicar, son consistentes, son más baratos y son documentables.

Sin embargo, hay ocasiones en que el comportamiento del experto humano supera el rendimiento que pueda proporcionar un sistema experto, especialmente si los problemas que se deben resolver necesitan creatividad, imaginación o sentido común.

Además, el sistema no se debe limitar a proponer soluciones, sino que debe poder mostrar cómo ha llegado a la solución y dar respuestas a preguntas del tipo ¿Qué pasaría sí...?.

3.2 DIFERENCIAS ENTRE UN PROGRAMA CONVENCIONAL Y UN SISTEMA EXPERTO

Las diferencias que existen entre un sistema convencional y un Sistema Experto las hemos condensado en la Tabla 2

Tabla 2 Diferencias entre sistema convencional y Sistema Experto

SISTEMA CONVENCIONAL	SISTEMA EXPERTO
Conocimiento y lógica de proceso mezclados en un programa	Base de conocimiento y mecanismo de inferencia separados
No explican porqué se necesitan los datos ni porqué se llegó a un resultado Es difícil efectuar cambios en los conocimientos programados	Lo explican Es más fácil modificar la base de conocimientos
Necesitan información completa para operar	Deben ser más tolerantes para operar aún con alguna información desconocida
Generalmente manejan datos cuantitativos	Maneja datos cualitativos primordialmente
Captura, amplifica y distribuye el acceso a datos numéricos o textuales	Captura, amplifica y distribuye el acceso a juicios basados en conocimientos

3.3 COMPONENTES DE UN SISTEMA EXPERTO

Una característica decisiva de los Sistemas Expertos es la separación entre conocimiento (reglas, hechos) por un lado y su procesamiento por el otro. A ello se añade una interface de usuario y un componente explicativo.

A continuación mostraremos una descripción de cada uno de los componentes:

3.3.1 La base de conocimientos: La Base de conocimientos contiene todos los hechos, las reglas y los procedimientos del dominio de aplicación que son importantes para la solución del problema. Para entender la base de conocimientos explicaremos con un pequeño ejemplo.

Los hechos son del tipo: El barco a vela "Cony " tiene una longitud de 6m. La representación de este conocimiento puede realizarse orientándola, por ejemplo, según objetos. Los objetos de una base de conocimientos pueden ser entonces: barco, barco a motor, barco a vela. Estos objetos están relacionados de tal forma que un barco a vela tiene todas las cualidades de un barco, y además todas las cualidades específicas de un barco a vela. Todas las cualidades de un barco, por ejemplo: Desplazamiento sobre el agua, vienen descritas con el "barco". A través de la relación formulada, el barco a vela hereda estas cualidades, de forma que sólo hará falta describir sus cualidades particulares.

Este tipo de programación se define como programación orientada a objetos y se utiliza con frecuencia en el desarrollo de los Sistemas Expertos. Puede darse el caso de que determinados procesos y funciones deban subordinarse a unos objetos en particular, por ejemplo la velocidad como función de la

fuerza y la dirección del viento. La velocidad se determinará en función de los datos particulares.

Cómo se lleva a cabo la clasificación en grupos de las características y de los procedimientos alrededor de un objeto con las técnicas de programación, y cómo deben ser las relaciones entre los objetos pueden variar mucho de aplicación a aplicación.

Junto a estos objetos, la base de conocimientos dispone de reglas. Estas reglas se representan en forma de:

Si *premisas* Entonces *Conclusión y/o Acción*

En las premisas se solicitan vinculaciones lógicas referentes a las cualidades de los objetos.

En la conclusión se añaden nuevos hechos y cualidades a la base de conocimientos y/o se ejecutan acciones. Esto se define a menudo como programación orientada a reglas.

Base de conocimiento vs Bases de datos.

El conocimiento puede ser complejo, pero trabaja con generalizaciones, se refiere más a tipos de entidades que a ejemplares individuales.

La complejidad de las clase de cosas que se desea registrar en una Base de Conocimiento (B.C.) es típicamente mayor que la que se almacena en una Base de Datos (B.D.):

- En una B.C. hay que manejar información incompleta, en una B.D. se pretende que existan todos los campos.

- Las B.C. deben expresar relaciones entre clases genéricas.

- Las B.D. no trabajan con lo que significan o implican sus datos. Las B.C. deben trabajar también respecto a las consecuencias lógicas.

- Muchos Sistemas Expertos poseen más relaciones que ejemplares; en las B.D. ocurre lo contrario.

3.3.2 El mecanismo de inferencia: El mecanismo de inferencia es la unidad lógica con la que se extraen conclusiones de la base de conocimientos, según un método fijo de solución de problemas que esta configurado imitando el procedimiento humano de los expertos para solucionar problemas.

Una conclusión se produce mediante aplicación de las reglas sobre los hechos presentes.

Ejemplo:

Una Regla es: Si p y q entonces r

p y q son justo aquellos hechos que se mencionan en la cláusula "si" de la regla, es decir, las condiciones para la aplicabilidad de la regla. Aplicar la regla es: deducir de los hechos p y q el hecho r.

En un Sistema Experto existirá un hecho sólo cuando esté contenido en la base de conocimientos.

Los hechos que constan en la cláusula "si" se llaman *premisas*, y el contenido en la cláusula "entonces" se llama *conclusión*. Cuando se aplica una regla sobre algunos hechos cualesquiera se dice que se *dispara*. El disparo de una regla provoca la inserción del nuevo hecho en la base de conocimientos.

Las funciones del mecanismo de inferencia son:

1. Determinación de las acciones que tendrán lugar, el orden en que lo harán y cómo lo harán entre las diferentes partes del Sistema Experto.

2. Determinar cómo y cuándo se procesarán las reglas, y dado el caso también la elección de qué reglas deberán procesarse.

3. Control del diálogo con el usuario.

La decisión sobre los mecanismos de procesamiento de reglas, es decir, qué estrategias de búsqueda se implementarán, es de vital importancia para la efectividad del sistema en su conjunto.

Ante problemas o clases de problemas distintos se estructuran, como es lógico, diferentes mecanismos de inferencia. El mecanismo de inferencia debe de estar "adaptado" al problema a solucionar. Una imposición de dinero exige, bajo ciertas circunstancias, una estrategia distinta de procesamiento del conocimiento que un diagnóstico de fallos de máquina.

3.3.3 El componente explicativo: Las soluciones descubiertas por los expertos deber poder ser repetibles tanto por el ingeniero del conocimiento en la fase de comprobación así como por el usuario. La exactitud de los resultados sólo podrá ser controlada, naturalmente, por los expertos.

Siempre es deseable que durante el trabajo de desarrollo del sistema se conozca el grado de progreso en el procesamiento del problema. Como os he dicho en anterioridad nos pueden surgir unas preguntas como las siguientes:

- ¿Qué preguntas se plantean y por qué?
- ¿Cómo ha llegado el sistema a soluciones intermedias?
- ¿Qué cualidades tienen los distintos objetos?

A pesar de insistir sobre la importancia del componente explicativo es muy difícil y hasta ahora no se han conseguido cumplir todos los requisitos de un buen componente explicativo.

Muchos representan el progreso de la consulta al sistema de forma gráfica. Además los componentes explicativos intentan justificar su función rastreando hacia atrás el camino de la solución. Aunque encontrar la forma de representar finalmente en un texto lo suficientemente inteligible las relaciones encontradas depara las mayores dificultades. Los componentes explicativos pueden ser suficientes para el ingeniero del conocimiento, ya que está muy familiarizado con el entorno del procesamiento de datos, y a veces bastan también para el experto; pero para el usuario , que a menudo desconoce las sutilezas del procesamiento de datos, los componentes explicativos existentes son todavía poco satisfactorios.

3.3.4 La interface de usuario: En este componente como todos bien saben es la forma en la que el sistema se nos presentará ante el usuario. Como en los anteriores nos surgen dudas y preguntas como por ejemplo:

- ¿Cómo debe responder el usuario a las preguntas planteadas?
- ¿Cómo saldrán las respuestas del sistema a las preguntas que se le planteen?
- ¿Qué informaciones se representarán de forma gráfica?

Los requisitos o características de la interface que presentaremos al usuario voy a resumirlas en cuatro, que a mi opinión son las más importantes y las más a tener en cuenta al desarrollar el sistema:

1. El aprendizaje del manejo debe ser rápido. El usuario no debe dedicar mucho tiempo al manejo del sistema , debe ser intuitivo, fácil en su manejo. No debemos olvidar que nuestro sistema simula al comportamiento de un experto. Debe sernos cómodo y relativamente sencillo en cuanto al manejo.

2. Debe evitarse en lo posible la entrada de datos errónea. Ejemplo: Poneros en la situación de que nuestro sistema a un médico. Cuando nosotros acudimos a un médico, le contamos y detallamos nuestros síntomas y el con sus preguntas junto con nuestras respuestas nos diagnostica nuestra "enfermedad".Imaginaros que acudimos a un medico y le decimos que nos

duele una pierna en lugar de un brazo, el diagnostico será inútil. El ejemplo es muy exagerado pero demuestra la importancia en la correcta introducción de los datos al sistema.

3. Los resultados deben presentarse en una forma clara para el usuario. Vuelvo al ejemplo del médico. Si nuestro médico nos diagnostica un medicamento pero en nuestra receta no nos escribe cada cuantas horas hemos de tomarlo por ejemplo, por muy bueno que sea el medicamento, la solución a nuestro problema será ineficiente por completo. Por eso se insiste en que los resultados debe ser claros y concisos.

4. Las preguntas y explicaciones deben ser comprensibles.

3.3.5 El componente de adquisición: Un buen componente de adquisición ayudará considerablemente la labor del Ingeniero del Conocimiento. Este puede concentrarse principalmente en la estructuración del conocimiento sin tener que dedicar tanto tiempo en la actividad de programación. Como hice en el campo de la interface , daremos unas reglas o requisitos para la realización de nuestro componente de adquisición.

Requisitos o características del componente de adquisición:

1. El conocimiento, es decir, las reglas, los hechos, las relaciones entre los hechos, etc.., debe poder introducirse de la forma más sencilla posible.
2. Posibilidades de representación clara de todas las informaciones contenidas en una base de conocimientos.
3. Comprobación automática de la sintaxis.
4. Posibilidad constante de acceso al lenguaje de programación.

Como se pone en práctica cada uno de los requisitos dependerá del lenguaje de programación elegido y del hardware que tengamos. El experto deberá estar algo familiarizado con el componente de adquisición para poder realizar modificaciones por sí sólo.

3.4 BENEFICIOS EN EL USO DE SISTEMAS EXPERTOS.

- La utilización de los Sistemas Expertos en las organizaciones pueden traer varios beneficios, los cuales serán comentados a continuación:

- *REDUCCIÓN EN LA DEPENDENCIA DE PERONAL CLAVE*. Esto se debe a que los conocimientos del personal especializado son retenidos durante el proceso de aprendizaje, y están listos para ser utilizados por diferentes personas. Esto es útil cuando la experiencia es escasa o costosa, o bien, cuando los expertos no se encuentran disponibles para la solución de un problema en particular.

- *FACILITA EL ENTRENAMIENTO DEL PERSONAL* . El Sistema Expertos puede ayudar de manera importante, y a costo menor, a la capacitación y adiestramiento del personal sin experiencia.

- *MEJORA EN LA CALIDAD Y EFICIENCIA EN EL PROCESO DE LA TOMA DE DECISIONES*. Lo anterior implica que las decisiones podrán tomarse de una forma más ágil con el apoyo de un Sistema Experto. Incuso, las decisiones podrán ser consistentes al presentarse situaciones equivalentes. Esto significa que un Sistema Expertos responderá siempre de la misma forma ante las mismas situaciones, lo cual no necesariamente ocurre con las personas.

- *TRANFERENCIA DE LA CAPACIDAD DE DECISIONES*. Un Sistema Experto puede facilitar la descentralización de datos en el proceso de la toma de decisiones en aquellos casos que se consideren convenientes. Así, el conocimiento de un experto puede transferirse a varias personas, de tal forma que las decisiones sean tomadas en el nivel más bajo.

3.5 DESARROLLO DE LOS SISTEMAS EXPERTOS.

Para desarrollar el software primero conocemos el equipo de gente necesario, después los métodos que utiliza ese equipo de gente y por último como prueban y construyen prototipos de software para terminar en el sistema final.

3.5.1 El Equipo de desarrollo: Las personas que componen un grupo o un equipo, como en todos los ámbitos deben cumplir unas características y cada uno de ellos dentro del equipo desarrolla un papel distinto.

A continuación detallaremos cada componente del equipo dentro del desarrollo y cual es la función de cada uno:

1. El experto

La función del experto es la de poner sus conocimientos especializados a disposición del Sistema Experto.

2. El ingeniero del conocimiento

El ingeniero que plantea las preguntas al experto, estructura sus conocimientos y los implementa en la base de conocimientos.

3. El usuario

El usuario aporta sus deseos y sus ideas, determinado especialmente el escenario en el que debe aplicarse el Sistema Experto.

En el desarrollo del Sistema Experto, el ingeniero del conocimiento y el experto trabajan muy unidos. El primer paso consiste en elaborar los problemas que deben ser resueltos por el sistema. Precisamente en la primera fase de un proyecto es de vital importancia determinar correctamente el ámbito estrechamente delimitado de trabajo. Aquí se incluye ya el usuario posterior, o un representante del grupo de usuarios. Para la aceptación ,y e consecuencia para el éxito, es de vital y suma importancia tener en cuenta los deseos y las ideas del usuario.

Una vez delimitado el dominio, nos pondremos a "engrosar" nuestro sistema con los conocimientos del experto. El experto debe comprobar constantemente si su conocimiento ha sido transmitido de la forma más conveniente. El ingeniero del conocimiento es responsable de una implementación correcta, pero no de la exactitud del conocimiento. La responsabilidad de esta exactitud recae en el experto.

El experto deberá tener comprensión para los problemas que depara el procesamiento de datos. Esto facilitará mucho el trabajo. Además, no debe ignorarse nunca al usuario durante el desarrollo, para que al final se disponga de un sistema que le sea de máxima utilidad.

La estricta separación entre usuario, experto e ingeniero del conocimiento no deberá estar siempre presente. Pueden surgir situaciones en las que el experto puede ser también el usuario.

Este es el caso, cuando exista un tema muy complejo cuyas relaciones e interacciones deben ser determinadas una y otra vez con un gran consumo de tiempo. De esta forma el experto puede ahorrarse trabajos repetitivos.

La separación entre experto e ingeniero del conocimiento permanece, por regla general inalterada.

3.5.2 Métodos auxiliares en el desarrollo: La eficiencia en la creación de Sistemas Expertos puede aumentarse en gran medida con la aplicación de *Shells*.

Un Shell es un Sistema Experto que contiene una base de conocimientos vacía. Existen el mecanismo de inferencia, el componente explicativo y a veces también la interface de usuario.

Ya el mecanismo de inferencia depende del problema o grupos de problemas. No existe ningún Shell para todas las aplicaciones, sino que hay que buscar un Shell para cada aplicación.

También es posible que haya que desarrollar adicionalmente partes del mecanismo de inferencia. Según el tamaño de esta parte tendrá que pensar si la aplicación de un Shell determinado sigue siendo apropiada. Si el ingeniero del conocimiento conoce bien este Shell, es decir si, por ejemplo, conoce exactamente cómo son procesadas las reglas, entonces sólo tendrá que concentrarse en la creación de la base de conocimientos. A menudo, el Shell contiene Frames. Estos son marcos previamente preparados, en los que, por ejemplo, sólo se introduce en nombre del objeto, sus cualidades y los correspondientes valores.

Las relaciones entre los objetos se indican mediante señalización de los dos objetos y del tipo de relación que exista entre ellos. El trabajo de implementación debemos procurar, en la medida de lo posible, reducirlo al máximo. Los Frames son componentes explicativos y/o el mecanismo de inferencia están dimensionados de forma distinta en los diferentes Shells.

3.5.3 Construcción de prototipos: En el desarrollo de Sistemas Expertos se nos plantean dos importantes riesgos:

- No existen implementaciones similares que puedan servir de orientación al encargado del desarrollo en casi la totalidad de los casos.

- En muchos puntos, los requisitos necesarios están esbozados con muy poca precisión.

El diseño y la especificación requieren una temprana determinación de la interface del software y de la funcionalidad de los componentes. En el desarrollo de Sistemas Expertos deben alterarse a menudo durante y también después de su implementación, ya que los requisitos se han ido configurando y han obtenido mayor precisión, o porque se ha descubierto que deben iniciarse otras vías de solución. Durante el desarrollo, resulta más apropiado

empezar con implementaciones tipo test para encontrar el camino hacia una solución definitiva y para hacerlas coincidir con las necesidades del usuario.

Un método efectivo es la implementación de un prototipo de Sistema Experto que permita llevar a cabo las funciones más importantes de éste, aunque con un esfuerzo de desarrollo considerablemente inferior al de una implementación convencional. Este proceder se define bajo el nombre de "Rapid Prototyping".

Las máquinas de Inteligencia Artificial especialmente desarrolladas, los lenguajes de programación de Inteligencia Artificial y en determinados casos los Shells, ofrecen una considerable ayuda para el "Rapid Prototyping". Para Sistemas Expertos, el "Rapid Prototyping es el procedimiento más adecuado, pues posibilita una rápida reacción a los deseos en constante cambio tanto por parte de los expertos como parte del usuario

3.6 CAMPOS DE APLICACIÓN

La aplicación de Sistemas Expertos será adecuada allí donde los expertos dispongan de conocimientos complejos en un área muy delimitada, donde no existan algoritmos ya establecidos o donde los existentes no puedan solucionar algunos problemas.

Otro campo de aplicación es allí donde encontremos teorías que resulten prácticamente imposibles de analizar todos los casos teóricamente imaginables mediante algoritmos y en un espacio de tiempo relativamente corto y razonable.

En estas situaciones hace falta el conocimiento que el experto ha adquirido por experiencia, para llegar a una solución en un espacio de tiempo aceptable.

Los dos tipos de problema que hemos descrito se caracterizan además por el hecho de que, aunque es posible la existencia de una o más soluciones, la vía de solución no está previamente fijada. Sin embargo, el experto encuentra a menudo una solución al problema gracias a las informaciones que posee sobre el problema y a su experiencia. Mientras esta solución sea susceptible de repetición y el planteamiento del problema esté claro, existe un razonamiento que puede ser reproducido por un Sistema Experto.

Debido a que la estructuración e implementación del conocimiento del experto requiere una gran cantidad de trabajo, sólo valdrá la pena realizar el esfuerzo de crear un Sistema Experto cuando un conocimiento sea válido durante un largo espacio de tiempo y vaya a ser utilizado por el mayor número de personas.

Resumiendo, los Sistemas Expertos ofrecen ayuda para:

- Evitar fallos en labores rutinarias complejas
- Ampliar de forma más rápida los conocimientos de los especialistas.
- Diagnosticar los fallos con mayor rapidez y conseguir tareas de planificación más completas y consistentes

En este sentido, el Sistema Experto supone una descarga del experto en el trabajo rutinario y, por lo tanto, la reducción de sus problemas. Cuando la labor del experto no está tan sobrecargada, se reducen las decisiones erróneas y se aceleran los procesos en la toma de decisiones.

Los Sistemas Expertos no deben considerarse como soluciones aisladas respecto a otros desarrollos de software. La aplicación del software convencional debe realizarse allí donde tenga sentido hacerlo.

Otro punto que afecta al sistema es el hardware disponible. En una situación en la que un sistema sea necesario en muchos puntos distintos, el Sistema Experto deberá poder funcionar, a ser posible, en el hardware allí disponible.

En la actualidad hay aparatos de Inteligencia Artificial muy avanzados, pero que por cuestiones de costo no pueden ser aplicados en todos los puestos de trabajo de los usuarios.

3.7 ¿CÓMO HACEMOS EL SISTEMA EXPERTO?

Hemos estudiado ya las diversas ventajas que podemos obtener de un sistema experto y es por eso que nos hemos inclinado por esta opción para garantizar la efectividad del diseño de nuestro sistema; diseño que será la base para la implementación de la aplicación que tendrá como usuarios finales a los ingenieros encargados del diseño de tuberías contra-incendio y achique en las diferentes embarcaciones.

Afortunadamente, los sistemas expertos no son algo nuevo sino una rama de la inteligencia artificial sobre la que se ha venido estudiando y trabajando desde hace mucho tiempo y es por eso que hoy en día contamos con herramientas bastante robustas y flexibles que nos permiten su implementación, ahorrándonos de esta manera la preocupación de tener que diseñar el sistema experto desde el principio. Una de estas herramientas es CLIPS.

3.8 LA INTERFAZ DE CLIPS

Los sistemas expertos CLIPS pueden ser ejecutados de tres maneras distintas: Utilizando una interfaz orientada a texto común, o utilizando una ventana como las que ofrece Windows con las ventajas que el entorno visual ofrece, o como sistema experto incrustado en el cual el usuario provee un programa principal y controles de ejecución del sistema experto.

La interfaz de CLIPS genérica es una interface sencilla, orientada a texto con un prompt de comando, que le da alta portabilidad. El uso estándar consiste en crear o editar una base de conocimientos usando un editor de texto corriente, guardar la base de conocimientos como uno o más archivos de texto, salir del editor y ejecutar CLIPS, luego se carga la base de conocimientos en CLIPS. La interface provee comandos para visualizar el estado actual del sistema, rastrear la ejecución, adicionar o remover información y limpiar CLIPS.

Existe una interfaz más sofisticada disponible para Macintosh y Windows y componentes para trabajar desde distintos lenguajes de alto nivel.

3.9 CLIPS: HERRAMIENTA PARA SISTEMAS EXPERTOS

Como ya sabemos un sistema experto tiene la intención de simular el modelo de experiencia humana o conocimiento. CLIPS es llamado herramienta de sistema experto porque es un ambiente completo para desarrollar sistemas expertos, el cual incluye un editor y una herramienta de depuración. La palabra “Shell” es reservada y se refiere a esa porción de CLIPS que desarrolla inferencias o razocinio. El shell de CLIPS proporciona los elementos básicos del sistema experto, estos son:

- La lista de hechos o Facts y las listas de instancias, que son la memoria global para datos.

- Base de conocimientos: Contiene todas las reglas.

- Motor de inferencia: Controla la ejecución general de reglas.

Un programa en CLIPS consiste en reglas, hechos y objetos. El motor de inferencia decide cuales reglas deberían ser ejecutadas y cuando. Un sistema experto basado en reglas escrito en CLIPS es un programa manejador de datos donde los hechos, y objetos si es el caso, son los datos que estimulan la ejecución vía motor de inferencia.

Un ejemplo de cómo CLIPS difiere de los lenguajes procedurales tales como Pascal, Ada, BASIC, FORTRAN y C, es que para lenguajes como estos la ejecución puede proceder sin datos. Esto es, que las instrucciones son suficientes en esos lenguajes para causar la ejecución. Por ejemplo, una instrucción como WRITE (2+2), puede ser inmediatamente ejecutado en Pascal. Esta es una instrucción completa que no necesita datos adicionales para causar su ejecución. Sin embargo, en CLIPS, son requeridos datos para la ejecución de reglas.

Originalmente, CLIPS tuvo capacidades para representar solamente reglas y hechos (facts). Sin embargo, las adiciones hechas en la versión 6.0 permiten que las reglas acepten objetos tan bien como hechos. También los objetos pueden ser usados sin reglas mediante el envío de mensajes y de esta manera el motor de inferencia se hace innecesario si se utilizan solo objetos.

3.10 EMPECEMOS CON CLIPS

A continuación haremos una incursión dentro de la programación con CLIPS presentando algunos ejemplos y comandos básicos. Cabe señalar que lo que se explica a continuación es apenas una ínfima parte de lo que abarca en realidad toda la herramienta como tal. Mas sin

embargo, tocamos estos tópicos por ser los que implementaremos en nuestro diseño del sistema experto.

En primer lugar ejecutamos nuestra aplicación CLIPS de la manera como ejecutamos cualquier aplicación desde nuestro sistema operativo. Es un archivo .exe que podemos adquirir fácilmente por Internet.

En primer lugar visualizamos el “prompt” de CLIPS el cual aparece

CLIPS>

A partir de ahora podemos empezar a entrar comandos directamente en
CLIPS. El modo en el que entramos comandos directos se le conoce como nivel alto. En las versiones recientes para Windows no es necesario digitar comandos sino simplemente escoger de un menú la acción a ejecutar similar a como programamos en lenguajes basados u orientados a objetos. Para salir de CLIPS basta con digitar

(exit)

Inmediatamente la ejecución de CLIPS cesa. Limpiando todo lo que había en memoria. Por eso es recomendable guardar en archivos las reglas y hechos que ocurran en nuestra aplicación. Estos archivos son comunes archivos de texto plano, pero CLIPS los reconoce con extensión CLP.

3.11 HECHOS

Nos interesa también saber como añadir hechos o facts a la memoria. Sencillo, mediante la función assert. Por ejemplo si queremos cargar un hecho llamado Buque, simplemente digitamos

CLIPS> (assert(Buque))

Inmediatamente, una vez dado enter, se carga en memoria dicho hecho. Nótese que es una sintaxis muy sencilla, la mayoría de las veces se utilizan los paréntesis para dar comandos. Otra cosa que cabe señalar es que CLIPS es sensible a mayúsculas y minúsculas, por lo tanto no se interpretará igual Buque que buque.

Una vez introducido el hecho, CLIPS nos da una idea del lugar donde lo guarda. Si nuestra primera inserción fue (assert(Buque)) entonces obtendremos la respuesta

<Fact-0>

que indica que CLIPS ha guardado el hecho Buque en la lista de hechos y le ha dado el identificador 0. CLIPS automáticamente nombra los hechos usando un número que incrementa en uno a medida que vamos insertando más y más hechos. Para ver que hechos tenemos cargados en memoria damos el comando facts, esto es:

CLIPS>(facts)

Damos enter y obtenemos una lista con los hechos que tenemos, que para nuestro caso será solamente buque. Lo que nos muestra CLIPS es esto:

f-0 (Buque)

For a total of 1 fact.

CLIPS>

El término f0 es pues el identificador de hecho que le asigna CLIPS a dicho hecho. A cada hecho insertado en la lista de hechos se le asigna un único identificador de hecho empezando con la letra "f" y seguido por un entero llamado índice de hecho. Al inicializar CLIPS, y después de ciertos comandos como clear y reset que serán descritos posteriormente, la lista de índices retornará desde cero, y luego se seguirá incrementando con cada hecho insertado. El comando (reset) insertará un hecho inicial automáticamente que es el (initial-fact), como f-0. Este hecho es a menudo usado por conveniencia para activar reglas. Entonces, si ejecutamos el comando (reset), e insertamos nuestro hecho (assert(Buque)) y le damos (facts) obtenemos que en la localización f-0 se encuentra (initial-fact) y en la f-1 tenemos (Buque).

Ahora bien, no podemos insertar dos hechos exactamente iguales, es decir, si intentamos insertar nuevamente el hecho (Buque), CLIPS nos responderá FALSE, lo que significa que ya el sistema conoce este hecho y no será necesario "enseñárselo" nuevamente. Sin embargo, para mayor flexibilidad existe un comando que nos permite duplicar hechos que es (set-fact-duplication), que permite duplicar la entrada de hechos.

Los hechos puede ser removidos mediante el comando retract + el índice asignado por CLIPS al hecho que queremos remover, por ejemplo:

(retract 1)

Cuando un hecho es removido, los otros hechos no cambian sus índices, y entonces los identificadores de los que removimos permanecen vacío. Aún cuando insertemos un nuevo hecho, este se ubicará en el índice posterior al último sin rellenar los que fueron desocupados. Por ejemplo, si damos (assert(buque)), después de un (reset), este se ubicará en la posición f-1 (recordemos que reset inserta en f-0 el hecho initial -fact). Demos ahora:

CLIPS>(retract 1)

Si ejecutamos (facts) notaremos que solamente se encuentra el hecho inicial. Ahora, si insertamos otro hecho, (assert(embarcación)), este se ubicará en la posición f-2 y no en la f-1.

Podemos insertar varios hechos con un solo comando y para esto hay dos maneras, con assert y con deffacts.

Para utilizar assert simplemente separamos con paréntesis cada hecho que vayamos a insertar, por ejemplo:

CLIPS>(assert (buque) (tipo) (arqueo-bruto))

Insertará tres hechos diferentes, cada uno con su identificador diferente, f-0 para buque, f-1 para tipo, f-2 para arqueo bruto. Nótese que se ha concatenado arqueo bruto mediante el guión "-", esto significa que es un hecho de un solo campo. Si se hubiera separado y puesto (arqueo bruto), CLIPS lo interpreta como un hecho de dos campos, lo cual es importante a la hora de la evaluación de las reglas.

Otra manera para insertar reglas es mediante deffacts, que se implementa de manera similar a assert en cuanto a la sintaxis, pero que no se ejecuta una vez dado enter. Además hay que especificar un nombre de deffacts seguido este comando. Para cargar los hechos en la memoria tendríamos que digitar reset, y entonces se cargan los hechos definidos, siendo el identificador f-0 para el hecho (ini tial-fact) y el resto en el orden en que quedan definidos por deffacts.

Por ejemplo:

CLIPS> (deffacts nombre ; se pueden hacer comentarios antecedidos

de ";"

(accesorio)

(valvula)

(material))

CLIPS>(reset)

Si damos (facts) obtenemos:

CLIPS>(facts)

f-0 (initial-fact)

f-1 (accesorio)

f-2 (valvula)

f-3 (material)

for a total of 4 facts.

CLIPS>

Nota: Hay caracteres que no son válidos y esta es la razón por la que en lugar de insertar (válvula), se insertó (valvula). La tilde es uno de estos casos inválidos.

Para limpiar la memoria de CLIPS, basta con digitar el comando (clear), pero hay que tener cuidado porque éste no solamente borra los hechos, sino también las reglas que hayamos especificado.

3.12 REGLAS

Para lograr una herramienta útil, un sistema experto debe tener tanto
hechos como reglas. Hasta ahora hemos visto, como podemos insertar y remover hechos, ahora veremos como trabajan las reglas. Una regla es similar a una instrucción IF THEN en un lenguaje procedural como Ada, C o Pascal. Una regla IF THEN puede ser expresada como

SI (IF) ciertas condiciones son verdaderas
ENTONCES (THEN) ejecute las siguientes acciones

La sentencia anterior es reconocida en nuestro medio como pseudocódigo, que literalmente significa código falso. Como sabemos, el pseudocódigo no puede ser ejecutado directamente desde el computador, pero nos es muy útil como guía para escribir código ejecutable. Una traducción de reglas de idioma natural a CLIPS no es muy difícil si se tiene presente la analogía IF THEN en mente. A medida que se adquiere experiencia en CLIPS notamos que escribir reglas es muy fácil.

Es posible escribir reglas directamente en CLIPS o cargarlas de un archivo de reglas creadas en un editor de texto. Veamos ahora un ejemplo:

El pseudocódigo para una regla en el diseño de tuberías para sistemas contraincendio y achique en buques podría ser:

SI el arqueo bruto del buque es de 2000 toneladas
ENTONCES utilice al menos dos bombas.

Ahora se expresa lo anterior en términos de CLIPS con un hecho definido y una regla. Primero se crea un hecho:

CLIPS> (assert (arqueo-bruto-es 2000)

Hemos insertado un hecho o fact de dos campos que son arqueo-bruto-es y 2000. Obviamente se ha pulsado la tecla enter una vez digitado el comando anterior. Definamos ahora la regla que se disparará cuando se dé este hecho, dicha regla la llamaremos numero-bombas. La sintaxis es como sigue:

CLIPS>(defrule numero-bombas
(arqueo-bruto-es 2000)
=>
(assert(numero-es 2))
CLIPS>

Si la sintaxis de la regla fue digitada correctamente aparecerá nuevamente el prompt de CLIPS, de otro modo, saldrá un mensaje de error. Estas regla se pudo escribir también con un encabezado, el cual se le conoce como encabezado de regla y puede estar encerrado en comillas dobles y se escribe después del nombre de la regla y antes del primer patrón. Otra cosa que cabe anotar es el uso del enter antes de digitar el comando completo. CLIPS interpreta los enter y los tabs de manera similar que los espacios comunes, así que para mayor organización en la presentación del código CLIPS podemos dar enter, una vez terminada una línea de comandos. Cabe anotar también que CLIPS solo acepta un nombre de regla distinto cada vez, si se le inserta una regla con un nombre que ya estaba, esta nueva reemplazará la que ya estaba. La regla completa va encerrada entre paréntesis. Los patrones son las instrucciones dadas antes de la sentencia"=>" y van encerradas cada una entre paréntesis también. Las acciones vienen después de la sentencia "=>" y también van encerradas cada una entre paréntesis. Una regla puede tener múltiples patrones y acciones y no necesariamente deben ser de igual número.

La sentencia "=>" es como el entonces de toda la instrucción como hemos podido apreciar.

Si todos los patrones de una regla coinciden con hechos, la regla es activada y colocada en la agenda. La agenda es una colección de activaciones, las cuales son aquellas reglas que encuentran coincidencia de sus patrones con los hechos en la memoria. Pueden haber en la agenda cero o más activaciones.

Para nuestro ejemplo ocurre que la regla es activada, puesto que su patrón (arqueo-bruto-es 2000) coincide con un hecho residente en memoria, por lo tanto dicha regla es almacenada en agenda, para comprobar esto, se digita el comando (agenda) y obtenemos una lista de las reglas que se han activado las cuales a su vez tienen un identificador que CLIPS les asigna

automáticamente y al dispararlas las va mostrando de acuerdo a la estrategia que estemos utilizando para nuestro motor de inferencia, lo cual se explicará más adelante. En general la parte de la regla antes de la flecha "=>" por así decirlo, se conoce como LHS (Left Hand side), o lado izquierdo de la regla y a la otra parte RHS o lado derecho. Para que se ejecute la acción de la regla activada basta con entrar el comando (run).

LAS ESTRATEGIAS

La palabra estrategia era originalmente un término militar para la planeación y operaciones bélicas. Hoy, el término estrategia es comúnmente usado en los negocios (que se puede decir que es un tipo de guerra) para referirnos a los planes de una organización en el logro de metas.

En sistemas expertos, el se utiliza el término estrategia en la resolución de conflictos de activación. Se podría decir que podría diseñarse un sistema experto de tal manera que solo puede ser activada una regla a la vez, entonces no habría necesidad de una resolución de conflictos de activación. Es algo muy cierto, el problema radica en que para hacer esto no es necesario un sistema experto pues un programa secuencial haría exactamente lo mismo, que es lo que hace un C o un Pascal o Basic.

CLIPS ofrece siete diferentes modos en la resolución de conflictos, estos son: depth, breadth, LEX, MEA, complexity, simplicity, y random. Es difícil determinar cual es mejor que la otra sin considerar la aplicación específica. Aún así, puede ser difícil juzgar cual es la mejor. La estrategia por defecto en CLIPS es depth, pero veamos de que trata cada una, para tener una idea de su base:

- Depth : Las reglas más recientemente activadas son colocadas sobre todas las reglas con la misma salida. La salida es el identificador que reciben. Por ejemplo, dado un hecho A que activa las reglas 1 y 2 y un hecho B que activa las reglas 3 y 4, entonces si el hecho A es insertado antes que el B, las reglas 3 y 4 estarán arriba de las reglas 1 y 2 en la agenda. Sin embargo, la posición de la regla 1 respecto a la 2 y la regla 3 respecto a la 4 serán arbitrarias.

- Breadth: Las reglas más recientes son colocadas debajo de las otras reglas de la misma salida. Por ejemplo, dado el hecho A, que activa las reglas 1 y 2 y el hecho B que activa las reglas 3 y 4, entonces si el hecho A es insertado antes que el B, la regla 1 y 2 estarán sobre la 3 y 4 en la agenda. Sin embargo, la posición de la regla 1 respecto a la 2 y la 3 respecto a la 4 serán arbitrarias.

- Simplicity: Entre reglas con la misma salida, las más recientemente activadas son colocadas sobre todas las activaciones de reglas con igual o más alta especifidad. La especifidad de una regla es determinada por el número de comparaciones que puedan haber en el LHS de la regla. Cada comparación con una constante o previamente una variable de límite aumenta la especifidad en uno. Cada llamado a una función en el LHS de una regla como parte de :, =, o elemento de comprobación de condición aumenta la especifidad en uno. Las expresiones booleanas utilizadas en la regla como and, or y not no agregan especifidad, pero sus argumentos sí. Llamados a funciones hechos dentro de un llamado a una función no aumentan la especifidad de la regla. Por ejemplo:

(defrule ejemplo

(item ?x ?y ?x)

(test (and (numberp ?x) (> ?x(+10 ?y)) (< ?x 100)))

=>)

tiene una especifidad de 5. La comparación con la constante item, la comparación de ?x con su previa asociación, y los llamados a las funciones numerop, < y > , aumentan en uno cada una a la especifidad para un total de 5. Los llamados a las funciones and y no aumentan especifidad a la regla.

- Complexity: Entre reglas de la misma salida, las más recientemente activadas son colocadas sobre todas las activaciones de reglas con igual o más baja especifidad.

3.12.1 Estrategia LEX: Entre reglas de igual salida, las más recientemente activadas son colocadas utilizando la estrategia OPS5 del mismo nombre.

Primero la regencia o menos tiempo de haber sido insertados los hechos que coincidieron con los patrones del LHS y que activaron la regla, son usados para determinar donde colocar la activación. Cada hecho, es marcado internamente con una “etiqueta de tiempo” para indicar su regencia relativa con respecto a cualquier otro hecho en la memoria del sistema. Los patrones asociados con cada activación de regla son almacenados en orden descendente para determinar su lugar. Una activación con más patrones recientes es colocada antes que una activación con menos patrones recientes. Para determinar el orden de lugar de dos activaciones, se comparan las etiquetas de tiempo de las dos activaciones una por una empezando con las

etiquetas de tiempo más largas. La comparación debería continuar hasta que una etiqueta de tiempo de una activación es más grande que la etiqueta de tiempo de la otra activación.

La activación con el tiempo de etiqueta más grande es colocada antes que otra activación en la agenda.

Si una activación tiene más patrones que otra y las etiquetas de tiempo son idénticas, entonces la activación con el mayor número de etiquetas de tiempo se coloca antes que la otra activación en la agenda. Si dos activaciones tienen la regencia exacta, la activación con la especifidad más alta es colocada sobre la otra.

Como ejemplo veamos las siguientes seis activaciones que han sido listadas en su orden LEX (donde la coma al final de la activación indica la presencia de un elemento condicional *not*). Cabe anotar que la etiqueta de tiempo del hecho no necesariamente es la misma que su índice, (esto es porque en CLIPS también pueden declararse instancias como en lenguajes orientados a objetos y a dichas instancias también se le asignan etiquetas de tiempo), pero si un índice de hecho es más grande que otro, entonces su etiqueta también es más grande. Para este ejemplo, asuma que las etiquetas de tiempo e índices son lo mismo.

Rule-6 : f-1, f4

Rule-5 : f-1, f-2, f-3,

Rule-1 : f-1, f-2, f-3

Rule-2: f-3, f-1

Rule-4 :f-1, f-2,

Rule-3 : f-2,f-1

La anterior es una lista arbitraria de activación de reglas que quedaría ordenada por medio de la estrategia LEX de la siguiente manera:

Rule-6 : f-4, f-1

Rule-5 : f-3, f-2, f-1,

Rule-1 : f-3, f-2, f-1

Rule-2 : f-3, f-1

Rule-4 : f-2, f-1,

Rule-3 : f-2, f-1

3.12.2 Estrategia MEA: Entre reglas de la misma salida, las más recientemente activadas son colocadas usando la estrategia OPS5 del mismo nombre. Primero la etiqueta de tiempo de la entidad patrón asociada con el primer patrón es usada para colocar donde colocar la activación. Una activación cuya etiqueta de tiempo del primer patrón es más grande que la primera etiqueta de tiempo de otro primer patrón es colocada antes en la lista de activación en la agenda. Si ambas activaciones tienen la misma etiqueta de tempo asociada al primer patrón entonces se utiliza la estrategia LEX para determinar la colocación de las activaciones.

Como ejemplo, la siguiente lista muestra seis activaciones que han sido mostradas en su orden MEA (donde la coma al final de la activación indica la presencia de un patrón negado).

Rule-2 : f-3, f-1

Rule-3 : f-2, f-1

Rule-6 : f-1, f-4

Rule-5 : f-1, f-2, f-3,

Rule-1 : f-1, f-2, f-3

Rule-4 : f-1, f-2,

3.12.3 Estrategia Random: A cada activación le es asignado un número aleatorio, el cual es usado para determinar localización entre activaciones de igual salida. Este número aleatorio es preservado cuando la estrategia es cambiada de tal manera que el mismo ordenamiento es reproducido cuando la estrategia random es seleccionada otra vez. (entre activacione estaban en la agenda cuando la estrategia fue originalmente cambiada).

4. DIAGRAMACIÓN DEL PROCEDIMIENTO DE DISEÑO Y COMPROBACIÓN DE SISTEMAS DE CONTRAINCENDIO CON AGUA DE MAR Y ACHIQUE

Los modelos de sistemas desempeñan un papel importante en el desarrollo de sistemas. Un modelo es la representación de la realidad. Como una imagen vale más que mil palabras, los modelos es su mayoría son representaciones gráficas de la realidad. En este capitulo se mostrará el diagrama de flujo de datos que se desarrollo para este proyecto y toda la teoría referente a esta modelización de procesos.

4.1 MODELIZACIÓN DE UN SISTEMA

Los modelos de sistemas desempeñan un papel importante en el desarrollo de sistemas. Una forma de estructurar dichos problemas es elaborar modelos. Como ya se anotó, un modelo es una representación de la realidad. Pueden establecerse modelos para los sistemas existente, con el fin de obtener un mejor conocimiento de dichos sistemas, o para lo sistemas propuestos como medio para definir sus requisitos y diseños.

Es importante distinguir entre modelos de implantación y modelos esenciales.

Los modelos de implantación muestran no solo lo que es o hace un sistema, sino también cómo es su implantación física. Es también conocido como modelo físico.

Los modelos esenciales son modelos, independientes de la implantación, que describen la esencia del sistema (lo que hace o lo que debe hacer) sin tener en cuenta el modo de implantación física de dicho sistema. Los modelos esenciales son también conocidos como modelos lógicos o modelos conceptuales.

En este trabajo se desarrolla un modelo esencial del procedimiento que sigue el departamento de dirección de proyectos para el diseño de los sistemas de achique y contraincendio con agua de mar. ¿Porque este modelo y no un modelo de implantación ? Aquí se exponen las razones: Primera razón; los modelos esenciales fomentan la creatividad ya que eliminan los sesgos que se producen al pensar en la implantación del sistema existente. Segunda razón; estos modelos reducen los riesgos de omitir necesidades funcionales por prestar demasiada atención a detalles técnicos, al separar lo que debe hacer el sistema de cómo lo hará podemos analizar mejor las necesidades para conseguir un sistema complejo, preciso y consistente. Última razón; estos modelos nos permite comunicarnos con los usuarios finales en el lenguaje menos técnico.

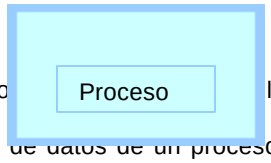
Existen diferentes herramientas para la modelización de procesos, este capitulo desarrolla uno de ellos; el diagrama de flujo de datos.

4.2 DIAGRAMA DE FLUJO DE DATOS

4.2.1 Definición de los diagramas de flujo de datos: Un diagrama de flujo de datos es una herramienta de modelización de procesos que representa el flujo de datos a través de un sistema y los trabajos o procesos llevados a cabo por dicho sistema.

4.2.2 Convenciones y directrices de los diagramas de flujo de datos: Existen diferentes propuestas para las notaciones simbólicas para estos diagramas. Usaremos la notación propuesta por Chris Ganes y Trish Sarson.

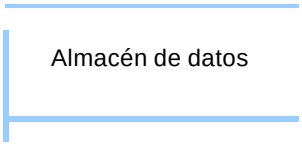
- Proceso:Un proceso es un conjunto de tareas o acciones realizadas a partir de un flujo de datos de entrada para producir flujos de datos de salida. Todos los procesos deben estar asociados a flujos de datos con al menos una entrada y una salida.



- Flujo de datos: El flujo la introducción de datos en un proceso o la obtención de datos de un proceso. Puede también representar la actualización de datos en un archivo, una base de datos u otro medio de almacenamiento de datos. El flujo de datos puede verse como una vía por la cual transitan paquetes de datos de composición conocida.

Flujo de datos

- Almacén de datos: La mayoría de los sistemas de información capturan datos para su uso posterior. Estos datos necesitan ser almacenados en“almacenes de datos”.



4.3 DIAGRAMA DE FLUJO DE DATOS PARA EL DISEÑO Y COMPROBACIÓN DE SISTEMAS DE CONTRAINCENDIO CON AGUA DE MAR Y ACHIQUE

El proceso de diseño y comprobación de sistemas de contraincendio con agua de mar y achique que se describió en el capitulo anterior puede modelarse con el siguiente diagrama de flujo de datos.

A lo largo del proceso de investigación y desarrollo de este proyecto se concluyó que tanto el proceso de diseño de sistemas de contraincendio con agua de mar como el proceso de diseño de sistemas de achique siguen los mismos patrones de desarrollo, es por esta razón que el diagrama de flujo de datos presentado anteriormente describe el diseño de estos dos sistemas.

Además, analizando otros sistemas de tuberías que se diseñan en el departamento de Dirección de Proyectos de Cotecmar, llegamos a la conclusión de que el diagrama desarrollado en este trabajo puede ser aplicado para el diseño y comprobación de cualquier sistema de tuberías y puede ser utilizado por los diseñadores e implementadores de futuros proyectos en esta área

En la figura 1 es el Diagrama de Contexto del diseño y comprobación de sistemas de contraincendio con agua de mar y achique. Este proceso se divide en seis procesos principales (Figura2). El alcance de este trabajo abarca los tres primeros procesos que se explican con mas detalle en las figuras 3, 4 y 5

DIAGRAMA DE CONTEXTO

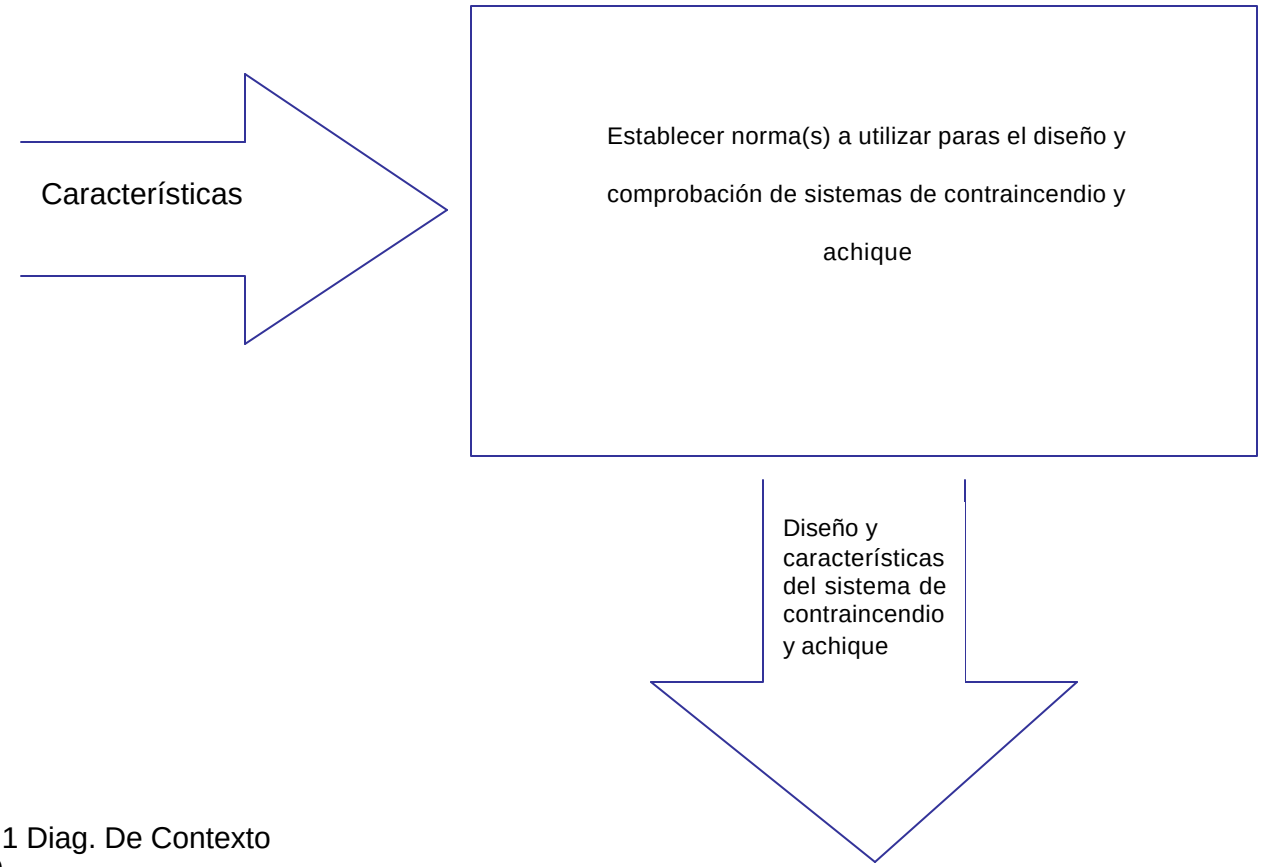
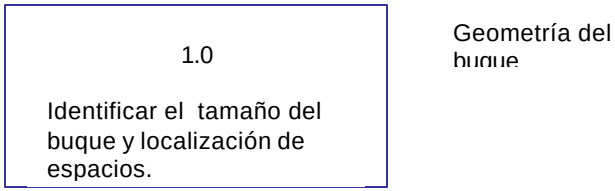
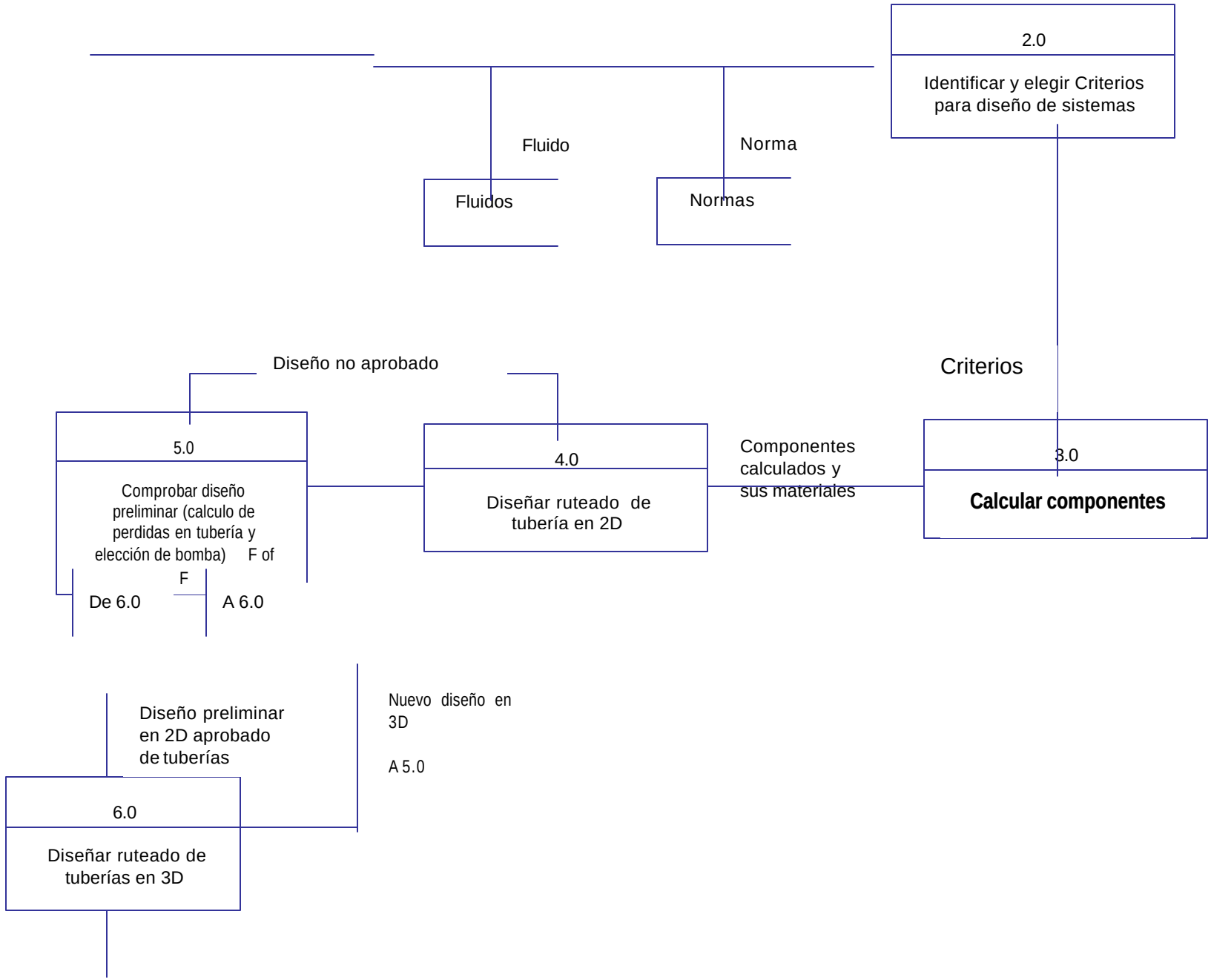


Figura 1 Diag. De Contexto

Nivel 0



Geometría del buque


2.0
Identificar y elegir Criterios para diseño de sistemas
Fluido
Norma
Fluidos
Normas
Diseño no aprobado
Criterios
5.0
Comprobar diseño preliminar (calculo de perdidas en tubería y elección de bomba) F of F
De 6.0
A 6.0
4.0
Diseñar ruteado de tubería en 2D
Componentes calculados y sus materiales
3.0
Calcular componentes
Diseño preliminar en 2D aprobado de tuberías
Nuevo diseño en 3D
A 5.0
6.0
Diseñar ruteado de tuberías en 3D

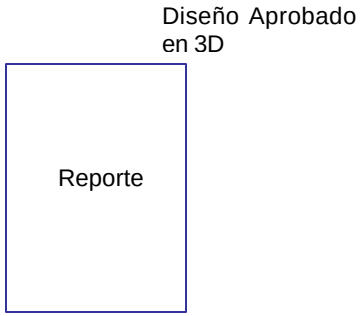
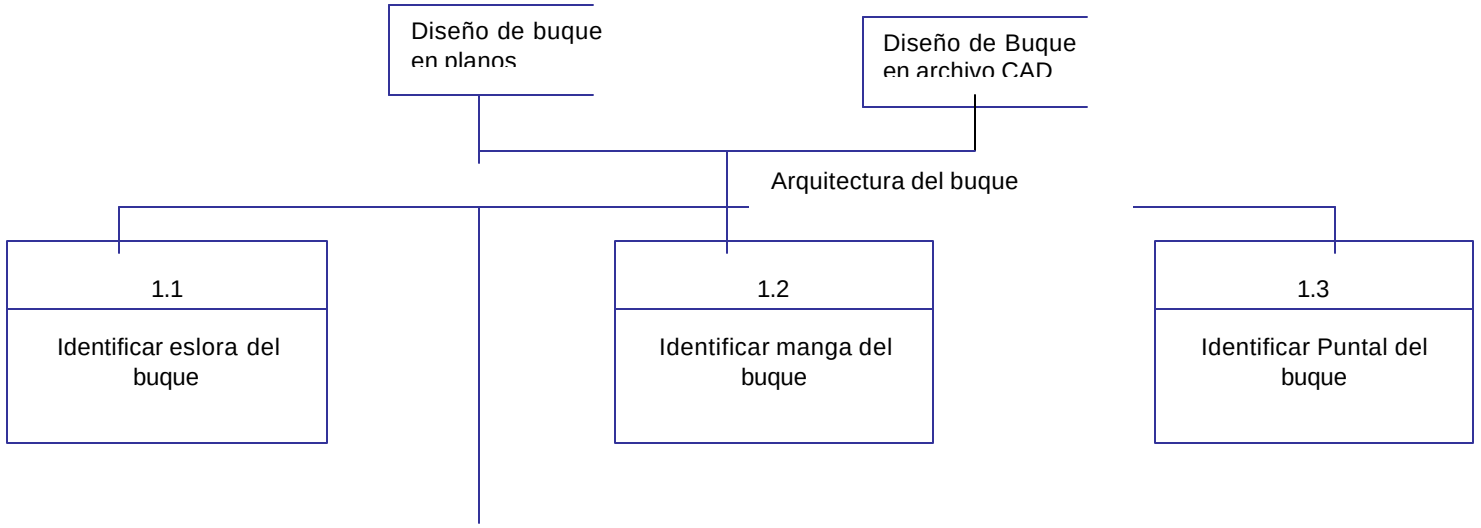


Figura 2 Nivel 0

Nivel 1
Proceso 1.0



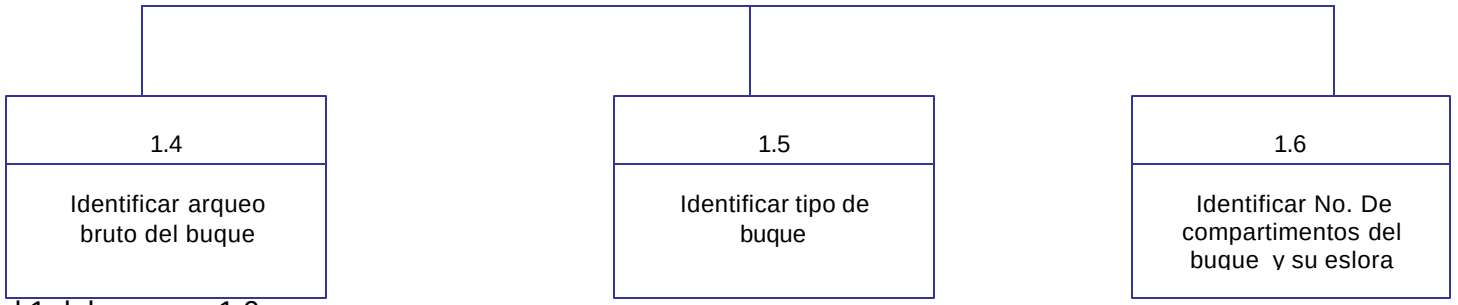


Figura 3 Nivel 1 del proceso 1.0

Nivel 1
Proceso 2.0

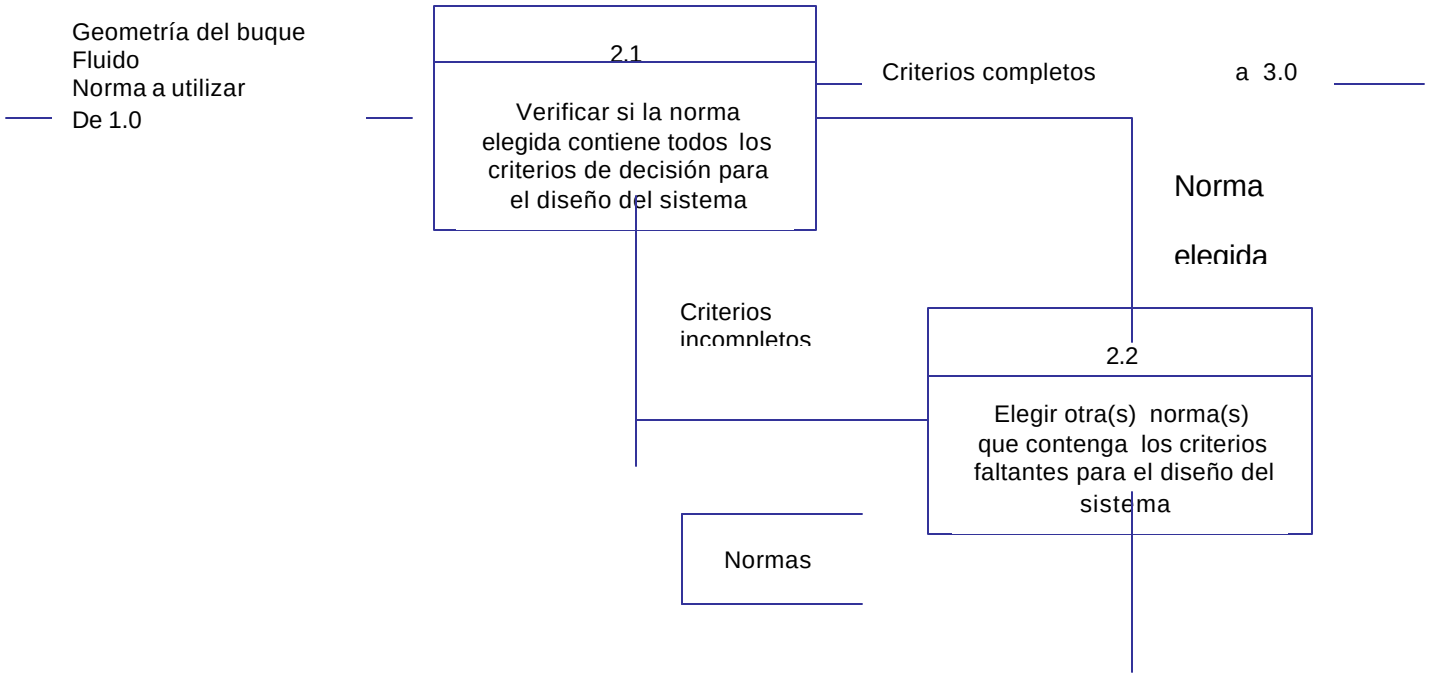


Figura 4 Nivel 1 del proceso 2.0

Nivel 1
Proceso 3.0

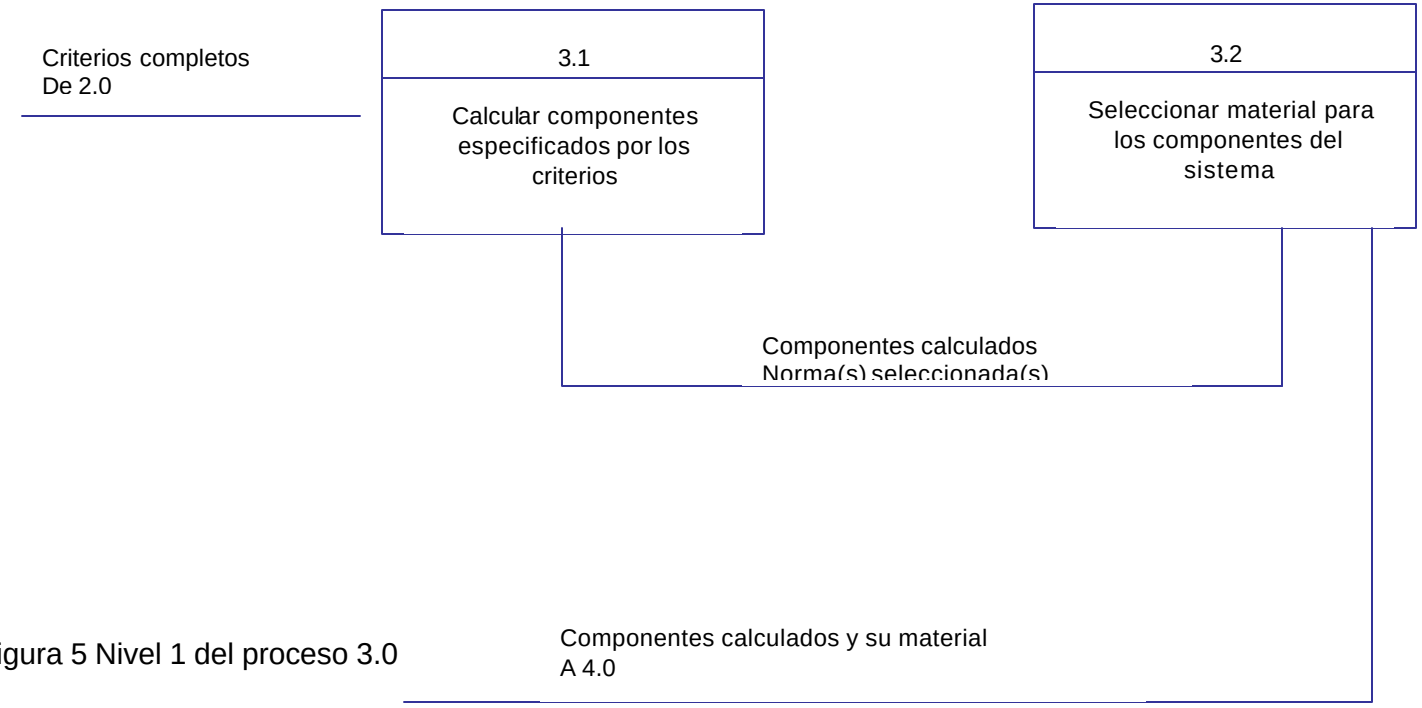


Figura 5 Nivel 1 del proceso 3.0

5. DISEÑO DE LA BASE DE DATOS

Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información. La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. En suma, los sistemas de base de datos deben proporcionar la fiabilidad de la información almacenada, a pesar de las caídas del sistema o los intentos de acceso sin autorización. Si los datos va a ser compartidos entre diversos usuarios, el sistema debe evitar posibles resultados anómalos.

5.1 MODELO ENTIDAD – RELACIÓN.

El modelo de datos entidad – relación está basado en una percepción del mundo real que consta de un conjunto de objetos básicos llamados entidades y de relaciones entre estos objetos. Se desarrollo para facilitar el diseño de bases de datos permitiendo la especificación de un esquema de la empresa que representa la estructura lógica completa de una base de datos. El modelo de datos E-R es uno de los diferentes modelo de datos semánticos ; el aspecto semántico del modelo yace en el intento de representar el significado de los datos. El modelo E-R es extremadamente útil para hacer corresponder los significados e interacciones de los desarrollos del mundo real con un esquema conceptual.

5.1.1 Entidades: Una Entidad es un objeto que existe y puede ser distinguido de otro objeto. Una entidad puede ser concreta (un libro, un automóvil etc.) o abstracta (fecha, edad, etc.). Un conjunto de entidades es un grupo de entidades del mismo tipo. Una entidad puede pertenecer

a mas de un conjunto de entidades a la vez. Por ejemplo, la entidad persona puede ser parte de los conjuntos de entidades alumnos, empleados, clientes etc.

Una entidad se distingue de otra porque posee ciertas características que la hacen única. A estas características se les conoce como atributo. El rango de valores validos para un atributo determinado será conocido como dominio del atributo.

Ejemplo:

Entidad Empleado X

Atributo :

-RFC

-Nombre

-Salario (2000..10,000)

-Edad (18..60)

Una entidad se describe por un conjunto de parejas en el siguiente formato (atributo, valor del dato); debiendo especificarse una pareja por cada atributo de la entidad.

Ejemplo:

{(Nombre,Juan), (Edad,15), (Carrera,LI) }

Teniendo en claro el concepto de entidades, hemos desarrollamos las siguientes entidades para este trabajo de investigación:

Entidad TIPO_BUQUE

Atributos:

- Codigo

  Tipo de dato: Código de dos caracteres.

  Descripción: Código que especifica el tipo de buque

- Buque

  Tipo de dato: 15 caracteres.

  Descripción: Especifica tipo de buque (Ej. Carga).

Ejemplo:

Tabla 3 Entidad Tipo_Buque

Codigo	Buque
01	Pasajeros
02	Carga

Entidad ARQUEO_BRUTO

Atributos:

- Codigo

  Tipo de dato: Código de dos caracteres.

  Descripción: Código que especifica el arqueo bruto de un buque.

- Arqueo

  Tipo de dato: Código de ocho caracteres.

  Descripción: Especifica un rango de arqueo bruto donde los cuatro primeros caracteres son el límite inferior y los cuatro últimos es límite superior. (Ej. 00001000 Este código indica un arqueo bruto mínimo de 0 toneladas y un arqueo bruto máximo de 1000 toneladas).

Ejemplo:

Tabla 4 Entidad Arqueo_Bruto

Codigo	Arqueo
01	00001000
02	10004000

Entidad NORMAS

Atributos:

- Codigo

  Tipo de dato: Código de dos caracteres.

  Descripción: Código que especifica la reglamentación

utilizada.

- Norma

  Tipo de dato: Cadena de cuarenta caracteres.

  Descripción: Nombre de la casa clasificadora o de la reglamentación utilizada.

Ejemplo:

Tabla 5 Entidad Normas

Código	Norma
01	Solas
02	Bureau Veritas

Entidad FLUIDO

Atributos:

- Codigo

  Tipo de dato: Código de dos caracteres.

  Descripción: Código que especifica el nombre del fluido.

- Fluido

  Tipo de dato: Cadena de 40 caracteres.

  Descripción: Nombre del fluido.

Ejemplo:

Tabla 6 Entidad Fluido

Codigo	Fluido
01	Agua de Mar
02	Agua de Río

Entidad SISTEMA

Atributos:

- Codigo

  Tipo de dato: Código de dos caracteres.

  Descripción: Código que especifica el nombre del sistema a diseñar.

- Sistema

  Tipo de dato: Cadena de 40 caracteres.

  Descripción: Nombre del sistema a diseñar.

Ejemplo:

Tabla 7 Entidad Sistema

Codigo	Sistema
01	Contraincendio-seco
02	Contraincendio-humedo
03	Achique
04	Contraincendio-CO2

Entidad BOMBAS_FIJAS

Atributos:

- Codigo

  Tipo de dato: Código de diez caracteres.

  Descripción: Los dos primeros son el código de la entidad TIPO_BUQUE, el tercer y cuarto carácter son el código de la entidad ARQUEO_BRUTO, el quinto y sexto carácter son el código de la entidad NORMA, el séptimo y octavo carácter son el código de la tabla FLUIDO, y los dos últimos caracteres son el código de la entidad SISTEMA .

- No_B_Fijas

  Tipo de dato: Código de dos caracteres.

  Descripción: No. mínimo de bombas fijas que se especifica para cada tipo de buque.

Ejemplo:

Tabla 8 Entidad Bombas_Fijas

Codigo	No_B_Fijas
0102010101	03

Entidad MANGUERAS

Atributos:

- Codigo

  Tipo de dato: Código de diez caracteres.

  Descripción: Los dos primeros son el código de la entidad TIPO_BUQUE, el tercer y cuarto carácter son el código de la Entidad ARQUEO_BRUTO, el quinto y sexto carácter son el

código de la entidad NORMA, el séptimo y octavo carácter son el código de la tabla FLUIDO, y los dos últimos caracteres son el código de la entidad SISTEMA .

- No_Mangueras

  Tipo de dato: Código de dos caracteres.

  Descripción: No. mínimo de mangueras se especifica para cada tipo de buque.

Ejemplo:

Tabla 9 Entidad Mangueras

Codigo	No_Mangueras
0102010101	

Entidad ACCESORIOS

Atributos:

- Codigo

  Tipo de dato: Código de cuatro caracteres.

  Descripción: Código que especifica cada uno de los ítem (bomba, válvula, tubo, etc.) y su tipo o estilo. Los dos primeros caracteres especifican el ítem y los dos últimos su tipo o estilo.

- Accesorio

  Tipo de dato: Cadena de cuarenta caracteres.

  Descripción: Nombre del ítem y su tipo o estilo.

Ejemplo:

Tabla 10 Entidad Accesorios

Codigo	Accesorios
0101	pipe-seamless
0102	pipe-welded
0103	pipe- filament wound
0104	pipe- centrifugally cast
0201	Takendown joints-flanges brazed
0202	Takendown joints-unions brazed
0408	fittings-socket weld or threaded
0409	fittings-sleeve coupling
0501	valves-butterfly water or lug
0502	valves-butterfly grooved end

Entidad MATERIALES

Atributos:

- Codigo

  Tipo de dato: Código de tres caracteres.

  Descripción: Código que especifica el nombre del material y la temperatura máxima que este material tolera.

- Material

  Tipo de dato: Cadena de 20 caracteres

  Descripción: Nombre del material

- Temp._Max

Tipo de dato: Numérico

Descripción: Temperatura máxima que tolera el material especificado.

Ejemplo:

Tabla 11 Entidad Materiales

Codigo	Material	Temp._Max
001	Carbon_Steel	120
002	Cooper_Niquel	150

Entidad ESPECIFICACION_MATERIAL

Atributos:

- Codigo

  Tipo de dato: Código de dos caracteres.

- Especif_Material

  Tipo de dato: Cadena de 40 caracteres.

  Descripción: Especificación de material.

Ejemplo:

Tabla 12 Entidad ESPECIFICACION_MATERIAL

Codigo	Especif_Material
01	B 584 Alloy 905
02	B 164 Class A
03	A 276 Type 304

Entidad ESPECIFICACION_DISEÑO

Atributos:

- Codigo

  Tipo de dato: Código de dos caracteres.

- Especif_Diseño

  Tipo de dato: Cadena de 40 caracteres.

  Descripción: Especificación de diseño de material.

Entidad ESPECIFICACIÓN DE DISEÑO

Atributos:

- Cod_Norma

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad NORMA

- Cod_Fluido

  Tipo de dato: Código de dos caracte res.

  Descripción: Código de la entidad FLUIDO.

- Cod_Sistema

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad SISTEMA

- Cod_Accesorio

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad ACCESORIO.

- Cod_Materiales

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad MATERIALES

- Cod_Especif_Material

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad ESPECIFICACIÓN_

  MATERIAL

- Cod_Especif_Diseño

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad ESPECIFICACIÓN_ DISEÑO.

Ejemplo:

Tabla 13 Entidad ESPECIFICACIÓN DE DISEÑO

Cod_norma	Cod_fluido	Cod_Sistema	Cod_Accesorios	Cod_Materiales	Cod_Especif_Material	Cod_Especif_diseño
03	01	01	01	001	01	01
03	02	01	01	005	02	02

Entidad TRIM

Atributos:

- Codigo

  Tipo de dato: Código de cuatro caracteres.

Descripción: Código que especifica en sus dos primero caracteres el número del grupo de trim (Ej: Grupo 1) y los dos último caracteres especifican el nombre del trim.

- Trim

  Tipo de dato: Cadena de 40 caracteres.

  Descripción: Nombre de los interiores de la válvula.

- Cod_Material

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad MATERIALES

- Cod_Especif_Material

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad ESPECIFICACIÓN_ MATERIAL

- Cod_Especif_Diseño

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la tabla ESPECIFICACIÓN_DISEÑO

Ejemplo:

Tabla 14 Entidad TRIM

Codigo	Trim	Cod_Material	Cod_Especif_Material	Cod_Especif_Diseño
0101	Steam	014	05	03

Entidad VÁLVULA_TRIM

Atributos:

- Cod_Norma

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad NORMA

- Cod_Fluido

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad FLUIDO.

- Cod_Sistema

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad SISTEMA

- Cod_Accesorio

  Tipo de dato: Código de dos caracteres.

  Descripción: Código de la entidad ACCESORIO

- Cod_Trim

  Tipo de dato: Código de cuatro caracteres.

  Descripción: Código de la entidad TRIM. Siempre los dos últimos caracteres serán cero para indicar que este registro toma todos los registro de la entidad TRIM donde los dos primero caracteres sean iguales a los dos primeros caracteres de Cod_Trim.

Ejemplo:

Tabla 15 Entidad VÁLVULA_TRIM

Cod_Norma	Cod_Fluido	Cod_Sistema	Cod_Accesorio	Cod_trim
01	03	02	04	0100

5.1.2 Relaciones: Una relación es una asociación natural entre una o más entidades. Estas asociaciones pueden determinarse de forma relativamente rápida una vez identificadas las entidades.

Así como las entidades se designan con nombres, se debe nombrar las relaciones con verbos o frases verbales.

5.2 BASE DE DATOS

Un sistema de gestión de bases de datos consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos.
La colección de datos, normalmente denominada Base de Datos, contiene información acerca de una empresa en particular.

Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información. La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información.

5.2.1 Tablas: Una base de datos consiste en un conjunto de tablas, a cada una de las cuales se le asigna un nombre exclusivo. Cada una de estas tablas nace de las entidades creadas en el capitulo anterior.

Las tablas creadas para nuestra base de datos son las siguientes:

- Tabla TIPO_BUQUE
- Tabla ARQUEO_BRUTO
- *Tabla NORMAS*
- *Tabla FLUIDO*
- *Tabla SISTEMA*
- *Tabla BOMBAS_FIJAS*
- *Tabla MANGUERAS*
- *Tabla ACCESORIOS*
- *Tabla MATERIALES*
- *Tabla ESPECIFICACION_MATERIAL*
- *Tabla ESPECIFICACION_MATERIAL*
- *Tabla ESPECIFICACIÓN DE DISEÑO*
- *Tabla TRIM*
- *Tabla VALVULA_TRIM*

Las tablas conservan el mismo nombre de las entidades de las cuales se derivan.

5.2.2 Campos: Un campo es la implantación de un atributo de datos. Los campos son las unidades mínimas de datos que se han de almacenar en una base de datos.

Existen cuatro tipo de datos susceptibles de ser almacenados: claves primarias, claves secundarias, claves externas y descriptores.

Las claves primarias son campos cuyos valores identifican uno y solo un registro en una tabla.

Las claves secundarias son índices alternativos para la identificación de una entidad. Los valores de una clave secundaria pueden identificar a presencias únicas de la entidad o a un subconjunto de todas las presencias de dicha entidad.

Las claves externas son punteros o enlaces con presencias de archivos diferentes. Una clave externa en una entidad debe ser clave primaria en otra entidad.

Los descriptores son los restantes campos que describen las entidades de empresa.

Miremos cada una de las tablas de la base de datos diseñada para Cotecmar y cada uno de sus campos. Los campos de cada tabla conserva los nombres y características de los atributos declarados en el capitulo anterior:

Tabla TIPO_BUQUE

Campos:

- Codigo: Clave primaria
- Buque : descriptor

Tabla ARQUEO_BRUTO

Campos:

- Codigo : Clave primaria
- Arqueo: descriptor

Tabla NORMAS

Campos:

- Codigo: Clave primaria
- Norma: Descriptor

Tabla FLUIDO

Campos:

- Codigo: Clave primaria
- Fluido : Descriptor

Tabla SISTEMA

Campos:

- Codigo : Clave primaria
- Sistema : Descriptor.

Tabla BOMBAS_FIJAS

Campos:

- Codigo : Clave externa
- No_B_Fijas : Descriptores

Tabla MANGUERAS

Campos:

- Codigo : Clave externa
- No_Mangueras : descriptor

Tabla ACCESORIOS

Campos:

- Codigo : Clave primaria
- Accesorio : Descriptor

Tabla MATERIALES

Campos:

- Codigo : Clave primaria
- Material : Descriptor
- Temp._Max : Descriptor

Tabla ESPECIFICACION_MATERIAL

Campos:

- Codigo : Clave primaria
- Especif_Material : Descriptor

Tabla ESPECIFICACION_MATERIAL

Campos:

- Codigo : Clave primaria
- Especif_Diseño : Descriptor

Tabla ESPECIFICACIÓN DE DISEÑO

Campos:

- Cod_Norma : Clave externa
- Cod_Fluido : Clave externa
- Cod_Sistema : Clave externa
- Cod_Accesorio : Clave externa
- Cod_Materiales : Clave externa
- Cod_Especif_Material : Clave externa
- Cod_Especif_Diseño : Clave externa

Tabla TRIM

Campos:

- Codigo : Clave primaria
- Trim : Descriptor
- Cod_Material : Clave externa
- Cod_Especif_Material : Clave externa
- Cod_Especif_Diseño : Clave externa

Tabla VÁLVULA_TRIM

Campos:

- Cod_Norma : Clave externa
- Cod_Fluido : Clave externa
- Cod_Sistema : Clave externa

- Cod_Accesorio : Clave externa
- Cod_Trim : Clave externa

5.3 DIAGRAMA RELACIONAL DE LA BASE DE DATOS

A continuación se muestra las tablas diseñadas y sus respectivos campos. Además, las relaciones que existen entre ellas.

Figura 6 Base de datos 1

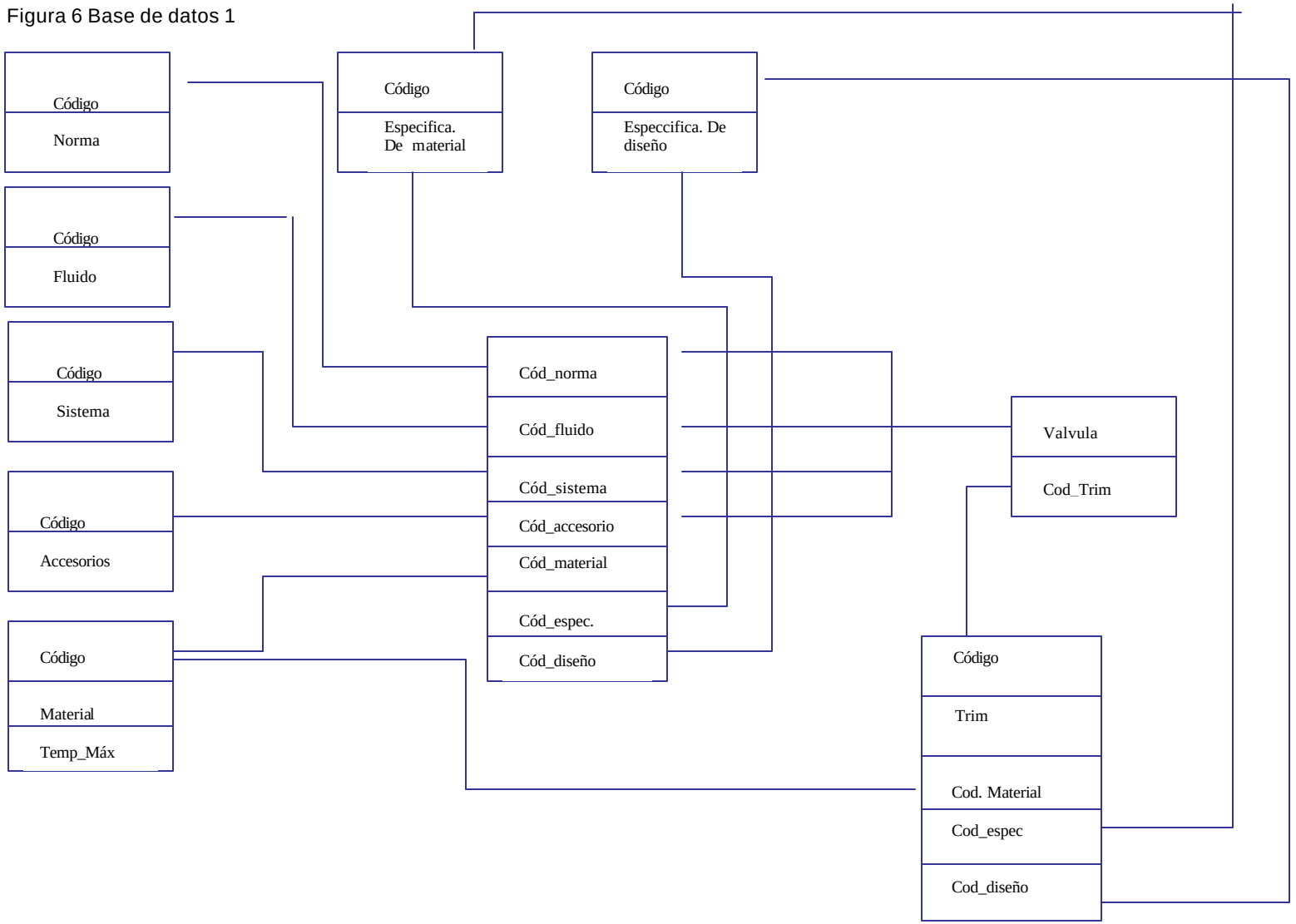
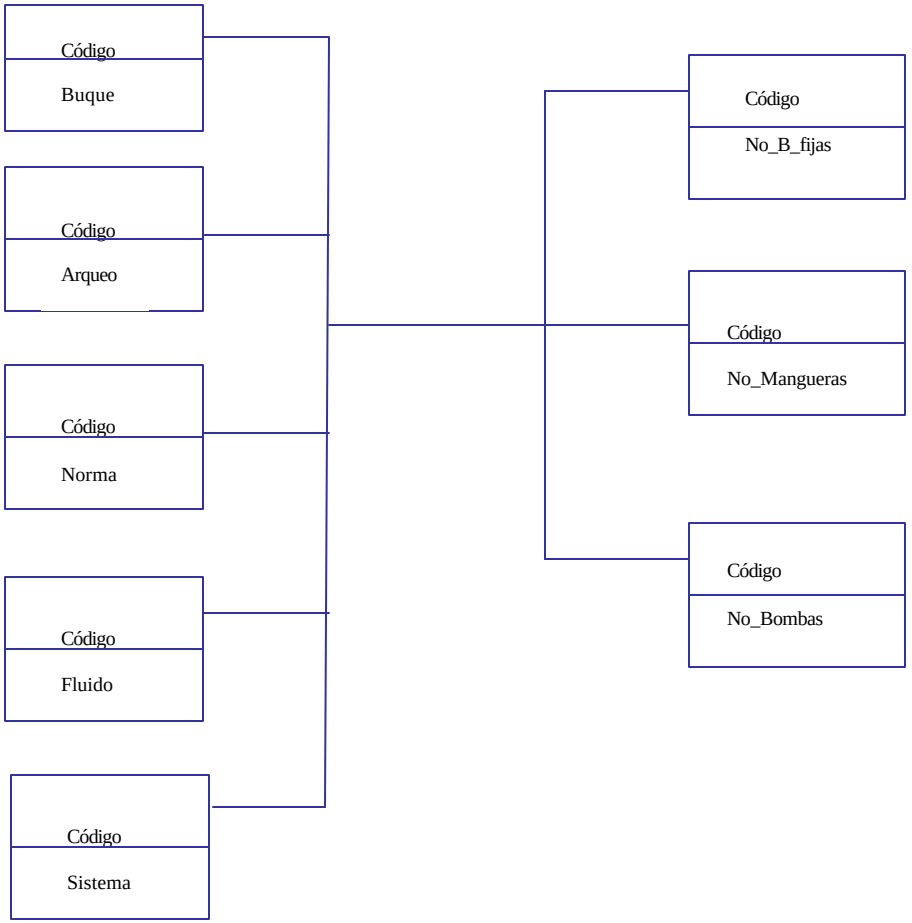


Figura 7 Base de datos 2



6. MODELACION DEL PROBLEMA EN CLIPS

Como primera medida hay que enfocarse en el logro que se quiere alcanzar.

Un software es una herramienta que debe facilitar el trabajo a los usuarios, mas no complicárselo, y ese es nuestro propósito como hemos explicado en capítulos anteriores. Diseñar una herramienta sencilla al usuario que para el caso específico son ingenieros mecánicos o navales encargados de diseñar la estructura de tuberías en sistemas contra incendio y achique en embarcaciones de cualquier tipo y además tener una fuente de información a la mano, más cómoda y rápida. Para esto último podemos pensar en la base de datos como tal, la cual contiene la información necesaria. Con este sistema el usuario podrá hacer una consulta sin necesidad de levantarse de su puesto de trabajo.

Pareciera que no se pierde mucho tiempo, considerando que los libros de consulta están muy accesibles y “siempre a la mano”, pero esto en realidad es un ideal. Al pararse del puesto e ir al estante se pierden unos segundos, buscando el libro se pierden otros segundos, y si no ocurre el contratiempo de que alguien lo tomó y no lo regresó y no se sabe donde está el texto de consulta adecuado (cabe anotar este punto que fue una vivencia personal), se corre el riesgo de que alguien se le acerque a contarle el último chiste. Muy diferente sería hacer click en un menú, buscar, digitar

lo necesitado y obtener respuestas en cuestión de menos segundos que los anteriormente mencionados.

El tipo de base de datos a utilizar para este propósito será acorde a las necesidades que juzgue el departamento de sistemas de la empresa que disponga de este software que para el caso particular es COTECMAR, ya que no está en nuestras manos decidir esto, puesto que cada Departamento de Sistemas en cada empresa tiene sus límites legales en cuanto al uso de software y no es el propósito de este proyecto evaluar si el tipo de base de datos es acorde a las necesidades o no de dicha empresa, además que se es conciente de las capacidades del personal encargado del Departamento de Sistemas para tomas estas decisiones. En conclusión el tipo de base de datos será cualquiera.

La herramienta sugerida para elaborar el sistema propuesto es delphi, gracias a la robustez que ofrece, su aceptación en el mercado, y por la interfaz visual que ofrece al usuario final. Y no solo el usuario final se vé beneficiado sino el programador.

Delphi provee componentes muy poderosos para gestionar bases de datos de cualquier tipo, y por lo mencionado en el párrafo anterior es perfecto para este propósito. Además, existe también un componente CLIPS para ser utilizado desde delphi, lo que se presta para los propósitos planteados.

6.1 LA BASE DE DATOS

El diseño de las bases de datos es muy importante, pues estas deben abarcar toda la información, y estar bien organizada y normalizada además para que la información esté compacta, o en otras

palabras, no ocupe más espacio del debido en el medio de almacenamiento y además optimice la eficiencia de consulta para el motor de base de datos, lo cual se verá reflejado en la optimización del tiempo también. Recuérdese que una base de datos con muchos registros (piénsese en miles) puede enlentecer y hacer ineficientes las aplicaciones, sin importar lo bien hecho y lo trabajado que esté el código. En el capítulo 5 se propone el modelo relacional de la base de datos para lograr este propósito.

6.2 EL EXPERTO

Para lograr el efecto deseado de este proyecto, se considera necesaria y fundamental la transparencia del sistema experto que hay detrás del diseño del software propuesto. No nos ayudaría mucho si tuviéramos que hacer una capacitación en sistemas expertos o CLIPS a los ingenieros encargados de las tareas de diseño de tuberías. Sería engorroso para ellos, una perdida de tiempo para la producción y pensarían además que en lugar de contratarlos a ellos deberían contratar programadores y estarían en lo cierto, lo cual complicaría más las cosas y ese no es el punto. Lo que se plantea como solución para este problema es una especie de máscara, y Delphi es de gran ayuda en este propósito.

En primera instancia, un profesional, por poner un ejemplo, no es un experto hasta que no ha adquirido experiencia, mientras tanto es simplemente un conocedor, así haya hecho la especialidad más especial del mundo. Suena un poco tonto, pero no es lo mismo una persona con mucho conocimiento en un campo que una persona con conocimiento y experiencia en ese mismo campo. Lo mismo ocurre con el software propuesto, el cual inicialmente parte de un conocimiento, que es

precisamente, la base de datos, que por decir algo, es la teoría. Y es exactamente lo que buscamos; esta misma teoría es la que utilizan los ingenieros para el diseño de las tuberías, pero en la experiencia no se siguen al pie de la letra por la teoría, a veces ocurren cambios que no se ciñen a la teoría, bien sea por razones económicas, o por satisfacción del cliente, en fin. Entonces, de acuerdo a esas decisiones que tome el ingeniero, el software propuesto irá aprendiendo, las irá almacenando en su memoria, en forma de facts o hechos y reglas. Pero esto como se logrará?

En primer lugar al iniciar la aplicación, la primera entrada que solicitará será la información básica de la estructura del buque, esto es, la eslora (longitud del buque), la manga (ancho del buque), el arqueo bruto, el tipo de buque(si es de carga, pasajeros o de guerra), además del tipo de reglas o reglas que se seguirán para su diseño, las cuales pueden ser Solas, Marine Engineering, etc.

Una vez capturadas estas características el usuario notificará a la aplicación que los datos de entrada son correctos, entonces la aplicación deberá buscar en su base de conocimientos de sistema experto qué reglas hay que aplicar a los datos entrados por el usuario. Si existen hechos o facts que relacionen los datos de entrada con reglas que el sistema experto haya aprendido, este propondrá al usuario los datos necesarios para lograr el primer paso del diseño de la tubería, que son el número de mangueras a utilizar, el tipo de tubería (anillo horizontal o vertical, por ejemplo para embarcaciones de la marina de los Estados Unidos), el número de bombas mínimas, el diámetro de la tubería, así como su material, y especificaciones de diseño. De no encontrarse relación entre los datos de entrada y los hechos de la base de conocimientos la aplicación recurre a la base de datos donde se encuentra la información propuesta por las normas y casas clasificadoras.

Ahora bien, cómo aprende el sistema experto una nueva regla? Si el ingeniero o usuario no utilizará la información exacta que propone la base de datos y modifica la información propuesta la aplicación guardará en un buffer estas modificaciones hasta que el usuario decida hacer un nuevo buque o decida cerrar la aplicación. En este momento la nueva información la aprende la aplicación del usuario y se guardará automáticamente en la base de conocimientos.

6.3 EL ALGORITMO

Hasta ahora hemos explicado la secuencia de pasos a nivel de usuario. Pero que debe hacer el programador para lograr lo descrito anteriormente?

La interfaz visual deberá visualizar los campos de entrada, o de estructura básica del buque, en un panel. En otro panel mostrará los campos que propone para el diseño de la tubería, indistintamente si son de la base de datos o de la base de conocimientos, estos son básicamente los campos necesarios que al usuario le interesa visualizar.

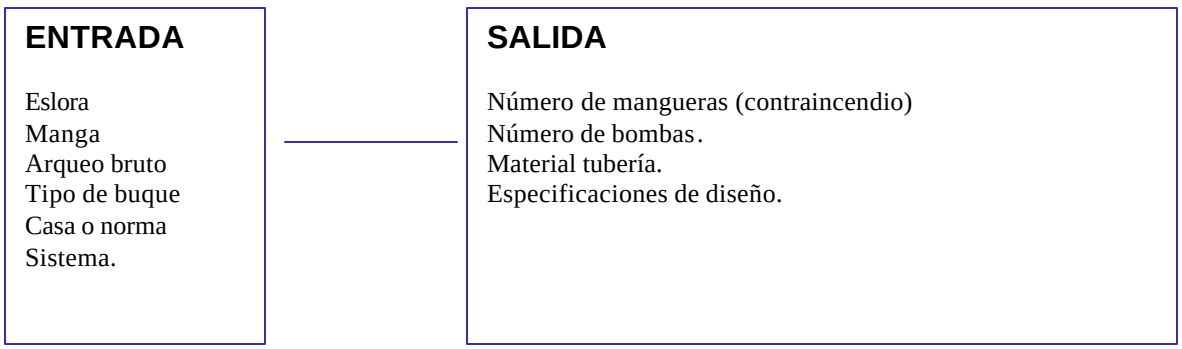


Figura 8 Campos a visualizar.

Entrando más al plano real, las entradas podrían visualizarse mediante una instancia de la clase TlistView de Delphi. (Ver figura 9)

Figura 9 Lectura de datos

Además debe haber una especie de memo que notifique algunos detalles de la selección, aunque esta parte está más ligada al sistema experto, dicho memo o interfaz extra debe ser de lectura para información o escritura si el usuario diseñador de la tubería desea especificar o aclarar algo en particular.

Supongamos que es la primera vez que se ejecuta la aplicación, entonces la experiencia es nula. Por lo tanto al buscar en la base de conocimientos no se encontrará nada y la aplicación buscará en la base de datos las especificaciones propuestas por las normas. Para lograr esto a nivel de programación es necesario declarar una variable de tipo archivo de texto (que es el archivo .clp que contendrá las reglas generadas automáticamente) en el código fuente del programa, entonces, si al intentar abrir el archivo este no existe, significa que el programa aún no ha aprendido nada nuevo y por consiguiente pasará a buscar la información requerida para el diseño en la base de datos propuesta. Si el usuario está conforme entonces continua su diseño basándose en dichas especificaciones. Si no, es importante que notifique o enseñe a la aplicación que modificaciones se tomarán. Por ejemplo, para un buque de pasajeros con un arqueo bruto mayor de 4000 toneladas, Solas recomienda un número mínimo de 3 bombas. Supongamos que la Armada Nacional de Colombia hace un requerimiento de varios buques de 5000 toneladas y las normas a seguir son Solas a excepción de un detalle, cada buque deberá usar máximo 2 bombas por razones de escasez de recursos y baja probabilidad de accidentes o riesgo de daños por combate. La aplicación en primera instancia propondrá un mínimo de 3 bombas (Ver figura 6.3), mas sin embargo el diseñador de la tubería especificará que solamente se usarán dos.

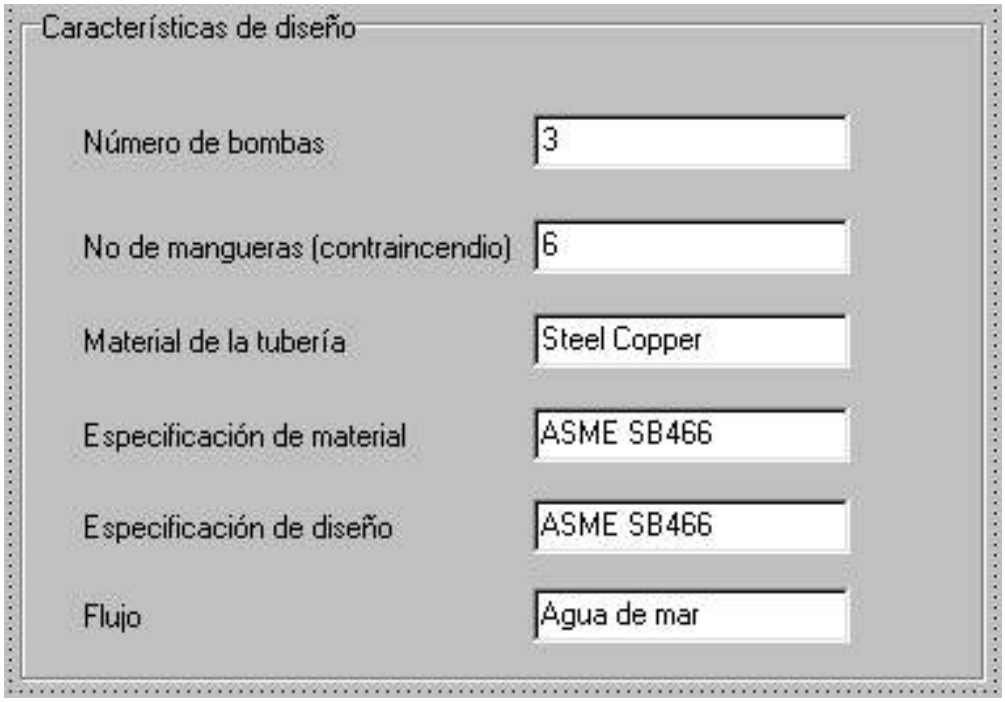


Figura 10 Especificaciones para sistema contraincendio

Una vez hecha la modificación, esta se almacena en un buffer de la memoria. De no hacerse más modificaciones la aplicación entiende que debe aprender de la modificación hecha y añadirla en la base de conocimientos de la siguiente manera:

En primer lugar será el de crear el archivo **.clp**, suponiendo que sea esta la primera vez que el sistema tenga que aprender, y sobre este se escribirán las reglas generadas. Para el ejemplo propuesto se genera una regla para nuestro sistema experto. Como la sintaxis de la regla es siempre la misma bastará con concatenar la estructura de la sintaxis de CLIPS con los parámetros de entrada para formar el LHS de la regla y para formar el RHS utilizaremos la variación hecha por el

usuario, luego almacenamos la cadena en el archivo de texto **.clp** que es el que CLIPS requiere para cargar las reglas en memoria. Esto quedaría entonces:

(defrule regla1

(tipobuque <entrada1>

(manga <entrada2>)

(eslora<entrada3>)

(arqueobruto <entrada4>))

=>

(assert (numeromangueras<modificacion1>))

Será un assert por cada modificación hecha a los resultados iniciales. El primer campo del hecho, para este caso numeromangueras se obtiene del nombre del campo dentro de la base de datos y modificación1 es el valor modificado por el usuario. Al hacer el assert dentro de la regla se dispara en el componente CLIPS para Delphi el evento onassert, que recibe como parámetro el hecho insertado, dicho evento se aprovecha en el momento en que la regla se ejecute para de esta manera obtener dicho hecho e informar de él al usuario la próxima vez que entre como parámetros las características que se encuentran dentro de la regla en el LHS. Sin embargo, debemos hacer una segunda regla que elimine el assert hecho por la primera regla, ya que de no hacerlo, la primera regla no volverá a hacer su assert debido a que CLIPS solo acepta un hecho o fact a la vez, no acepta duplicaciones y es posible que para buques con otras características se requieran diseños similares. Además no es necesario tenerlos en memoria o agrandar el archivo si la acción a tomar ya está en la regla. La manera para quitar el hecho insertado sería

(defrule quitar-de-regla1 (

?quitar<-(numeromangueras<modificacion1>)

=>

(retract ?quitar))

La sentencia en el LHS de la regla significa que si se cumple el patrón indicado, su identificador de hecho se guardará en la variable ?quitar, (la interrogación adelante significa que es una variable), para luego hacer un retract de lo que se encuentra en esa dirección.

La siguiente vez que el usuario ingrese un buque con características similares estas características entrarán con un deffacts y CLIPS las cargará en memoria para de esta manera determinar si se cumplen dichas reglas, esto se logra gracias al método Reset que ofrece la clase TClips del componente CLIPS para Delphi.

Como se describió anteriormente el programa buscará si existe el archivo de texto con las instrucciones CLIPS con extensión **.clp**, y como para este caso si existe, lo primero que hay que hacer es cargar las reglas existentes en memoria y para lograr esto basta con aprovechar el método LoadFromFile de la clase Tclips utilizando la instrucción Tclips.LoadFromFile (nombredelarchivo.clp), y una vez cargadas las reglas hacemos los respectivos asserts de los hechos ocurridos como se describió en el párrafo anterior, pero ocurre que la clase solamente carga en memoria mediante el método LoadFromFile, lo cual nos dice que debemos crear un archivo de texto temporal donde escribamos los datos leídos del buque para luego sí utilizar la propiedad, entonces, una vez cumplidas o no cumplidas las reglas, los hechos agregados serán removidos, pues no es necesario tenerlas almacenadas ni en memoria ni en medio magnético. Para removerlas es muy sencillo pues

como no queremos guardarlas en el archivo **.clp** simplemente damos mediante código el comando (clear) una vez ejecutadas las reglas y mostrados los resultados, finalmente borramos el archivo de texto temporal creado.

Para ir agregando reglas con diferentes nombres automáticamente la aplicación creará un archivo de tipo número entero. Al lanzarse la aplicación por primera vez el contenido del archivo es cero, pero al ingresar una regla este se incrementa en 1 y al crearse la nueva regla a insertar se concatenará este valor quedando las reglas guardadas sucesivamente como regla1, regla2, etc.

Una vez incrementado el contador deberá guardarse el nuevo valor en el archivo.

Una vez establecidas las especificaciones del diseño el usuario pasa a una siguiente interfaz en donde la aplicación procederá al cálculo de la capacidad de las bombas a utilizar. Esto es importante porque necesitamos que la bomba sea capaz de abastecer completamente todas las dependencias en caso de incendio o para el caso de achique debe ser capaz de desalojar cierta cantidad de agua en un tiempo determinado basado en las normas establecidas por las casas clasificadoras o normas internacionales. Sin embargo, no se puede ser extremista y adquirir bombas con una exagerada capacidad, puesto que esto significaría incurrir en un gasto innecesario. Hay que buscar el equilibrio y para esto se tiene en cuenta la cantidad de fluido que la bomba puede succionar con base en una altura. La altura es una medida de la energía, para el caso de las bombas, esto se debe a que mientras más alta sea la posición de la bomba con respecto a la boquilla de la manguera de succión, mayor será la energía requerida para llevar a cabo dicha succión.

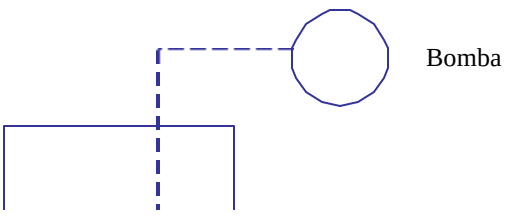


Figura 11 Funcionamiento de una bomba

Esto no significa que la altura del estanque sea el único factor que incide en la energía requerida por la bomba. Durante el trayecto del fluido a través de la tubería las pérdidas de energía por fricción son considerables en trayectos largos, y son mucho más considerables si tenemos en cuenta que las tuberías no son lisas y rectas sino que el fluido chocará con codos dentro de la tubería, cruces, válvulas, etc.

Estos son los accesorios de las tuberías para los cuales tenemos una tabla definida en nuestra base de datos. Para cada accesorio, existe un valor por pérdida para determinado fluido, y la suma de todas las pérdidas de todos los accesorios de las tuberías nos resultan en la pérdida total. Estas pérdidas por accesorios individuales se pueden manejar dentro del algoritmo del programa con un Case, el cual determinará cual es el valor por pérdida para un accesorio de un determinado código. La lectura de cada accesorio puede hacerse de manera similar a como se propone la lectura de las características del buque utilizando la clase TListView de Delphi. Una vez arrojado el resultado de la pérdida total el diseñador de la tubería debe comparar con unas gráficas que otorgan los fabricantes de las bombas que hay en el mercado. Dicha gráfica es conformada por los ejes Energía (altura) vs. Capacidad (Galones de agua) (Ver figura 12).

En realidad se grafican dos funciones en el mismo plano, las cuales nos indican la capacidad de succión de la bomba, y las permisibilidad de pérdida de la bomba, esta última más reconocida como gráfica de N.P.S.H.

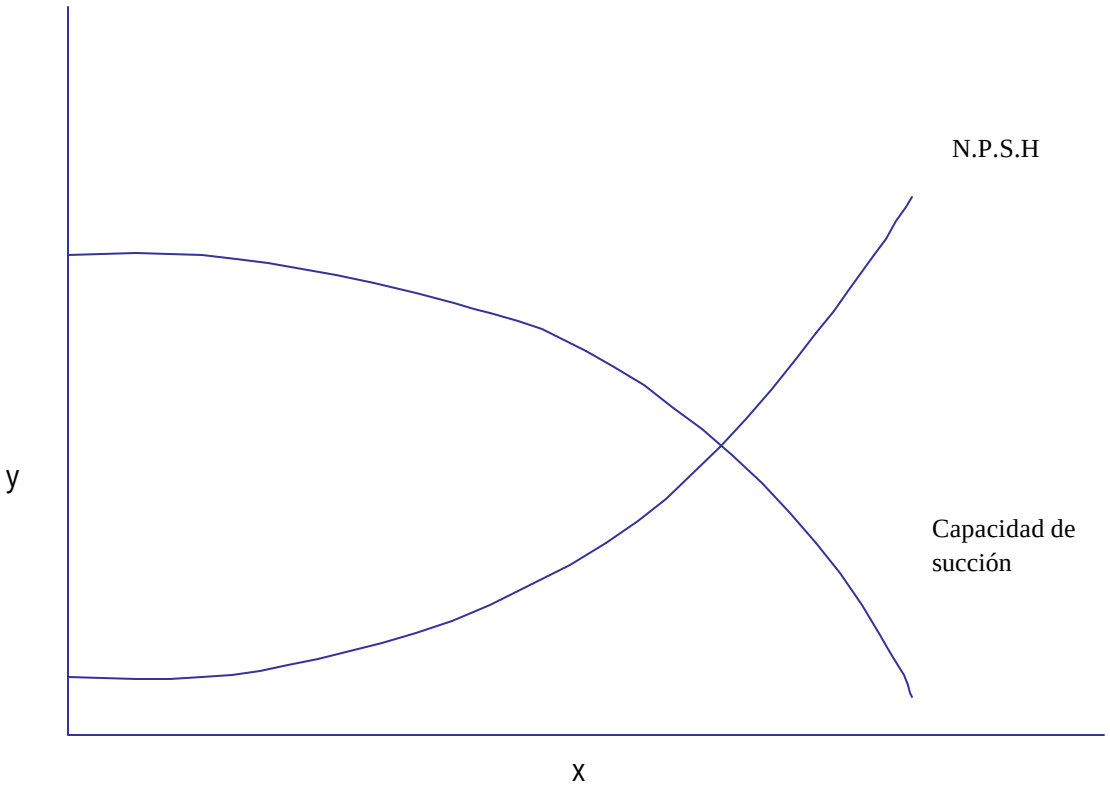


Figura 12 Gráfica de N.P.S.H.

X = Energía (altura)

Y = Capacidad (Galones de agua)

El punto de intersección es en donde la bomba trabaja de manera óptima, puesto que abastece suficiente agua y además no se está subutilizando. En el procedimiento convencional el ingeniero debe buscar que tipo de bomba es la adecuada o en el peor de los casos rediseñar el sistema de tuberías poniendo o quitando accesorios. Con el sistema propuesto inicialmente ocurre lo mismo, pero este sistema tiene una gran ventaja y es que aprende.

Si el tipo de bomba (considerada como accesorio de la tubería) propuesto, no satisface las necesidades del ingeniero y es requerido un cambio, es conveniente que el ingeniero indique al programa cual es la bomba más adecuada escogiendo de una lista que la aplicación propone, la cual inicialmente es creada en tiempo de diseño con base en las documentaciones existentes y de esta manera es como el diseñador le enseña a CLIPS que tipo de bomba debe escogerse en ese caso. Hecho esto, el programa toma la lista de accesorios creada y va concatenando cada item de dicha lista que representa un accesorio diferente dentro de una cadena de caracteres con la estructura de la nueva regla a añadir en el archivo **.clp** que contiene nuestra base de conocimientos, quedando la estructura de esta manera:

(defrule regla_n

(<accesorio1>)

(<accesorio2>)

...

(<accesorio_m>))

=>

(assert (tipodebomba<modificacionX>))

De tal forma que la siguiente vez que el programa reciba sus datos de entrada, cargará la lista de accesorios en el buffer del componente CLIPS de Delphi, utilizando los métodos Reset y LoadFromFile de la clase Tclips de la misma manera que se propone para las reglas de las especificaciones de diseño.

Cabe anotar también que si se dispara la regla se hace un assert del tipo de bomba a implementar y este no es necesario tenerlo almacenado, por lo tanto se elimina con un retract.

(defrule quitar-de-regla_n (

?temp<-(tipodebomba<modificacionX>))

=>

(retract ?temp))

Si no se trata de un cambio de bomba, sino de un cambio total en el sistema de tuberías la aplicación aprenderá qué accesorios y qué cantidad de estos se utilizaron en un buque con las características especificadas, de esta manera la próxima vez que al programa se le proponga un buque con esas características propondrá los accesorios y el número de estos antes que el usuario le indique y mostrará inmediatamente el resultado de las pérdidas. El modelado de este procedimiento en CLIPS se logra haciendo una regla cuyo LHS contenga la lista de accesorios

actuales que el diseñador de la tubería propone, añadiendo del lado del RHS el listado de los accesorios que el usuario indica. Quedando de la forma:

(defrule regla_k

(<accesorio1>)

(<accesorio2>)

...

(<accesorio_m>))

=>

((assert(<accesorio1n>))

((assert(<accesorio2n>))

...

((assert(<accesorio_m>)))

No necesariamente el número de accesorios del LHS debe ser igual al número de accesorios del RHS. Se agrega la regla al archivo clp. Cabe insistir en que por cada assert que se dispare al ejecutarse una regla, se almacenará en memoria los hechos declarados en el RHS de la regla y estos no cumplen ninguna función para nuestro diseño, mas si podrían entorpecer el desempeño de la aplicación, por consiguiente conviene hacer el respectivo retract de los hechos insertados al dispararse la regla:

(defrule quitar-de-regla_k (

?quitar1<-(<accesorio1n>)

?quitar2<-(<accesorio2n>)

?quitar3<-(<accesorio3n>)

...

=>

(retract ?quitar1)

(retract ?quitar1)

(retract ?quitar1)...)

Nótese que por cada accesorio a remover hay que declarar una variable de indicador de índice del hecho. Para lograr la automatización en la generación de esta regla es necesario concatenar una variable ?quitar, con un índice que puede ser copiado del atributo índice de la clase Tlist de Delphi, convertido en carácter y concatenarlo con el string ?quitar para que pueda quedar de la forma ?quitar<indice>.

De esta manera ya el ingeniero no perdería tiempo revisando las gráficas que los proveedores de bombas suministran, sino, que contactará inmediatamente al proveedor de bombas o hará una reestructuración de la tubería basándose en la experiencia de la aplicación. En la medida en que el ingeniero usuario vaya enseñando al programa y aumente la diversidad y el número de reglas en la fabricación de buques mayor será la base de conocimientos y por consiguiente, mayor será la experiencia y utilidad de la aplicación.

El resultado de este proyecto de investigación es una documentación detallada y completa en la que se resume el proceso actual de diseño de tuberías contra-incendio y achique en embarcaciones de todo tipo, y se propone además un sistema para mejorar tal procedimiento. Dicho sistema propuesto

ofrece un diseño de una base de datos óptima, bien relacionada, con la información necesaria que proveen los organismos internacionales para la seguridad de la vida humana en el mar, y una descripción detallada de la información almacenada en esta base de datos.

Se explica además en detalle como implementar una herramienta para sistemas expertos para que el programa o aplicación adquiera los conocimientos de las personas que diseñan los buques basándose no solo en las normas preestablecidas sino en la propia experiencia y en los requerimientos necesarios en ocasiones determinadas. Como frutos personales, cosechados de este proyecto hemos aprendido a implementar conocimientos adquiridos durante nuestra carrera estudiada, y además conocimientos extra curriculares, en el campo de sistemas expertos buscando el mejoramiento y la calidad de los sistemas de diseño en ingeniería, utilizando herramientas de informática.

CONCLUSIONES

En conclusión cualquier persona con conocimientos de programación en lenguajes de alto nivel, y con ayuda de esta documentación está en capacidad de hacer la implementación de un software que proporcione la ayuda necesaria para el diseño de sistemas contra-incendio y achique en embarcaciones de todo tipo, haciendo uso de una base de datos y una herramienta de sistema experto.

Además, aunque el objetivo inicial fue estudiar solo los sistemas de contraincendio con agua de mar y achique, se descubrió durante la investigación que el diseño de todos los sistemas de tuberías de un buque sigue los mismos parámetros.