CAPITULO 1: INTRODUCCIÓN.

1.1 Introducción al proyecto.

Todos y cada uno de nosotros pasamos los días y las horas de nuestra vida tomando decisiones. Alguna decisiones tienen una importancia relativa en el desarrollo de nuestra vida, mientras que otras tienen una repercusión mayor.

En un ambiente empresarial, la toma de decisiones es también una tarea cotidiana, pero además tiene una repercusión mayor. Dada la gran competitividad que existe actualmente en el tejido empresarial, tomar las decisiones adecuadas en cada momento se antoja vital para el devenir de la empresa. De ahí que la toma de decisiones en la empresa tenga cada vez mayor importancia.

Para los administradores, el proceso de toma de decisión es sin duda una de sus mayores responsabilidades. En la actualidad, la toma de decisiones en una organización se circunscribe a una serie de personas que forman parte de un mismo proyecto. Dejar en manos de una sola persona la toma de decisiones puede suponer para la organización un riesgo muy grande.

Con frecuencia se dice que las decisiones son algo así como el motor de los negocios y en efecto, de la adecuada selección de alternativas depende en gran parte el éxito de cualquier organización.

1.1.1. Introducción a la Toma de Decisiones (TD).

La Toma de Decisiones, es el proceso mediante el cual se realiza una elección entre un conjunto de alternativas o formas de resolver un problema. Este proceso de Toma de decisiones se puede presentar en cualquier contexto de la vida laboral o familiar.

1.1.2. Introducción a la Toma de Decisiones en Grupo (TDG).

Un problema de toma de decisiones en grupo (GDM o TDG) es aquel en el que intervienen varios expertos o decisores (que pueden proceder de distintos campos), y cuyo propósito es encontrar la mejor solución para dicho problema. En aquellos problemas en los que intervienen varios expertos, se suelen obtener soluciones de mejor calidad que en los que interviene un único experto, aunque no siempre es así.

Un problema de Toma de Decisiones en Grupo se caracteriza por:

i) La existencia de un problema que hay que resolver y un conjunto $X=\{x_1, x_2, x_3, \ldots, x_n\}$ de posibles alternativas solución para dicho problema.

ii) La existencia también de un conjunto de expertos $E=\{ e_1, e_2, e_3, \ldots , e_m\}$. Estos expertos expresan sus opiniones o sus preferencias sobre el conjunto de alternativas. Las opiniones o referencias del experto e_i, se representan: P_i ó Pe_i.

iii) Un objetivo, que no es otro que encontrar una solución común (compartida por todos los expertos) a dicho problema.

Un problema de Toma de Decisiones en Grupo se puede estructurar básicamente en dos etapas:

- Proceso de consenso.

- Proceso de selección de alternativas.

Un Proceso de Consenso es la búsqueda del mayor acuerdo del conjunto de opiniones dadas por los expertos, a partir de la opinión inicial de cada uno de ellos. Este Proceso de Consenso, está formado por varias sesiones o rondas de discusión. En cada una de ellas, los expertos dan sus opiniones y se intenta que progresivamente converjan a la opinión colectiva. Este proceso acaba cuando:

- Se han agotado el número de rondas previstas para el proceso. Se debe de decidir si prolongar más el proceso o culminar en ese punto.

- Se ha alcanzado un umbral de consenso fijado inicialmente. Se ha alcanzado consenso.

El Proceso de Selección, es el paso posterior al Proceso de Consenso. Una vez alcanzada una posición común entre los expertos, o agotado el tiempo asignado a este Proceso de Consenso, hay que seleccionar la alternativa o conjunto de alternativas que dan la solución mejor o más adecuada al problema.

1.2. La importancia del Proceso de Consenso.

El Proceso de Consenso es la etapa más importante de la Toma de Decisiones en Grupo. Hablamos de la importancia del Proceso de Consenso, porque gracias a éste obtenemos una opinión colectiva de todos los expertos involucrados en la discusión del problema. Además, es importante también porque todos los expertos se encuentran identificados con dicha opinión colectiva.

El esquema básico de funcionamiento del Proceso de Consenso lo podemos ver en la siguiente figura:

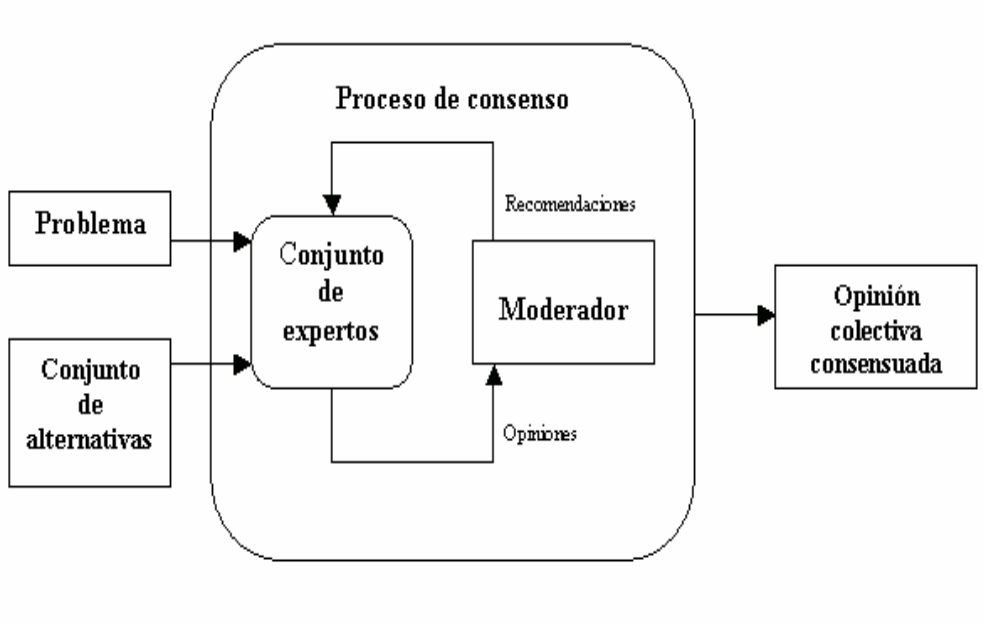


Figura 1: Estructura básica del Proceso de Consenso.

1.3. Sistema de Apoyo al Consenso.

En este Proyecto de Fin de Carrera, vamos a diseñar e implementar un Sistema de Apoyo al Consenso que automatice el Proceso de Consenso de un problema de Toma de Decisiones en Grupo.

Lo que pretendemos en este Proyecto de Fin de Carrera es primero diseñar un Modelo de Consenso, y posteriormente implementarlo.

En un Sistema de Apoyo al Consenso se maneja mucha información acerca de los problemas tales como: parámetros del problema, alternativas de solución, expertos asignados al problema, opiniones de cada uno en las distintas rondas del Proceso de Consenso, etc... Por ello vamos a usar un formato de Base de Datos **MySQL** para la gestión de los datos.

La implementación del Modelo de Consenso la vamos a llevar a cabo utilizando el lenguaje de programación **Java** y el entorno de desarrollo **NetBeans**. La aplicación estará basada en el modelo cliente-servidor, por lo que tendremos que emplear el lenguaje **html** y un servidor **Apache Tomcat** para su implementación.

1.4. Propósito.

Diseño e implementación de un sistema de apoyo al consenso, basado en técnicas Soft-Computing, que automatice el proceso de consenso en problemas de toma de decisiones en grupo en un contexto de incertidumbre.

1.5. Objetivos.

1. Revisión y análisis de la bibliografía.

2. Diseño de un modelo teórico de consenso automatizable para problemas de Toma de Decisiones en Grupo en ambiente de incertidumbre.

3. Análisis, diseño e implementación de dicho sistema de apoyo al consenso.

4. Verificación y validación del sistema.

CAPITULO 2: ANTECEDENTES.

2.1. La Toma de Decisiones (TD).

Es el proceso por el cual una persona (o grupo de ellas) debe de escoger entre dos o más alternativas a la hora de resolver un determinado problema.

La Toma de Decisiones (TD) es un proceso complejo y una de las actividades fundamentales de los humanos. Para algunos autores, la TD es una característica fundamental que diferencia al género humano de los animales. Constantemente nos estamos enfrentando en nuestra vida a situaciones en las que existen varias alternativas para un determinado problema que se nos platea. Al menos en algunas de esas ocasiones, tenemos que decidir cuál de estas alternativas es mejor, o cuál llevar a cabo.

En TD se habla de dos áreas bien diferenciadas:

A) La selección de alternativas que trata de encontrar el mejor conjunto de alternativas para dar solución a un problema de Decisión.

B) El Consenso, que mide el grado de acuerdo existente entre los expertos participantes en el problema de Toma de Decisiones.

Un problema clásico de Toma de Decisiones, consta de los siguientes elementos básicos:

- Un conjunto de alternativas o decisiones posibles. Son las distintas vía de solucionar el problema planteado. No existe un número ni mínimo ni máximo.

- Un conjunto de estados de la naturaleza que definen el contexto de definición del problema.

- Un conjunto de valores de utilidad, cada uno de los cuales está asociado a un par formado por una alternativa y un estado de la naturaleza.

- Una función que establece las preferencias del experto sobre los posibles resultados.

2.1.1. Etapas de la Toma de Decisiones.

El proceso de toma de decisiones comienza con la identificación de una situación en la que hay que tomar una decisión. Es decir la identificación de un problema con varias alternativas de solución (*Reconocimiento de la necesidad de tomar una decisión*). Ejemplo: Comprar un coche.

Una vez identificado el problema, la siguiente etapa es la del *Análisis de decisión*. En ella se identifican aquellos criterios que son de importancia para la toma de decisiones. En nuestro ejemplo anterior tendremos infinidad de posibles criterios. Algunos de ellos podrían ser: marca, precio, modelo, número de puertas, tamaño, etc...

De este conjunto de criterios, se escogerán aquellos criterios que se consideren oportunos. Para algunas personas pueden ser importantes unos pocos, y para otra otros.

Los criterios identificados en la anterior etapa no tienen por qué tener la misma importancia. Puede ser que alguno de ellos tenga más relevancia que el resto. Es necesario por tanto ponderar cada uno de ellos, y priorizar su importancia en la decisión (*Asignar pesos a los criterios*). En nuestro ejemplo anterior, es razonable que los criterios de tamaño y coste tenga mayor relevancia que el resto.

La siguiente etapa de la toma de decisiones consiste en identificar y desplegar el conjunto de todas las alternativas posibles de solución al problema (*Identificar el conjunto de alternativas*). En el ejemplo que estamos desarrollando, posibles alternativas serían los modelos de coches que vamos a analizar.

La penúltima de las etapas de la Toma de Decisiones es aquella en la que se analizan cada una de las alternativas identificadas en la fase anterior en función de los criterios de ponderación anteriores. Se evalúa de manera crítica cada una de ellas. De ahí se obtendrá una serie de ventajas e inconvenientes (*Evaluación de las alternativas*).

Por último en la etapa final del proceso de toma de decisiones se escoge la alternativa deseada. Se suele escoger aquella alternativa de mejor valoración (*Seleccionar la mejor alternativa*).

2.1.2. Criterios de Clasificación.

Existen una serie de criterios a la hora de clasificar los problemas de Toma de Decisiones pueden clasificarse según una serie de criterios.

Según el contexto de la Toma de Decisiones, tenemos:

A) Ambiente de certidumbre: La utilidad de cada alternativa se conoce con exactitud y precisión.

B) Ambiente de riesgo. El conocimiento sobre las alternativas consiste en sus distribuciones de probabilidad.

C) Ambiente de incertidumbre. En esta situación no conocemos la probabilidad de las alternativas. La utilidad de cada una de ellas, se caracteriza de forma aproximada.

Según en número de expertos involucrados en el problema tenemos:

A) Toma de Decisiones simple → Interviene una persona nada más en el proceso.

B) Toma de Decisiones en Grupo → Intervienen varias personas en el proceso.

Según el número de criterios, pueden intervenir uno o más criterios a la hora de evaluar las distintas alternativas de solución del problema.

2.1.3. Modelado de preferencias.

El modelado de preferencias consiste en elegir el dominio de expresión y la estructura para representar las preferencias de los expertos sobre las distintas alternativas de solución de un problema de TD. Estas preferencias se pueden fijar de distintos modos:

- Numéricas o binarias → Este tipo de modelado se usa cuando las preferencias se fijan mediante valores numéricos exactos en un determinado intervalo. Por ejemplo un valor 0.77 en una escala [0,1].

- Mediante intervalos → Cuando el grado de certeza del problema es menor, usar un valor numérico exacto se antoja complicado. En estos casos, podemos usar un intervalo numérico en vez de un valor. Por ejemplo el valor [0.65,0.75] en una escala [0,1].

- Lingüísticas → En muchos problemas y situaciones de la vida real, a veces es más adecuado usar una variable cualitativa que una cuantitativa. En estos casos, un planteamiento mejor para valorar el problema puede ser el de utilizar variables lingüísticas en vez de valores numéricos. El planteamiento lingüístico difuso, valora aspectos cualitativos por medio de variables lingüísticas.

El enfoque lingüístico es adecuado cuando intentamos calificar fenómenos relacionados con la percepción humana.

2.2. La Toma de Decisiones en Grupo (TDG).

Llamamos Toma de Decisiones en Grupo a aquel proceso de Toma de Decisiones en el que intervienen dos o más personas. A cada una de las personas que intervienen en el proceso se les llama expertos.

La Toma de Decisiones en Grupo tiene un serie de ventajas que describimos a continuación:

A) *Información y conocimiento más completo* → Un grupo es capaz de recopilar más información que una persona individualmente. Se tiene acceso a más fuentes informativas. De la misma manera, un grupo de expertos aporta mayor diversidad (distintos puntos de vista, más énfasis en ciertos aspectos, experiencia, etc...) a la toma de decisiones.

B) *Mayor aceptación de la solución* → Muchas decisiones fracasan al no estar un sector de la empresa de acuerdo con la decisión tomada.

C) *Aumenta el número de puntos de vista diferentes* → Cada uno de los integrantes tiene un punto de vista propio, que difiere en cierta mediada del de los demás. De esta manera, la cantidad y tipos de opciones son mayores que las del individuo que trabaja solo.

D) *Incremento de la legitimidad* → Los métodos democráticos son aceptados por todos los componentes de un grupo social.

E) *Disminuyen los problemas de comunicación* → Al participar el grupo en la toma de decisiones, todos sus integrantes tienen conocimiento del proceso.

La Toma de Decisiones en Grupo presenta también una serie de inconvenientes que describimos a continuación.

- *Requiere mucho tiempo* → Reunir y organizar al grupo de expertos conlleva su tiempo. Es ideal por tanto, el confeccionar un programa de tiempos. Los grupos consumen más tiempo en alcanzar una decisión que una persona de manera individual.

- *Presiones de aceptación* → Si bien todos los expertos que forman el grupo expresan sus opiniones, sugerencias y recomendaciones en libertad, a veces puede existir cierta presión sobre ellos para que se alcance una posición global. Esta presión puede provocar que el grupo pase por alto un consejo o sugerencia positiva de alguno de los presentes.

- *Responsabilidad ambigua* → Los miembros del grupo comparten la responsabilidad. Por tanto la responsabilidad individual desaparece, dándoles especial significación a los resultados.

- *Estancamiento del proceso* → En ciertas ocasiones puede ocurrir que el grupo se estanca y se puede mostrar incapaz de llegar a un acuerdo sobre qué soluciones recomendar. Esto se suele dar en aquellas situaciones en las que se subdivide el grupo.

2.2.1. Estructura de un problema de Toma de Decisiones en Grupo.

La Toma de Decisiones en Grupo es un proceso, y como tal a su vez está dividido en otros procesos. Los dos procesos básicos de un problema de Toma de Decisiones en Grupo son el Proceso de Consenso y el Proceso de Selección. Podemos verlos en la siguiente figura:

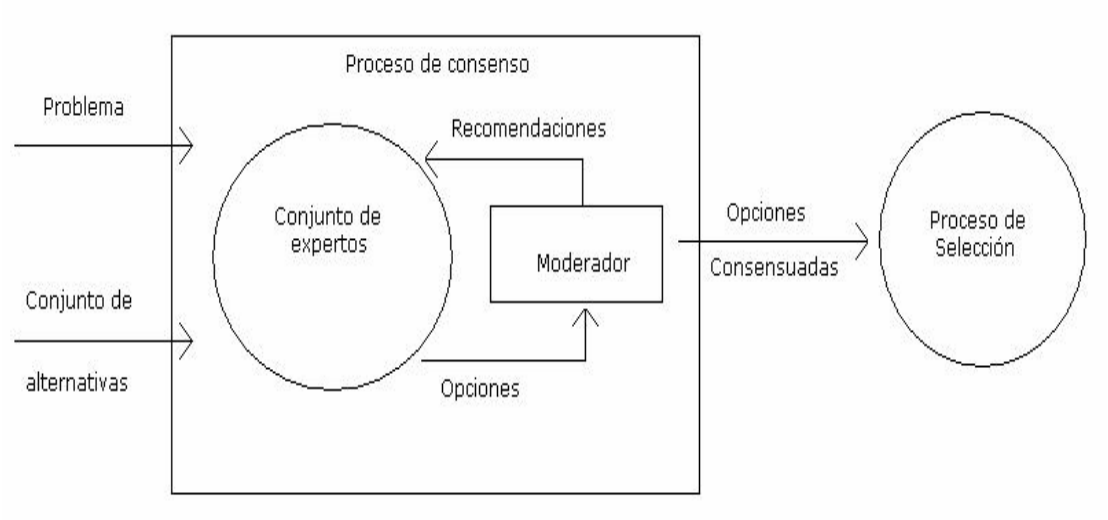


Figura: Proceso de Consenso y de Selección de la TDG.

El Proceso de Consenso es el más importante de la Toma de Decisiones. A partir de una descripción del problema y un conjunto de alternativas, los expertos van a dar sus preferencias y discutir el problema durante un determinado número de rondas. Una vez se hayan consumido el total de rondas fijadas para discutir el problema, si no se ha alcanzado un determinado umbral de consenso previamente, el Proceso de Consenso acabará, y el problema pasará al Proceso de Selección.

El Proceso de Selección es posterior al Proceso de Consenso. A partir de las preferencias colectivas de los expertos discutidas en el Proceso de Consenso, en el Proceso de Selección se escogerá aquella alternativa que sea la mejor solución para el problema.

A la hora de representar la información en la Toma de Decisiones en Grupo, no existe una única posibilidad. Algunas de las posibilidades de representar la información más conocidas son las siguientes:

A) *Pares de alternativas* → En este nivel, son calculados tanto el grado de consenso como la medida de proximidad para cada par de alternativas. Se realiza por parejas de alternativas.

B) *Alternativas* → En este nivel, se calcula la medida de proximidad y el grado de consenso para cada alternativa. Se realiza de manera individual para cada alternativa.

C) *Relaciones de preferencia o expertos* → En este nivel es calculado el grado de consenso global entre todos los expertos, y la medida de proximidad entre cada opinión individual y la del grupo.

En los problemas de Toma de Decisiones en Grupo, los expertos suelen usar relaciones de preferencia para modelar sus opiniones.

Las relaciones de preferencia, permiten modelar las preferencias de cada una de las alternativas del problema frente al resto. Cada valor expresa la bondad de la alternativa frente a la otra.

Una relación de preferencia es una matriz cuadrada, de tamaño MxM. Esta M indica el número total de alternativas de solución al problema. De esta manera, la celda (i,j) servirá para denotar el grado de preferencia de la alternativa i sobre la alternativa j para dicho experto. En cada una de las rondas del proceso de consenso tendremos una matriz de estas para cada uno de los expertos.

$$Pe_1 = \begin{pmatrix} P^1_{11} & P^1_{12} & \dots & P^1_{1M} \\ \dots & \dots & \dots & \dots \\ P^1_{M1} & P^1_{M2} & \dots & P^1_{MM} \end{pmatrix} \qquad Pe_N = \begin{pmatrix} P^1_{N1} & P^1_{N2} & \dots & P^1_{NM} \\ \dots & \dots & \dots & \\ P^1_{N1} & P^1_{N2} & \dots & P^1_{NM} \end{pmatrix}$$

Pe_i = Matriz de preferencias del experto i.

P^k_{ij}= Preferencia de la alternativa i sobre la alternativa j para el experto k.

Un ejemplo de una matriz de preferencias para un determinado problema con cinco alternativas podría ser:

$$P1 = \begin{pmatrix} 0.5 & 0.439 & 0.373 & 0.556 & 0.649 \\ 0.561 & 0.5 & 0.583 & 0.651 & 0.615 \\ 0.627 & 0.417 & 0.5 & 0.709 & 0.68 \\ 0.444 & 0.349 & 0.291 & 0.5 & 0.603 \\ 0.351 & 0.385 & 0.32 & 0.397 & 0.5 \end{pmatrix}$$

Como podemos comprobar en la matriz anterior, las relaciones de preferencia tienen las siguientes propiedades:

- La preferencia de una alternativa frente a ella misma es 0.5.

- Reciprocidad → $P^k_{ij} + P^k_{jj} = 1$.

2.2.2. Modelo de decisión.

Un proceso de TDG se compone básicamente de dos fases:

- Fase de Agregación → En esta fase se combinan las preferencias individuales de los distintos expertos para obtener un valor de preferencia colectiva sobre cada alternativa.

- Fase de Explotación → En esta se aplica un criterio de precedencia que ordena los valores de preferencia colectiva. De esta forma se obtiene la alternativa o conjunto de alternativas solución al problema.

Figura: Fases de agregación y explotación en un problema de TDG.

2.3. Procesos de Alcance del Consenso.

En este punto, veremos en que consiste el PC, veremos cómo se lleva a cabo, las herramientas que comúnmente se suelen emplear en él, y por último veremos las figuras que intervienen en un PC.

2.3.1. Concepto de Consenso.

Como paso previo a la explicación de qué es un Proceso de Consenso, hay que tener claro qué entendemos por el término consenso. Entendemos por consenso como aquel estado de acuerdo producido por consentimiento entre todos los miembros de un grupo o de varios grupos.

En el contexto de nuestro problema, vamos a considerar que se ha alcanzado consenso cuando existe unanimidad de todos los expertos en cuanto a los resultados del PC.

2.3.2. Proceso de Alcance del Consenso.

Como hemos visto anteriormente, un problema de Toma de Decisiones en Grupo se caracteriza por lo siguiente:

- Existe un problema de decisión que puede ser resuelto de diversas maneras.

- Existen dos o más expertos, cada uno con sus ideas, conductas y experiencias que opinan sobre dicho problema.

- Los expertos deben de llegar a una solución común.

El Proceso de Consenso (PC) es uno de los procesos básicos de un problema de Toma de Decisiones, que consiste en que un grupo de expertos discuten acerca de un determinado problema en busca de una solución común o consensuada. De esta manera, el PC se dará por concluido cuando se haya alcanzado un determinado umbral de consenso o se haya sobrepasado un determinado número de rondas máximas. Ambos parámetros son fijados en la definición del problema que lleva a cabo el moderador o el experto líder. Si este umbral es alcanzado antes de consumir el máximo de rondas, el PC se dará por concluido.

En cada una de las rondas que dura el PC, cada uno de los expertos da su opinión sobre el problema. Una vez recogidas todas las opiniones de la ronda actual, se mide el grado de consenso alcanzado en la ronda. Si se ha alcanzado el umbral, el PC se da por concluido. En caso contrario, el PC pasará a la ronda siguiente. Para lograr en posteriores rondas una convergencia de las opiniones de los expertos, el

moderador del problema envía a estos expertos una serie de recomendaciones. Con estas recomendaciones, lo que se persigue es que los expertos más alejados de la opinión colectiva se vayan acercando poco a poco a ella.

Para que el Proceso de Consenso haya terminado satisfactoriamente, se deben cumplir los siguientes requisitos:

- Todos los miembros del grupo de expertos deben de estar de acuerdo con la decisión.

- No puede haber un miembro que se oponga a la decisión.

- Todos los miembros se deben sentir identificados con la decisión tomada.

2.3.2.1. Estructura del Proceso de Consenso.

A la hora de realizar el PC, existen varias metodologías o formas de llevarlo a cabo. Un esquema básico podría ser el siguiente:

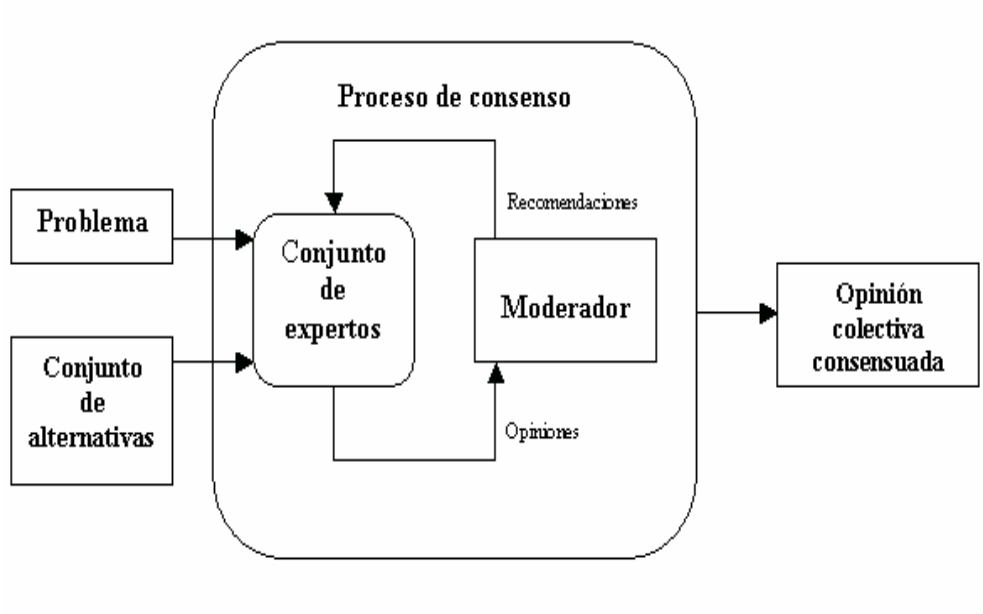


Figura: Estructura del Proceso de Consenso en un problema de TDG.

En el diagrama tenemos los siguientes elementos:

Por un lado tenemos el problema de TDG del que queremos debatir. De este problema necesitamos una descripción detallada, sus parámetros más importantes, etc...

Además del problema de TDG, tenemos el conjunto de alternativas de solución del problema asociadas a dicho problema.

Una vez tenemos determinado el problema y las alternativas de solución, necesitamos el conjunto de expertos que van a debatir sobre el problema. Además de este conjunto de expertos, es necesaria la figura de un moderador que guíe el PC.

El Proceso de Consenso se descompone a su vez en una serie de subprocesos. A continuación describimos estos subprocesos:

1) *Elaborar una descripción formal y detallada del problema*.

   En este primer paso, se define el problema con el mayor detalle posible. Además se eligen a aquellos expertos que se van a encargar de discutir el problema.

2) *Identificar y formalizar las reglas básicas que va a seguir el Proceso de Consenso.*

   A continuación, se definen los parámetros del PC del problema tales como el umbral de consenso, número de rondas máximas, etc..

3) *Identificar el conjunto de alternativas del problema.*

   Se identifica y selecciona el conjunto de posibles alternativas de solución del problema.

4) *Obtener las preferencias de los expertos para la ronda actual.*

   Cada experto envía sus preferencias al moderador para que se pueda calcular el consenso de la ronda.

5) Calcular la opinión colectiva y el grado de consenso alcanzado.

A partir de las preferencias de los expertos, se procesan los datos y se calcula la opinión colectiva, así como el grado de consenso alcanzado en la ronda. Si se ha alcanzado el umbral de consenso o el máximo de rondas, el proceso se acaba. En caso contrario, el PC continúa.

6) Analizar las preferencias individuales de cada miembro.

A partir de las opiniones de los expertos y de la opinión colectiva, se calcula la proximidad de cada experto a la opinión colectiva. En función de esta proximidad, se elabora un conjunto de recomendaciones que son enviadas a los expertos. Estas recomendaciones pueden ser enviadas a todos, a los más alejados, etc...

El PC vuelve al punto 4.

A la hora de calcular el grado de consenso alcanzado en cada una de las etapas del proceso de consenso, hay que calcular la similitud existente entre las opiniones de cada uno de ellos. Existen dos tipos de medidas: el grado de consenso y la medida de proximidad.

El grado de consenso medida es usado para calcular el grado de nivel global de acuerdo existente entre las distintas opiniones de los expertos. Permite también identificar las opiniones de los expertos cuando existe un alto grado de desacuerdo entre ellos.

La otra medida es la de proximidad. Esta medida, mide la distancia existente entre las opiniones de cada uno de lo expertos respecto de la opinión colectiva. Nos permite identificar a aquellos miembros del grupo que están más alejados de los postulados del grupo. De esta manera obtenemos a aquellos expertos cuyas preferencias deben cambiar.

En los problemas de toma de decisiones en grupo, se suelen usar ambas medidas de forma conjunta.

2.3.3. Revisión de modelos clásicos.

A continuación vamos a describir algunos de los modelos clásicos de Procesos de Consenso. Estos modelos clásicos constituyen la base de el modelo que posteriormente describiremos e implementaremos. Vamos a describir resumidamente tres de ellos. En la literatura pueden encontrarse muchos más.

2.3.3.1. El modelo de Bryson y Zadrozny.

El modelo de Bryson y Zadrozny es un ejemplo de modelo sencillo. Las etapas están muy bien diferenciadas. Pese a ello, no se trata de un modelo que profundice demasiado.

Aparece la figura del moderador como guía del proceso, y de los expertos que aportan sus opiniones. No interviene ninguna persona más en el proceso.

Veamos en el siguiente gráfico la estructura del proceso de consenso en el modelo de Bryson y Zadrozny:

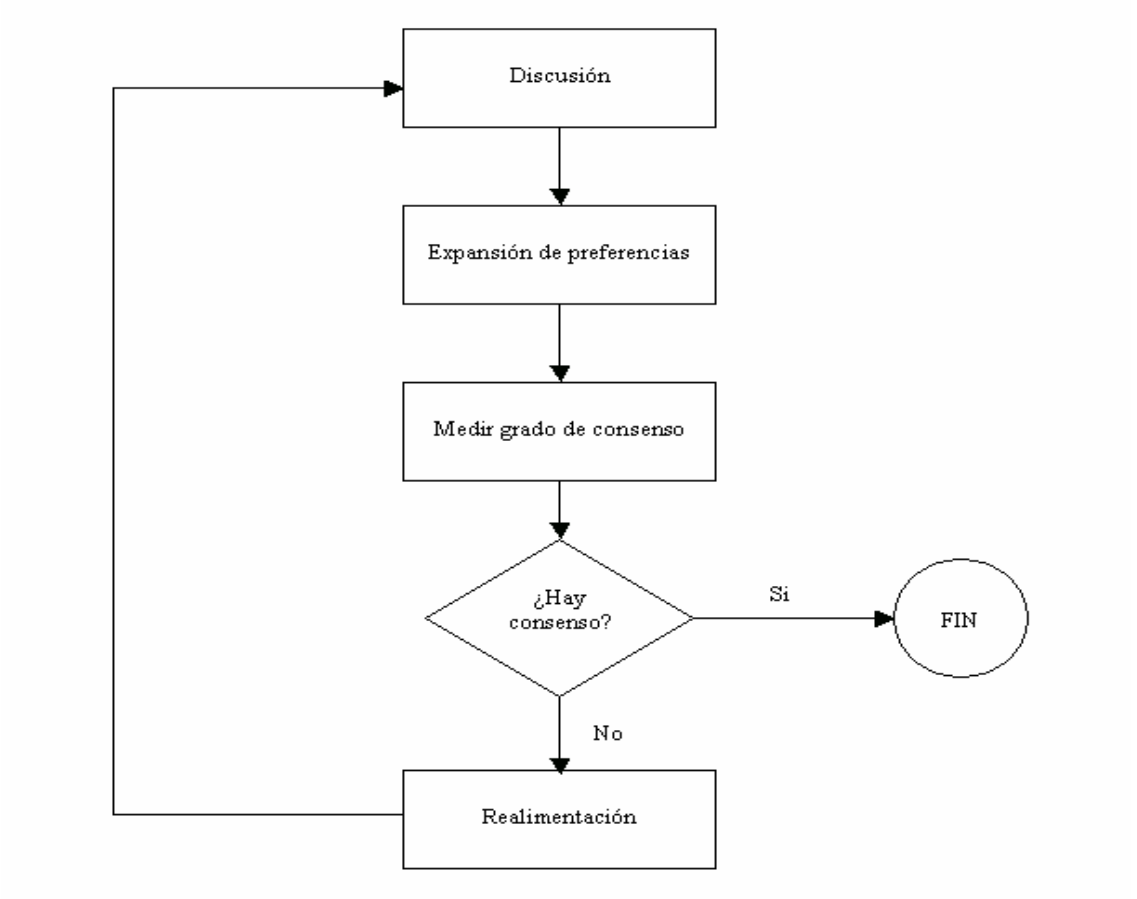


Figura: Modelo de Bryson y Zadrozny.

Discusión → Se exponen las ideas. Se discute sobre las alternativas.

Expansión de preferencias → Cada experto expone sus preferencias entre las distintas alternativas.

Medir grado de consenso → Se ve cuál es la coincidencia entre todos los expertos; se intenta llegar a una decisión final.

¿Hay consenso? → Se comprueba si el consenso alcanzado en la ronda llega al umbral fijado previamente. Si es así, finaliza el proceso.

Realimentación → Al no llegar a consenso, se elaboran y transmiten una serie de recomendaciones a los expertos, para que cambien sus opiniones.

2.3.3.2. El modelo de Saint.

Es un modelo mucho más depurado que el anterior, pero también mucho más complejo. Se suele emplear en un ambiente empresarial. Aparte del moderador y de los expertos, aparecen una serie de actores adicionales:

- *Facilitador* → Es el encargado de poner en marcha todo el proceso de consenso.

- *Redactor del acta* → Se encarga de ir tomando nota de las decisiones que se van alcanzando.

- *Cronometrador* → es el encargado de controlar el tiempo asignado a cada una de las fases de consenso.

- *Monitor* → Es el encargado de controlar, desde el punto de vista emocional todo el proceso de consenso.

En este modelo, el proceso se divide en tres partes: preconsenso, proceso de consenso y cierre del proceso. Veamos en el siguiente gráfico su estructura:

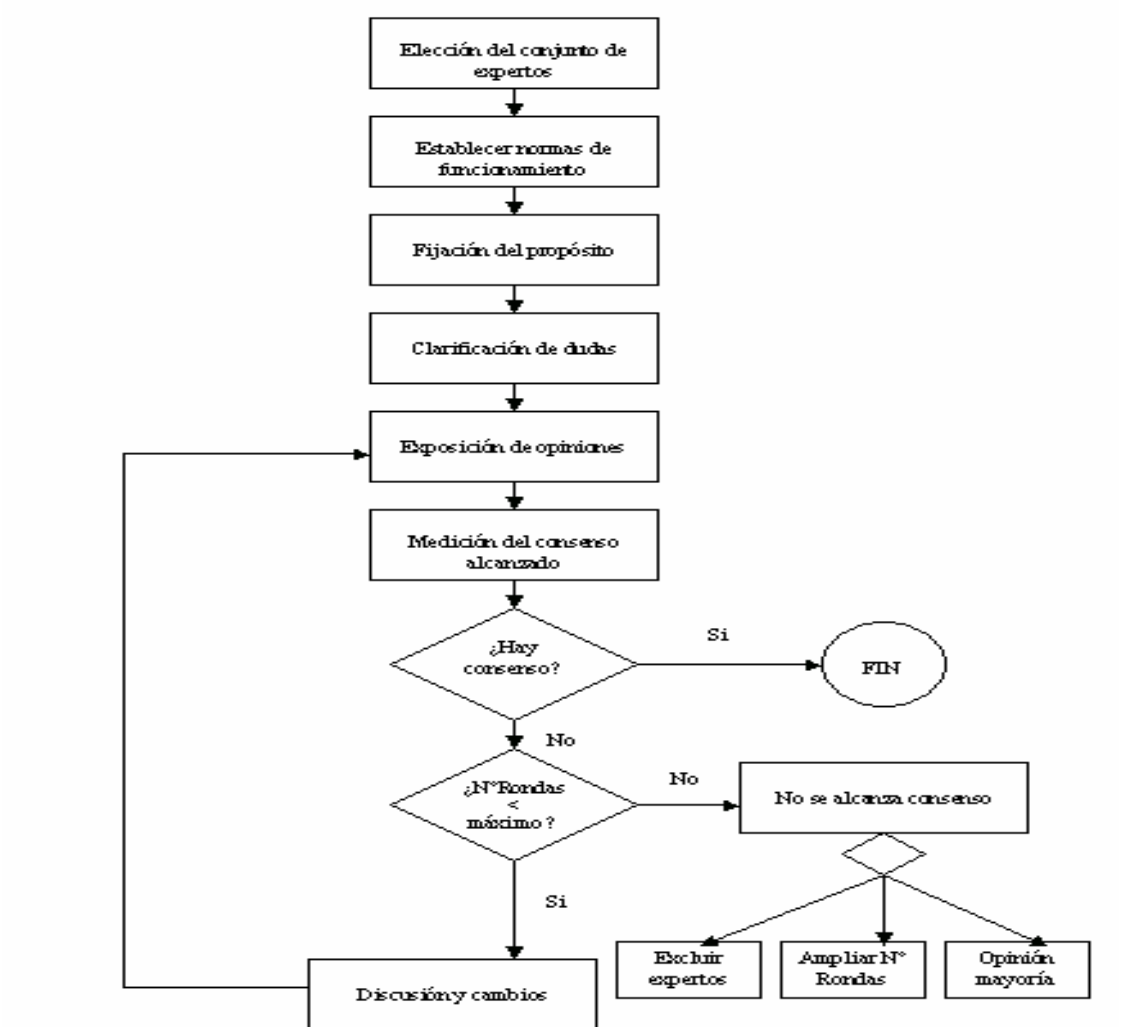


Figura: Modelo de Saint.

En este modelo, además de más actores, incluye una aspecto muy a tener en cuenta. Es el número de rondas. Para evitar que el proceso de consenso se alargue más tiempo del deseado, se puede fijar un umbral o número de rondas máximo del proceso. Si se sobrepasa ese número de rondas, se puede decir que no se ha alcanzado consenso en un tiempo determinado.

Cuando esta circunstancia se produzca en este modelo, las posibles soluciones son varias:

- *Excluir expertos* → Eliminar del grupo a aquellos expertos que estén muy alejados

- *Ampliar número de rondas* → Prolongar el proceso una serie de rondas más.

- *Opinión de la mayoría* → También se puede dar por concluido el proceso de consenso y dar mayor peso a la opinión de la mayoría. Esta solución puede dar lugar a fuertes discrepancias.

2.3.3.3. El modelo de E. Herrera-Viedma, L.. Martínez y F. Mata.

Este modelo ha sido desarrollado por profesores de la Universidad de Jaén y de Granada. Ellos mismos lo han bautizado como:

"*Sistema de apoyo al consenso para problemas de toma de decisiones en grupo en entornos ligüísticos multigranulares.*"

En el siguiente gráfico podemos ver la estructura del modelo:

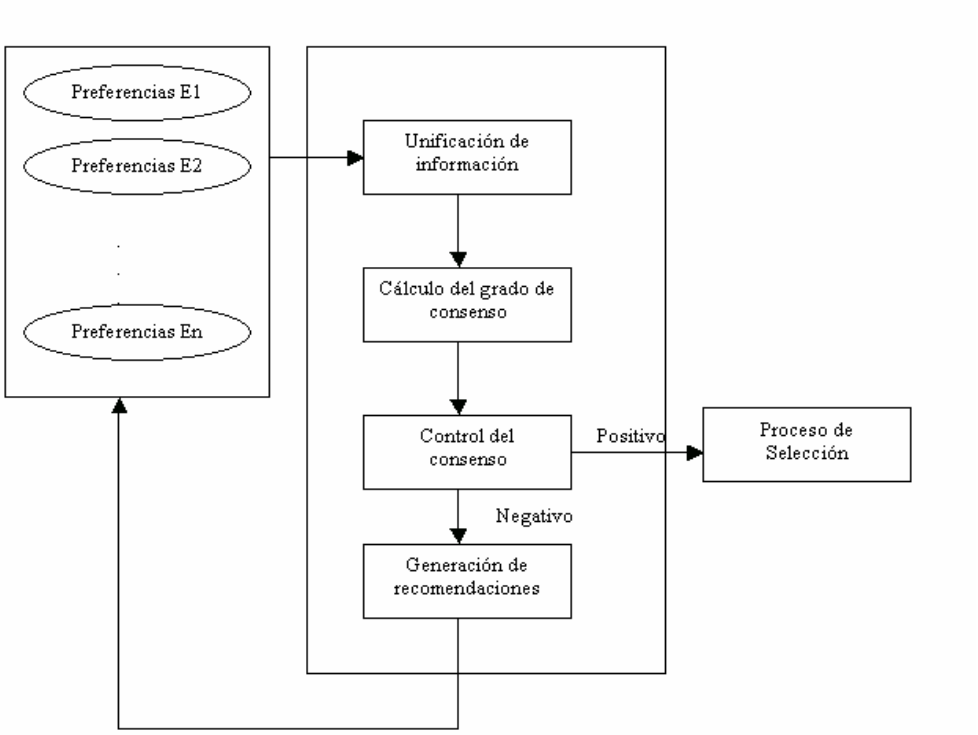


Figura: Modelo de Herrera-Viedma, Martínez y Mata.

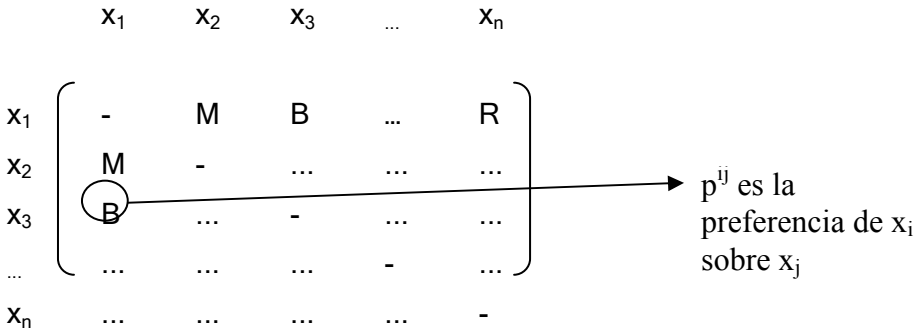
Veamos ahora algunas de las características más importantes de este modelo:

- Utiliza un dominio de valores lingüístico para expresar las opiniones de los expertos. Ej: {Muy Bajo, Bajo, Medio, Alto, Muy Alto}.

- Es multigranular. Cada experto puede utilizar un conjunto de etiquetas lingüísticas diferente para mostrar sus opiniones.

- Los expertos utilizan relaciones de preferencia para expresar sus opiniones:

"Dado un conjunto de expertos, $E = \{e_1, \ldots, e_m\}$, y un conjunto de alternativas, $X = \{x_1, \ldots, x_n\}$ se representa como una matriz:



Donde cada posición **ij** de la matriz representa la preferencia de la alternativa x_i sobre la alternativa x_j ".

Ejemplo: p^{12} indica que se prefiere la alternativa x_2 a x_1, si el valor fuese M (mejor), se dice que x_2 es mejor que x_1.

La estructura del modelo se puede dividir en las siguientes etapas o fases:

- **Unificación de la información** → Se unifica toda la información lingüística multigranular. Para unificar dicha información, se utilizan conjuntos difusos.

- **Cálculo del nivel del consenso** → Se calcula el grado de consenso.

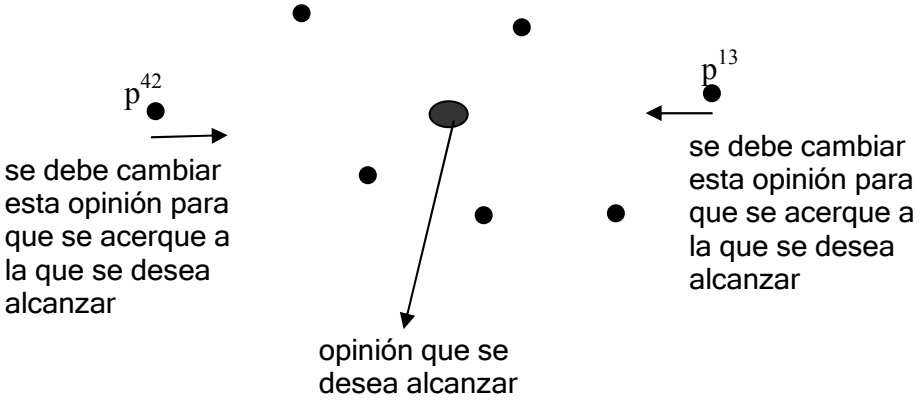
- **Control del grado del consenso** → Si se ha alcanzado el grado de consenso se finaliza el proceso y se pasa al proceso de selección. Si no se alcanza el grado de consenso deseado, se pasa a la generación de recomendaciones.

- **Generación de recomendaciones** → Se realiza para intentar que los expertos cambien de opinión, para ello se utiliza:

- **El cálculo de medidas de proximidad:** consiste en calcular la opinión colectiva de todos los expertos y comprobar la proximidad de cada experto respecto a esa opinión colectiva.

- **Sistema de recomendaciones:** se recomienda a los expertos más alejados que cambien su opinión y en qué dirección deben cambiarla.

Veamos algún ejemplo de manera gráfica:



El objetivo perseguido por este modelo, consiste controlar desde las fases iniciales que todos los expertos cambien de opinión para no imponer la opinión de la mayoría a los que están más alejados de esa mayoría.

2.3.4. Modelo del proyecto.

Una vez hemos comprendido qué es un Modelo de Consenso, y visto sus características y conceptos en los que se basa, vamos a proponer un Modelo de Consenso e implementarlo.

En esta sección, vamos a explicar detalladamente el modelo que vamos a desarrollar. Tras unos planteamientos iniciales en los que explicamos sus características, vamos a explicar cómo vamos a automatizarlo.

2.3.4.1. Modelo inicial.

En este punto describimos las características de la base teórica del Modelo de Consenso que vamos a implementar posteriormente. Este modelo teórico que vamos a describir permite la automatización de las tareas del moderador y facilita la labor de los expertos.

El Modelo de Consenso que proponemos en este PFC tiene las siguientes características:

A) Modelo de Consenso en Grupo.

Interviene un moderador y dos o más expertos. Estos últimos discuten sobre un conjunto de dos o más alternativas.

B) Basado en el Proceso de Consenso.

Está basado en el Proceso de Consenso. No engloba el Proceso de Selección.

C) Basado en el operador de la Distancia Euclídea.

Para el cálculo de la similaridad y la proximidad entre las opiniones de los expertos, se utiliza un operador basado en la Distancia Euclídea.

D) Máximo de rondas y umbral de consenso.

El Proceso de Consenso de este modelo, dura hasta que se alcanza un umbral de consenso o un máximo de rondas. Ambos parámetros son fijados al iniciar el proceso.

E) Numérico.

Las opiniones de los expertos vienen expresadas en valores numéricos normalizados en una escala de 0 a 1. Solo maneja información numérica.

F) Pares de Alternativas.

Las preferencias de los expertos vienen expresadas en pares de alternativas. Es decir, miden la bondad que tiene una alternativa frente a la otra para ese experto.

G) No existe pesos.

Las opiniones de todos los expertos tienen el mismo valor y repercusión en el proceso. No hay expertos cuya opinión sea más importante que el resto.

Por esa razón, no se asignan pesos a las opiniones de los expertos. O todos tienen el mismo peso.

En el siguiente gráfico podemos ver un esquema del Modelo.

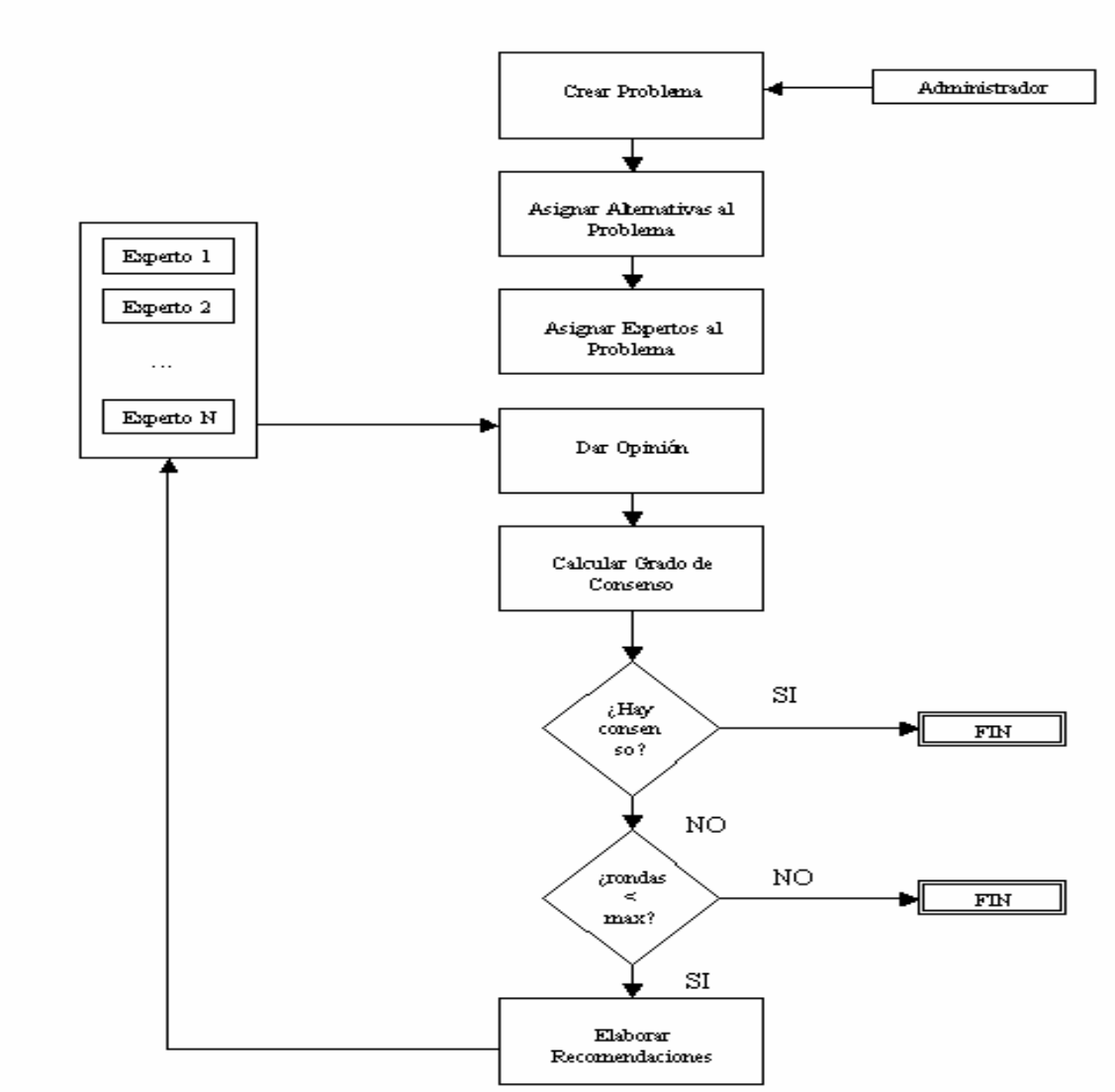


Figura: Esquema del Modelo de Consenso Propuesto.

2.3.4.2. Automatización.

El modelo arriba descrito va a ser implementado respetando su estructura, pero minimizando el papel tanto del administrador como de los expertos. Todas las características expuestas en el punto anterior están presentes en el modelo de consenso implementado.

En él va a existir un usuario administrador, que crea el problema con todos sus parámetros (umbral máximo y número de rondas máximo), alternativas y le asigna un conjunto de expertos a él. También puede consultar cuál es el estado del problema en cualquier instante.

El papel de los expertos se reduce a introducir sus opiniones en cada una de las rondas que dure el proceso. En cada ronda éste recibirá las recomendaciones de la anterior.

Una vez dadas todas las opiniones de una ronda para el problema, el sistema obtiene los datos suministrados por los expertos (almacenados en una Base de Datos), los procesa y elabora las recomendaciones para la ronda posterior (en caso de que no se haya llegado al final).

Una vez que un experto ha dado su opinión del problema para una determinada ronda, deberá esperar a que el resto de los expertos den sus opiniones para dicha ronda.

Mediante un sistema de validación, el sistema evitará que un experto pueda modificar las opiniones de otros, o dar opinión para un problema al que no está asignado.

Algunas de las características más importantes de este modelo automatizado son:

A) Centralizado.

Los datos del problema tales como: rondas máximas, rondas consumidas, umbral de consenso, alternativas del problema, expertos asignados, así como las

opiniones de los expertos en cada una de las rondas, son almacenados en una Base de Datos centralizada y protegida.

Cuando un experto da una opinión válida para un problema que tenga asignado, el sistema almacenará estos datos en la Base de Datos.

Cuando todas las opiniones de una ronda para un determinado problema estén dadas, el sistema accede a las opiniones de todos los expertos involucrados, procesa los datos y los almacena en la Base de Datos.

B) Cliente - Servidor.

El modelo está basado en la arquitectura Cliente-Servidor. De esta manera a través del protocolo TCP/IP, los expertos podrán enviar sus preferencias. Cada experto podrá interactuar con el sistema desde su propia máquina en un entorno LAN o WAN.

C) Basado en interfaz web.

La interacción de los usuarios (expertos y administradores), se realiza a través de un interfaz web. Los datos introducidos por los usuarios son enviados al servidor mediante protocolo http.

Toda la gestión del problema se lleva a cabo mediante esta interfaz web. Desde la creación del problema, la consulta de su estado, el envío de las preferencias de los expertos, etc...

D) Sin límite de expertos y alternativas.

En el proceso de consenso implementado, pueden intervenir tantos expertos como se desee. La única limitación la pone el número de expertos registrados en la Base de Datos.

De igual manera, no se ha definido un máximo de problemas soportados por el sistema. Cada experto puede formar parte de tantos problemas como así lo deseen los administradores del sistema.

Por último, de nuevo no existe limitación en lo referente al número de alternativas por problema. El administrador del sistema cuando crea el problema, puede crear para él tantas alternativas como desee.

CAPITULO 3:
PROYECTO FIN DE CARRERA.

3.1. Sistema de Apoyo al Consenso.

Una vez presentado el objetivo y los propósitos de nuestro modelo de consenso, vamos a presentar en detalle el desarrollo de nuestro prototipo de modelo basado en la arquitectura cliente-servidor.

Por lo tanto, esta segunda parte es un proyecto de desarrollo software y, como tal, para su desarrollo deben seguirse las actividades de la Ingeniería del Software. A la hora de definir el concepto de Ingeniería del Software, nos damos cuenta de que no existe una definición única o mejor que el resto. Algunas de las más utilizadas son las siguientes:

- La Ingeniería del Software es la construcción de software de calidad con un presupuesto limitado y un plazo de entrega en contextos de cambio contínuo.

- La Ingeniería del Software es el establecimiento y uso de una serie de principios y métodos firmes de ingeniería para obtener software económico, de calidad, que sea fiable y que funciones de manera eficiente en máquinas reales.

Las actividades básicas de la Ingeniería del Software son las siguientes:

A) Especificación de Requerimientos → En esta primera etapa se obtienen el propósito del sistema y las propiedades y restricciones del mismo.

B) Análisis del Sistema → En la segunda fase de la Ingeniería del Software se obtiene un modelo del sistema correcto, completo, claro, consistente y verificable.

C) Diseño del Sistema → En esta tercera fase de la Ingeniería del Software, se definen los objetivos del proyecto y las estrategias a seguir para conseguirlos.

D) Implementación → En esta fase se traduce el modelo anterior a código fuente usando algún lenguaje de programación.

3.2. Especificación de Requerimientos.

Los requerimientos de un sistema o producto software son el conjunto de propiedades o restricciones definidas con total precisión, que el sistema o producto debe satisfacer. Los requerimientos de un sistema pueden ser de dos tipos:

- *Requerimientos funcionales* → Son aquellos que se refieren específicamente al funcionamiento del producto software.

- *Requerimientos no funcionales* → Son aquellos que no están referidos al funcionamiento estricto del producto software sino a una serie de factores externos.

En los siguientes subapartados, vamos a definir estos requerimientos (tanto funcionales como no funcionales) para el Modelo de Consenso que vamos a implementar.

3.2.1. Requerimientos funcionales.

Los requerimientos funcionales de un sistema software son aquellos que se encargan de describir las funcionalidades que el sistema debe proporcionar a sus usuarios. Estos requerimientos son obtenidos mediante la interacción y reunión con el cliente del producto. Para ello se suelen encargar encuestas o entrevistas para definirlos. En nuestro caso, y debido a la inexistencia de un cliente de nuestra aplicación, he pedido la colaboración de algunos de mis compañeros del trabajo para poder definirlos.

Los requerimientos funcionales de nuestro Modelo de Consenso a implementar será los siguientes:

A) Que en el sistema puedan convivir varios problemas en curso.

B) Que un experto del sistema pueda participar en más de un problema.

C) Que en un problema puedan participar simultáneamente tantos expertos como sean asignados.

D) Que pueda existir un administrador del problema con permiso para crear nuevos problemas.

E) Que un administrador pueda consultar tanto el estado del problema, parámetros de este, así como su historial en cualquier momento.

F) Que solo puedan participar en los problemas usuarios que están registrados y asignados por el administrador del problema.

G) Que se puedan introducir nuevos usuarios (expertos y administradores) en cualquier momento al sistema.

H) Que los expertos puedan consultar la lista de los problemas en los que participa y ha participado.

I) Que los expertos puedan enviar sus opiniones concurrentemente.

J) Que los expertos puedan disponer de las recomendaciones antes de enviar sus preferencias.

3.2.2. Requerimientos no funcionales.

Los requerimientos no funcionales son aquellos que restringen los requerimientos funcionales. A menudo estos requerimientos no funcionales suelen ser críticos para la aceptación del sistema. Los requerimientos no funcionales frecuentemente sirven para especificar propiedades del sistema o del producto en sí, tales como plataforma en la que se asienta, velocidad mínima, memoria necesaria, etc...

En nuestro PFC a la hora de referirnos a los requerimientos no funcionales, estaremos hablando de los recursos hardware y software necesarios para que los requerimientos funcionales anteriores puedan ser cumplidos.

3.2.2.1. Requerimientos Hardware.

Los requerimientos hardware de nuestro Modelo de Consenso, los hemos agrupado en dos grupos. Uno de ellos hace referencia a los requerimientos software en el desarrollo de la aplicación. El otro grupo hace referencia a los requerimientos hardware para su uso.

A) Desarrollo.

Velocidad → Necesitamos un procesador que permita compilar y construir proyectos Java en un tiempo razonable. Con un procesador a 1000 MHz es suficiente.

Memoria → La máquina virtual de java (JVM) tiene muchos procesos y recursos. Utilidades como el recolector de basuras consumen bastante memoria. Para poder manejar el entorno Netbeans cómodamente junto a la JVM y la Base de Datos vamos a necesitar al menos 256 MB de memoria RAM.

Entorno de Red → Para poder probar la aplicación en un entorno web, necesitamos un entorno de red local (LAN) y acceso a Internet basado en TCP/IP (Ethernet).

B) Uso.

Velocidad → Para dar soporte a los distintos usuarios del sistema, necesitamos un servidor con un procesador rápido, que pueda atender todas las peticiones que se vayan produciendo en cada instante. Para cada uno de los clientes, no son requeridos equipos dotados de procesadores rápidos, pues el procesamiento se produce en el servidor. Las máquinas de los clientes (expertos) únicamente muestran y envían datos en formularios HTML.

Memoria → El servidor necesita al menos 512 MB de memoria RAM para poder manejar con soltura los accesos a Base de Datos que se produzcan, la JVM y las peticiones que le llegan de los clientes.

Entorno de Red → No es necesario Internet para poder usar la aplicación. Al menos hará falta tener una Red de Área Local (LAN) que tenga una máquina cliente y otra servidor.

3.2.2.2. Requerimientos Software.

De nuevo, y al igual que hicimos en el apartado anterior, vamos a describir los requerimientos software tanto para el desarrollo como en el manejo de la aplicación.

A) Desarrollo.

Sistema Operativo (SS.OO) → Al ser el lenguaje de programación empleado multiplataforma, y las herramientas de libre distribución, podemos empelar tanto Microsoft Windows 2000/XP como cualquier distribución de Linux.

Entorno de desarrollo (IDE) → Netbeans 5.0. También podemos usar otro entorno muy recomendable como es el Eclipse.

Servidor de Aplicaciones → Servidor Apache Tomcat 5.5 instalado en la máquina servidor. Los clientes no necesitan tener instalado este programa.

Base de Datos → MySQL Query Browser versión 1.1.18 (para la monitorización) y MySQL Administrator versión 1.1.6 (para el manejo y manipulación de tablas). Esta aplicación deberá estar instalada en el servidor. Únicamente es usado por el administrador de la Base de Datos.

Diseño de diagramas → Visual Paradigm para diseñar diagramas UML y Casos de Uso.

Navegación Web → Microsoft Internet Explorer o Mozilla Firefox.

Documentación → Suite Microsoft Office y/o OpenOffice.

Programación → Jdk Java 1.5 J2EE (Java 2 Enterprise Edition).

B) Uso

Sistema Operativo (SS.OO) → Microsoft Windows 2000/XP o cualquier distribución de Linux.

Servidor de Aplicaciones → Servidor Apache Tomcat 5.5 instalado en el servidor.

Base de Datos → MySQL Query Browser versión 1.1.18 (para la monitorización) y MySQL Administrator versión 1.1.6 (para el manejo y manipulación de tablas). Ambas en el servidor. En el cliente no es necesario tener instalado un cliente de BBDD, ya que el acceso lo realiza el servlet.

Navegación Web → Para la visualización de las ventanas de la aplicación, se recomienda Mozilla Firefox.

3.3. Análisis del Sistema.

Una vez conocidos tanto el propósito del proyecto, como las propiedades y restricciones que deberá cumplir este sistema, llega el momento de analizar el sistema y crear el Modelo de Consenso que queremos implementar. Para que el modelo que vamos a implementar se correcto, completo, consistente y ajustado a los requerimientos, vamos a describir los diagramas de casos de uso necesarios.

3.3.1. Casos de uso.

Los diagramas de casos de uso, son la representación de una situación o tarea de interacción de un usuario con la aplicación. Viene dado como un flujo de eventos. Describen en cierta manera cómo se realiza una tarea de manera exacta. Sus elementos son:

- *Identificador único* → Cada diagrama de casos de uso del sistema tiene un nombre o identificador único que lo distingue de los demás.

- *Actores* → Son aquellas personas o entes que intervienen en el proceso para dicho caso de uso. Cada uno tiene también un identificador único para distinguirlo del resto.

- *Flujo de eventos* → Es la secuencia de acciones que forman la funcionalidad del caso de uso.

- *Condiciones de entrada y salida* → Son los datos y/o condiciones de entrada y de salida del sistema.

- *Requerimientos especiales* → Otro tipo de elementos.

En nuestro sistema, hemos identificado los siguientes actores:

- *Experto* → Es el usuario de la aplicación que va a enviar sus opiniones acerca de los problemas en los que ha sido asignado. En el sistema van a existir muchos expertos.

- *Administrador del Problema* → Es aquella persona que crea los problemas, asigna recursos a ellos y comprueba su estado.

- *Administrador de la BBDD* → Es aquel usuario del sistema que interactúa de manera directa con la BBDD. Es el gestor de la BBDD. Tiene permisos para crear/modificar datos de la aplicación.

- *BBDD* → Es la base de datos. Proporciona y almacena los datos de la aplicación.

Una vez identificados tanto los actores como las funcionalidades del sistema, podemos ponernos ya a crear los casos de uso. Estos casos de uso deben de englobar todos los requerimientos funcionales que describimos anteriormente.

El diagrama de caso de uso principal de nuestro sistema lo vamos a llamar Sistema de Consenso Automatizado. Este diagrama principal del sistema es conocido con el nombre de diagrama frontera. Se le llama así porque describe completamente la funcionalidad de un sistema.

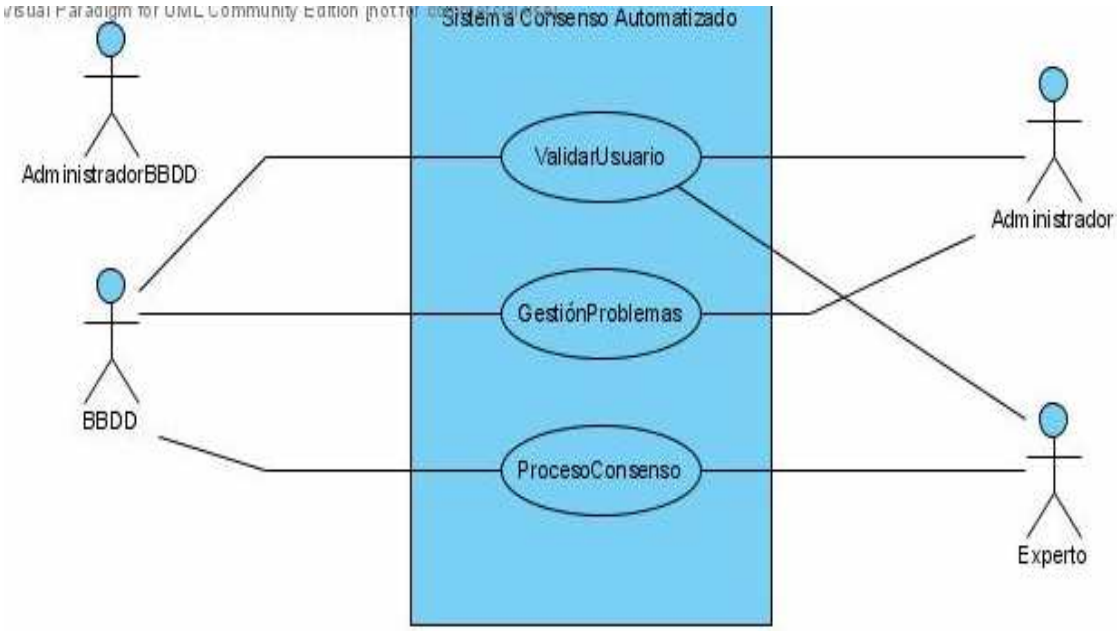


Figura: Diagrama frontera Sistema de Consenso Automatizado.

En el diagrama, el actor Administrador BBDD es aquel que puede crear, modificar o eliminar datos referentes a los problemas o los expertos. Estas

operaciones, las realiza Administrador BBDD desde el cliente de la Base de Datos. No las realiza a través de la aplicación. Por ello aparece en el diagrama frontera sin relación con ninguno de los casos de prueba posteriores.

Una vez descrito el diagrama frontera, es el turno de los diagramas de casos de uso. En éstos pueden aparecer dos tipos de relaciones:

- <<extend>> → Es una relación que nos permite representar en un diagrama de casos de uso comportamientos excepcionales de él.

- <<include>> → Es una relación que representa un comportamiento común.

En nuestro caso, vamos a describir más detalladamente los casos de uso *GestiónProblemas* y *ProcesoConsenso*:

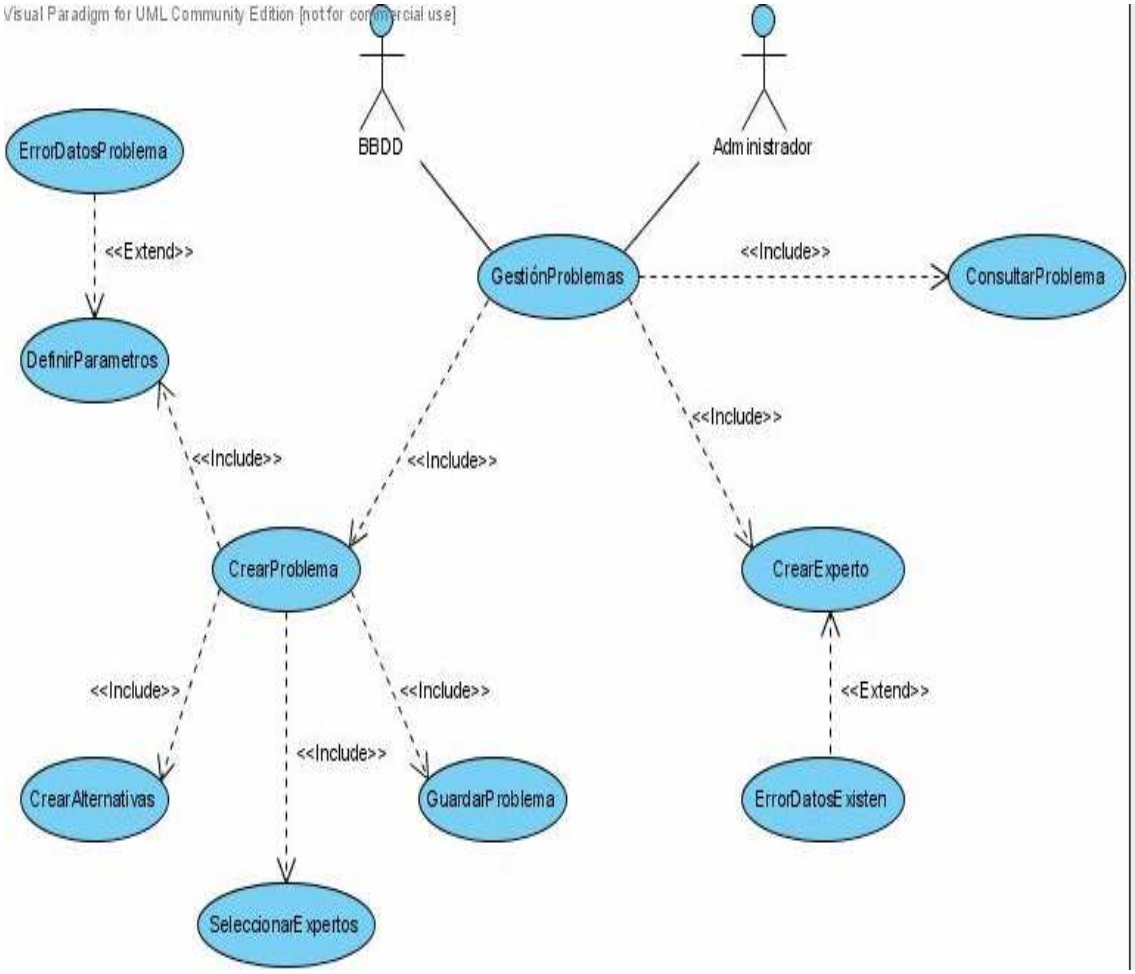


Figura: Diagrama del caso de uso GestiónProblemas.

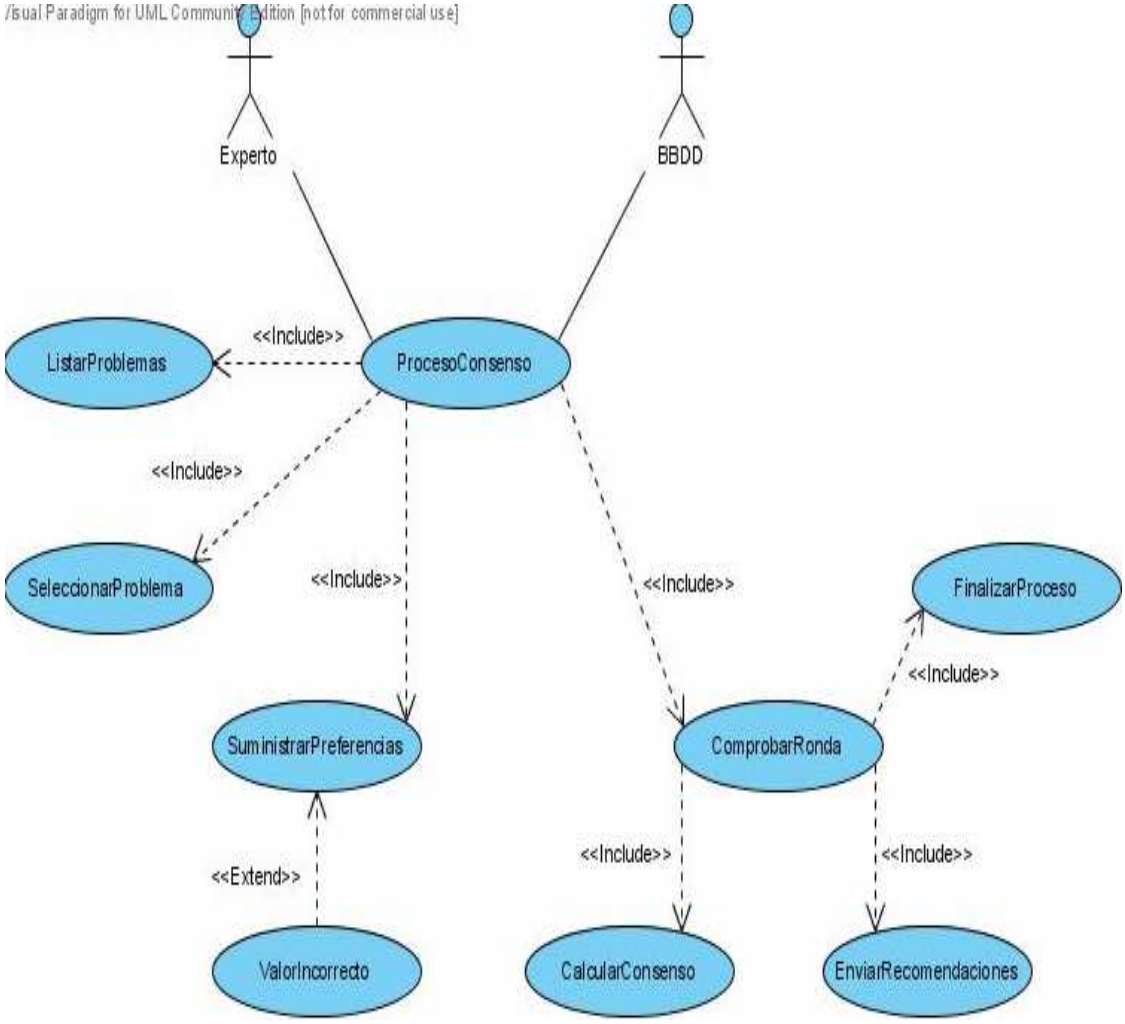


Figura: Diagrama del caso de uso ProcesoConsenso.

Caso de Uso 1: *ValidarUsuario*.

Actores participantes: *Experto*, *Administrador* y *BBDD*.

Condiciones de entrada: Que existan usuarios dados de alta en el sistema.

Flujo de eventos:

1. El usuario (*Experto* o *Administrador*) abre la ventana de entrada al sistema para expertos o administradores según su rol.
2. El sistema muestra un formulario de entrada al mismo.
3. El usuario introduce su nombre de usuario y contraseña del sistema.
4. El sistema comprueba en la *BBDD* que los datos son correctos.
5. Si el usuario es:

- *Administrador* → El sistema mostrará el panel de control del administrador.
- *Experto* → El sistema mostrará el listado de problemas asignados al experto.

Condiciones de salida: El usuario (*Experto* o *Administrador*) ha sido autentificado en el sistema.

Excepciones:

E-1 → El identificador introducido por el usuario no es válido. El sistema informa al usuario del error y vuelve a mostrar la ventana de entrada al sistema. El usuario puede decidir intentarlo de nuevo o cerrar la ventana.

Caso de Uso 2: *GestionProblemas*.

Actores participantes: *Administrador* y *BBDD*.

Condiciones de entrada: El *Administrador* está identificado correctamente, y la *BBDD* está operativa.

Flujo de eventos:

1. El sistema muestra el menú principal del *Administrador* con tres opciones, y le pide al usuario que elija una:
   - Si el *Administrador* elige la opción Introducir nuevo problema de TDG, se realiza S-1.
   - Si el *Administrador* elige la opción Introducir nuevo experto al sistema, se realiza S-2.
   - Si el *Administrador* elige la opción Consultar problema, se realiza S-3.

Subflujos de eventos:

S-1: CrearProblema

1. El *Administrador* introduce el número de alternativas y de expertos que tendrá el problema a crear, y pulsará Introducir.
2. El sistema muestra el menú principal de crear nuevo problema.
3. El *Administrador* define todos los parámetros que se le indica y realiza una breve descripción del mismo.
4. El *Administrador* define las alternativas del problema.
5. El *Administrador* selecciona los expertos que formarán parte de la discusión del problema.
6. El *Administrador* pulsa el botón Enviar, y los datos son almacenados en la *BBDD*.

S-2: CrearExperto

1. El *Administrador* pulsa el botón Introducir de la opción número 2.
2. El sistema muestra una ventana principal para introducir nuevo experto al sistema.
3. El *Administrador* introduce el identificador del nuevo usuario, la contraseña y la confirmación de la contraseña.
4. El *Administrador* pulsa el botón Enviar, y los datos son almacenados en la *BBDD*.

S-3: ConsultarProblema

1. El *Administrador* selecciona un problema de la lista en el menú principal de administrador, y pulsa el botón Consultar.
2. El sistema muestra al *Administrador* la información del problema seleccionado.

Condiciones de salida: La acción elegida se ha llevado a cabo.

Excepciones:

E-1 → El número de alternativas y/o expertos introducidos en CrearProblema no es un número entero positivo.

E-2 → El experto a crear en CrearExperto, ya existe en la BBDD.

Caso de Uso 3: *ProcesoConsenso.*

Actores participantes: *Experto* y *BBDD*.

Condiciones de entrada: El *Experto* está identificado correctamente, y la *BBDD* está operativa.

Flujo de eventos:

1. El sistema muestra al *Experto* la lista de problemas en los que está asignado. Estos pueden mostrar tres estados.
   - Si está esperando la opinión del *Experto*, mostrará el estado Dar Opinión.
   - Si está esperando la opinión de otros expertos, mostrará el estado Esperando Expertos.
   - Si el problema ya ha terminado, mostrará el estado Finalizado.
2. El *Experto* selecciona un problema cuyo estado sea Dar Opinión. Pulsa el botón Dar Opinión.
3. El sistema muestra la ventana principal del problema.
4. Se realiza S-1
5. El sistema comprueba si dispone de las preferencias de todos los expertos.
   - Si tiene todas las preferencias de los expertos del problema, se realiza S-2.
   - Si faltan las preferencias de algún experto, no realiza nada.

Subflujo de eventos:

S-1: Suministrar Preferencias

1. El Experto rellena cada una de las casillas de la matriz de preferencias que aparece en la ventana del problema.
2. El *Experto* pulsa el botón Enviar, y la *BBDD* almacena los datos.

S-2: ComprobarRonda

1. El sistema accede a la *BBDD* a por las preferencias de todos los expertos del problema.
2. El sistema calcula el grado de consenso y la proximidad y lo almacena en la *BBDD*.
3. El sistema comprueba si se ha alcanzado el consenso.
   - Si se ha alcanzado el consenso, finaliza el proceso de consenso.
   - Si no se ha alcanzado el consenso, envía las recomendaciones a los expertos.

Condiciones de salida: El *Experto* ha dado su opinión del problema para la ronda.

Excepciones:

E-1 → Los valores de las preferencias suministradas por el *Experto* no son correctos. Debe ser un número real positivo comprendido entre 0 y 1.

3.3.2. Escenarios.

Un caso de uso es una representación abstracta, una abstracción, de una funcionalidad del sistema a realizar. La representación concreta de un caso de uso se realiza mediante la creación de uno o más escenarios que muestren todas las interacciones posibles entre el sistema y sus usuarios.

Un escenario está formado por los siguientes elementos:

- Un nombre único y unívoco.
- Una descripción.
- Los actores participantes.
- El flujo de eventos.

Como se ha indicado antes, para cada caso de uso puede haber varios escenarios. Para nuestro proyecto, se han creado y descrito una cantidad importante de casos de uso, por lo que solo vamos a definir algunos escenarios que puedan servir como ejemplo de las principales funcionalidades que el sistema va a desarrollar: dar opinión de un problema y crear un nuevo problema.

A) Escenario DarOpinionProblema1.

Nombre: DarOpinionProblema1.

Descripción: El usuario con identificador pedro quiere mandar sus preferencias sobre el problema cuyo identificador es 1.

Actores: Experto cuyo identificador es pedro y BBDD.

Flujo de eventos:

1. El usuario entra al sistema. Para ello ejecuta el fichero *index_exp.html*.
2. El sistema muestra el formulario de entrada.
3. El usuario introduce el identificador pedro y la contraseña.
4. El sistema valida que los datos son correctos y el usuario entra en la aplicación como pedro.
5. El sistema muestra el menú principal del experto cuyo identificador es pedro.
6. El sistema se comunica con BBDD y obtiene el listado de problemas asignados al usuario cuyo identificador es pedro.
7. El usuario selecciona el problema cuyo identificador es 1 y pulsa el botón Dar Opinión.
8. El sistema muestra un formulario al usuario para que introduzca su opinión del problema cuyo identificador es 1.
9. El usuario introduce su opinión en la matriz de opinión y pulsa el botón enviar.
10. El sistema valida los datos suministrador por el usuario.
11. El sistema envía los datos a BBDD para que actualice la información.
12. El sistema informa al usuario de que la operación ha finalizado con éxito.

B) Escenario CrearProblema7.

Nombre: CrearProblema7.

Descripción: El usuario con permisos de administrador cuyo identificador es paco, crea el problema 7.

Actores: Administrador cuyo identificador es paco y BBDD.

Flujo de eventos:

1. El usuario entra al sistema. Para ello ejecuta el fichero *index_adm.html*.

2. El sistema muestra el formulario de entrada para el usuario administrador paco.
3. El usuario introduce el identificador paco y la contraseña.
4. El sistema valida que los datos son correctos y el usuario entra en la aplicación como paco con permisos de administrador.
5. El sistema muestra el menú principal del administrador cuyo identificador es paco. Este menú tiene tres opciones: crear nuevo problema, crear nuevo experto y consultar estado del problema.
6. El usuario selecciona la opción Introducir nuevo problema de TDG. Para ello introduce en la casilla de expertos y alternativas el numero de expertos y de alternativas que tendrá el problema.
7. El usuario pulsa el botón Introducir de la opción 1.
8. El sistema comprueba que el número de expertos y alternativas es correcto.
9. El sistema muestra al usuario la pantalla principal de introducir nuevo problema de TDG.
10. El usuario rellena los datos del problema 7.
11. El usuario especifica el nombre de las alternativas del problema.
12. El usuario selecciona el conjunto de expertos a partir de la lista suministrada.
13. El usuario pulsa el botón Enviar.
14. El sistema comprueba que todos los datos han sido introducidos y son correctos.
15. El sistema envía los datos a BBDD para que actualice la base de datos de problemas.
16. El sistema informa al usuario paco de que el problema ha seido creado satisfactoriamente.

3.4. Diseño del Sistema.

Una vez realizado el análisis de nuestro sistema, la siguiente actividad del proceso de la Ingeniería del Software es el Diseño del Sistema. Este Diseño del Sistema es la actividad más delicada y laboriosa de la Ingeniería del Software.

Decimos que es la actividad más delicada de llevar a cabo, porque es la antesala de la implementación. Si en la fase de Diseño del Sistema se cometen errores, la implementación los reflejará más agudamente, y volver a rediseñar conlleva un gran coste. Por tanto, tan importante es realizar un buen análisis del Sistema, como que el Diseño se adapte perfectamente al Análisis. De esta manera la implementación será más corta y menos costosa.

Podemos definir por tanto al Diseño del Sistema como la actividad de la Ingeniería del Software en la que se identifican los objetivos finales del sistema, y se fijan las distintas estrategias para alcanzarlos en la fase de implementación.

El Diseño del Sistema lo hemos estructurado en dos fases. Estas dos fases son:

- Diseño de Datos → Describimos las estructuras de datos empleadas en la aplicación.

- Diseño de Interfaz → Describimos las interfaces gráficas con las que el usuario interactúa con la aplicación.

3.4.1. Diagramas de clases.

En esta sección vamos a describir el diagrama de clases UML de nuestra aplicación. Los diagramas de clases muestran las diferentes clases que componen un sistema y cómo se relacionan unas con otras. Se dice que los diagramas de clases son diagramas "estáticos" porque muestran las clases, junto con sus métodos y atributos, así como las relaciones estáticas entre ellas: qué clases "conocen" a qué otras clases o qué clases "son parte" de otras clases, pero no muestran los métodos mediante los que se invocan entre ellas.

En un diagrama de clases intervienen los siguientes elementos:

- Clases
- Relaciones.

Clases

Una clase define los atributos y los métodos de una serie de objetos. Todos los objetos de esta clase (instancias de esa clase) tienen el mismo comportamiento y el mismo conjunto de atributos (cada objetos tiene el suyo propio). En ocasiones se utiliza el término "tipo" en lugar de clase.

Las clases están representadas por rectángulos, con el nombre de la clase, y también pueden mostrar atributos y operaciones de la clase.

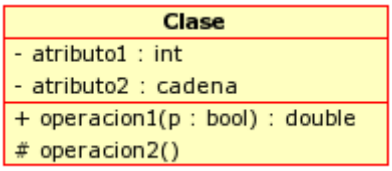


Figura: Ejemplo de clase.

En UML, los atributos se muestran al menos con su nombre, y también pueden mostrar su tipo, valor inicial y otras propiedades. Los atributos también pueden ser mostrados visualmente:

- + Indica atributos *públicos*
- # Indica atributos *protegidos*
- - Indica atributos *privados*

La operaciones o métodos, también se muestran al menos con su nombre, y pueden mostrar sus parámetros y valores de retorno. Las operaciones, al igual que los atributos, se pueden mostrar visualmente:

- + Indica operaciones *públicas*
- # Indica operaciones *protegidas*
- - Indica operaciones *privadas*

Asociaciones

En un diagrama de clases, las asociaciones entre clases representan las dependencias existentes entre ellas. Estas pueden ser de cuatro tipos:

A) Generalización.

La herencia es uno de los conceptos fundamentales de la programación orientada a objetos, en la que una clase “recoge” todos los atributos y operaciones de la clase de la que es heredera, y puede alterar/modificar algunos de ellos, así como añadir más atributos y operaciones propias.

En UML, una asociación de *generalización* entre dos clases, coloca a estas en una jerarquía que representa el concepto de herencia de una clase derivada de la clase base. En UML, las generalizaciones se representan por medio de una línea que conecta las dos clases, con una flecha en el lado de la clase base.

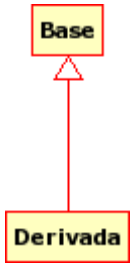


Figura: Representación visual de una generalización en UML

B) Asociación.

Una asociación representa una relación entre clases, y aporta la semántica común y la estructura de muchos tipos de “conexiones” entre objetos.

Las asociaciones son los mecanismos que permite a los objetos comunicarse entre sí. Describe la conexión entre diferentes clases (la conexión entre los objetos reales se denomina conexión de objetos o *enlace*).

Las asociaciones pueden tener una papel que especifica el propósito de la asociación y pueden ser unidireccionales o bidireccionales (indicando si los dos objetos participantes en la relación pueden intercambiar mensajes entre sí, o es únicamente uno de ellos el que recibe información del otro). Cada extremo de la asociación también tiene un valor de multiplicidad, que indica cuántos objetos de ese lado de la asociación están relacionados con un objeto del extremo contrario.

En UML, las asociaciones se representan por medio de líneas que conectan las clases participantes en la relación, y también pueden mostrar el papel y la multiplicidad de cada uno de los participantes. La multiplicidad se muestra como un rango [mín...máx] de valores no negativos, con un asterisco (*) representando el infinito en el lado máximo.

Figura: Representación visual de una asociación en UML

C) Acumulación.

Las acumulaciones son tipos especiales de asociaciones en las que las dos clases participantes no tienen un estado igual, pero constituyen una relación “completa”. Una acumulación describe cómo se compone la clase que asume el rol

completo de otras clases que se encargan de las partes. En las acumulaciones, la clase que actúa como completa, tiene una multiplicidad de uno.

En UML, las acumulaciones están representadas por una asociación que muestra un rombo en uno de los lados de la clase completa.

Figura: Representación visual de una relación de acumulación en UML

D) Composición.

Las composiciones son asociaciones que representan acumulaciones *muy fuertes*. Esto significa que las composiciones también forman relaciones completas, pero dichas relaciones son tan fuertes que las partes no pueden existir por sí mismas. Únicamente existen como parte del conjunto, y si este es destruido las partes también lo son.

En UML, las composiciones están representadas por un rombo sólido al lado del conjunto.

Figura: Representación visual de una relación de acumulación en UML

Debido a la complejidad del diagrama de clases de nuestra aplicación, éste lo vamos a mostrar desglosado según la funcionalidad. Estos son los diagramas de clase:

ServletAdministrador

Este Servlet monitoriza la entrada al sistema del Administrador, así como gestiona el menú principal del Administrador que acaba de autentificarse.

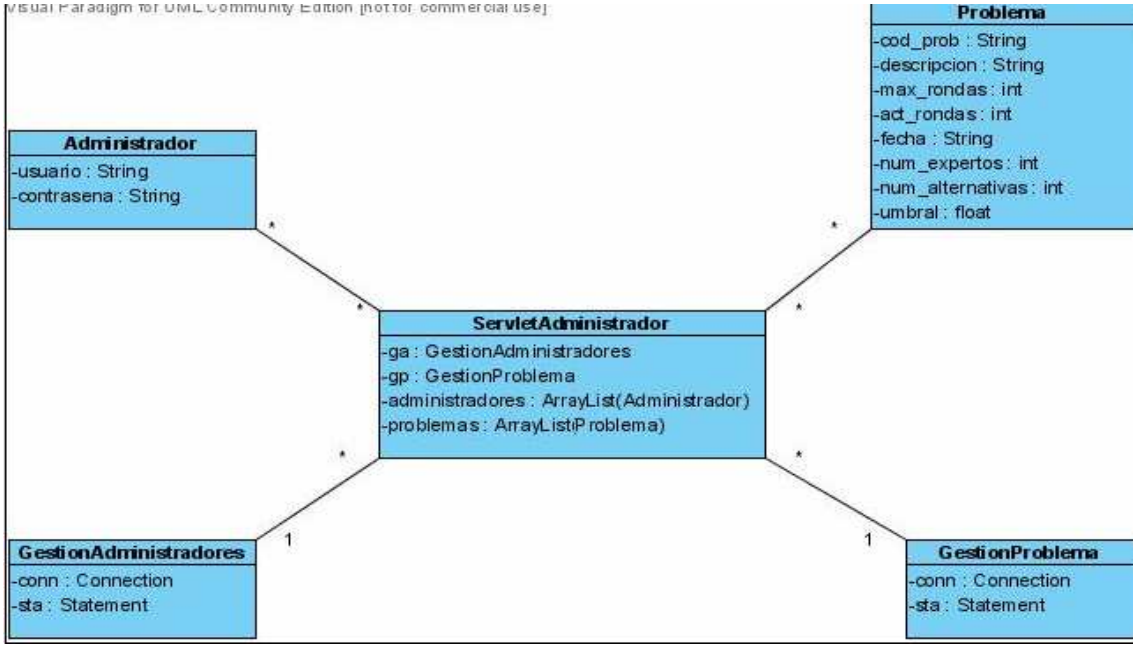


Figura: Diagrama de clases de ServletAdministrador.

ServletExperto

Este Servlet monitoriza la entrada al sistema de los Expertos. Además gestiona también el menú principal del Experto.

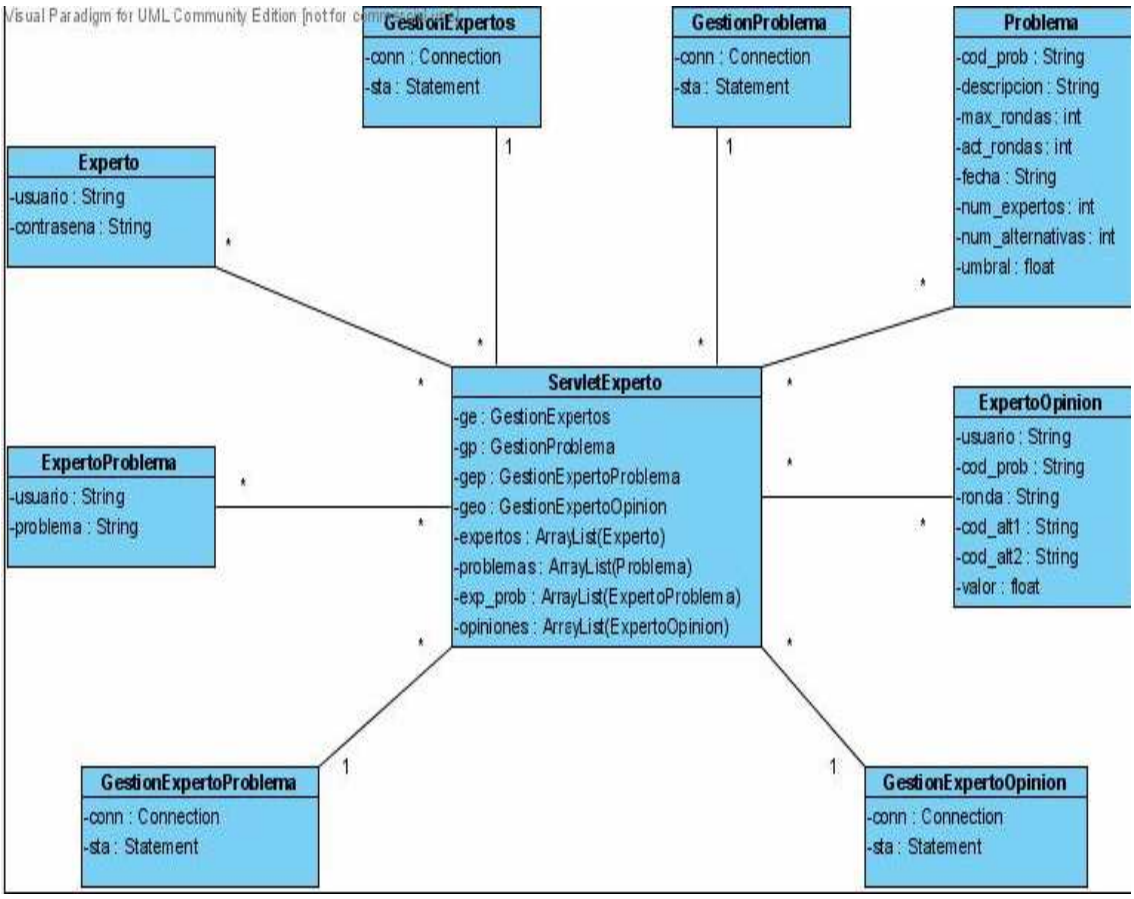


Figura: Diagrama de clases de ServletExperto.

ServletAltaExperto

Este Servlet gestiona la entrada de un nuevo Experto al sistema. Debe ser realizado por un Administrador. Solo es accesible para éstos.

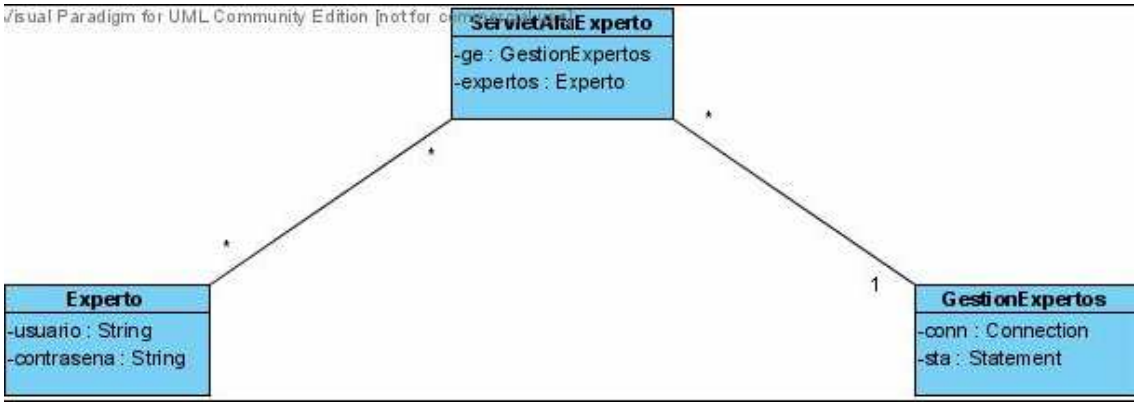


Figura: Diagrama de clases de ServletAltaExperto.

ServletAltaProblema

Este Servlet gestiona la creación de un nuevo problema en su plenitud. Solo puede ser ejecutado por un Administrador.

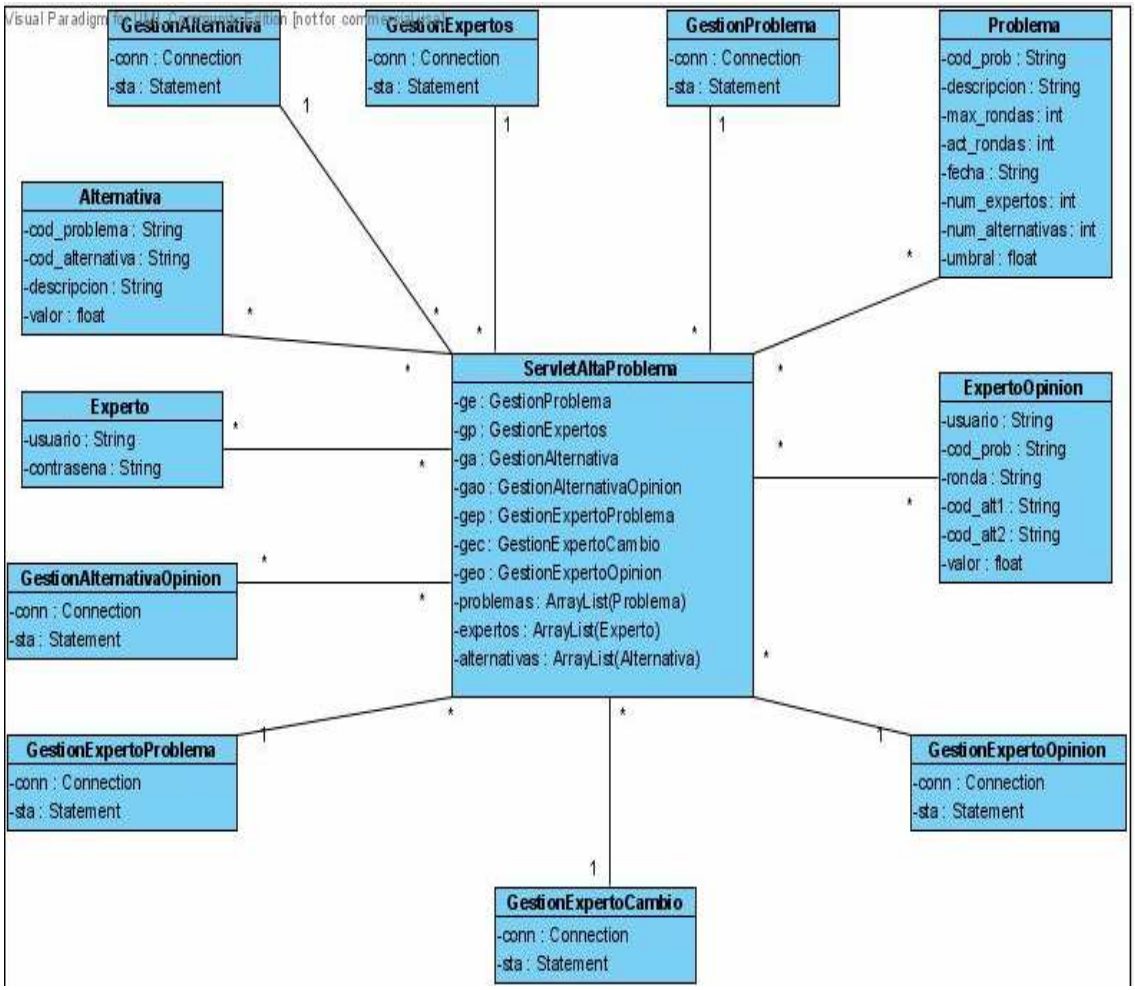


Figura: Diagrama de clases de ServletAltaProblema.

ServletVerProblema

Este Servlet monitoriza la consulta del estado e historial de un problema dado. Solo es accesible para el Administrador que creó dicho problema.

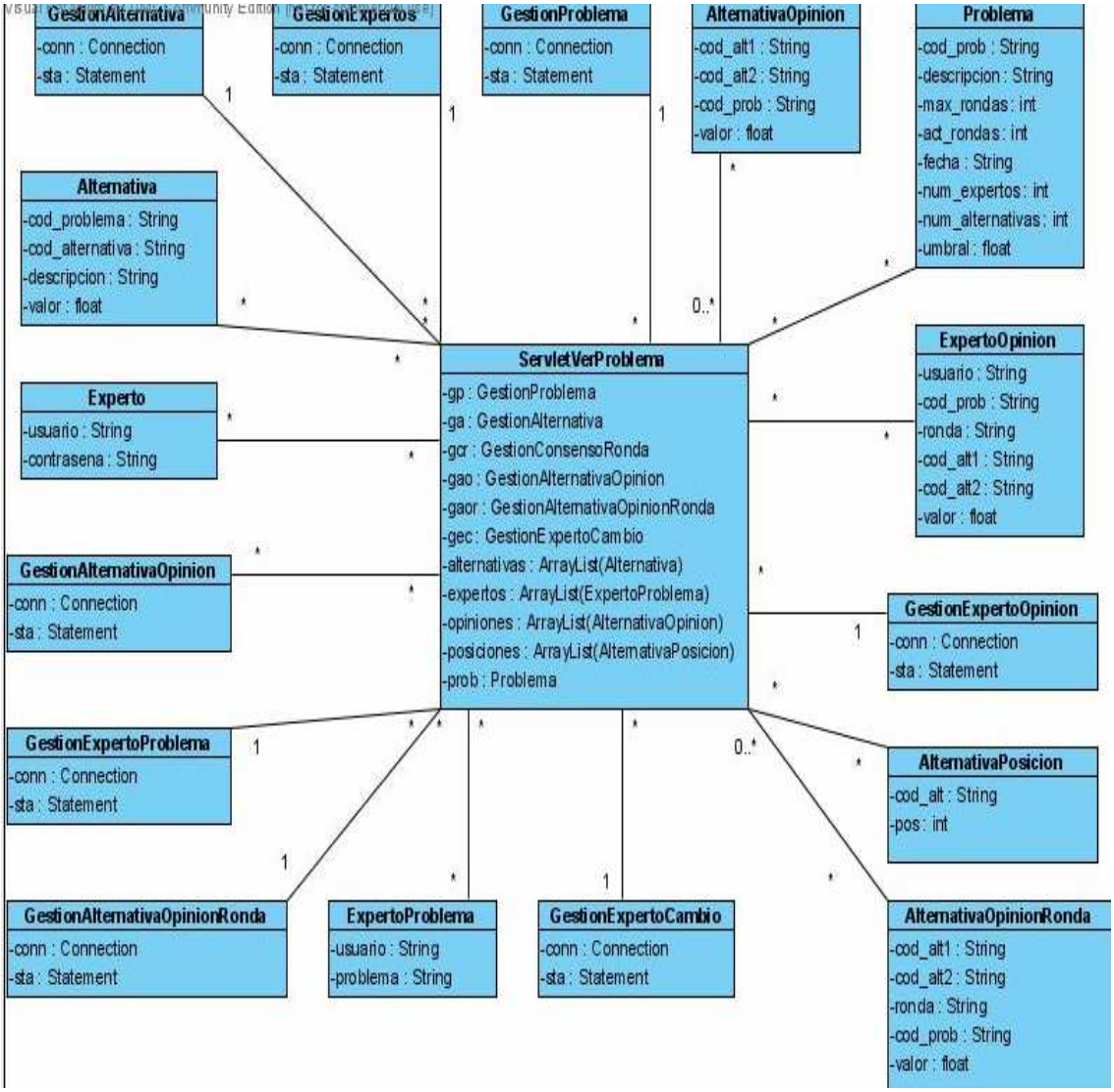


Figura: Diagrama de clases de ServletVerProblema.

ServletVerCambios

Este Servlet procesa y gestiona la elaboración de las recomendaciones para los Expertos de un problema. Para ello necesita de los valores obtenidos por el ServletDarOpinion.

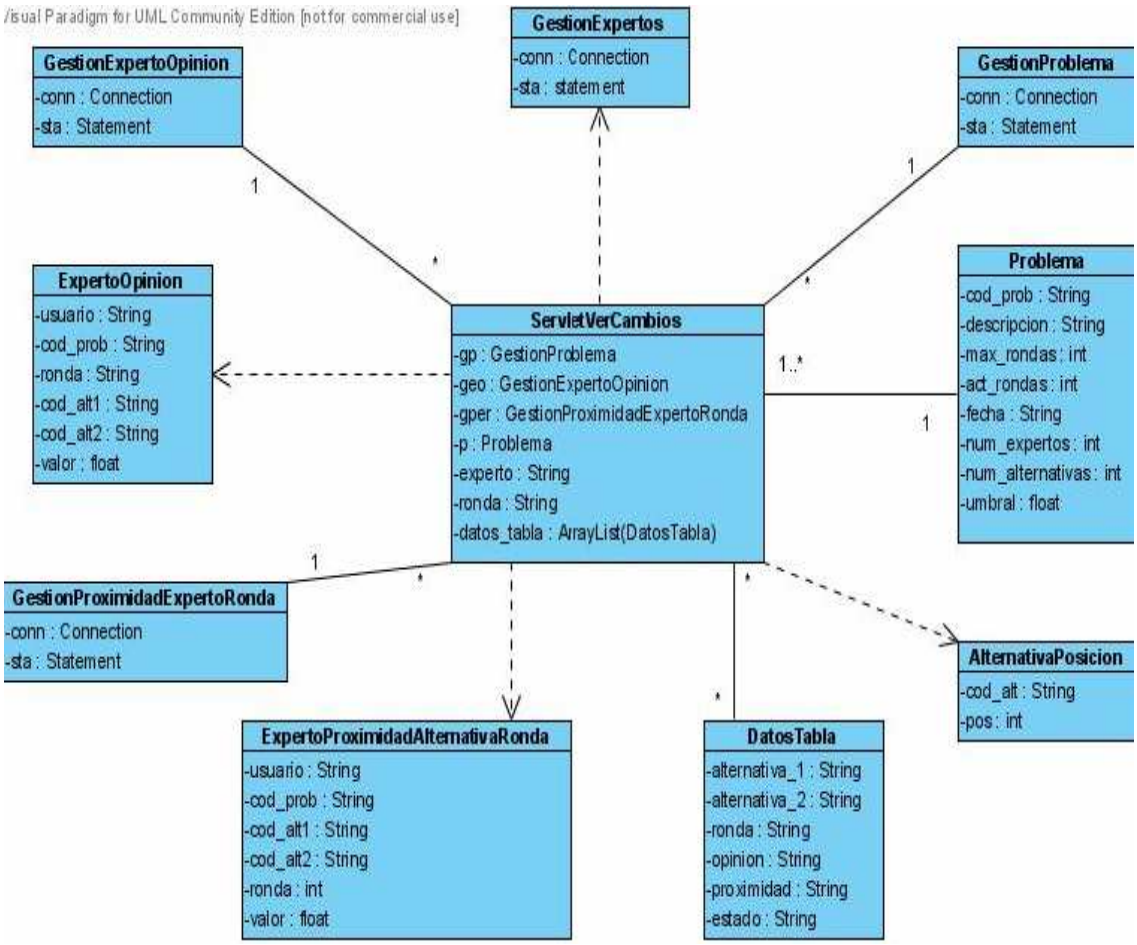


Figura: Diagrama de clases de ServletVerCambios.

ServletDarOpinion

Este es el Servlet principal del sistema implementado. Recoge y gestiona las opiniones de aquellos Expertos asignados al problema, en cada una de las rondas que dure el PC.

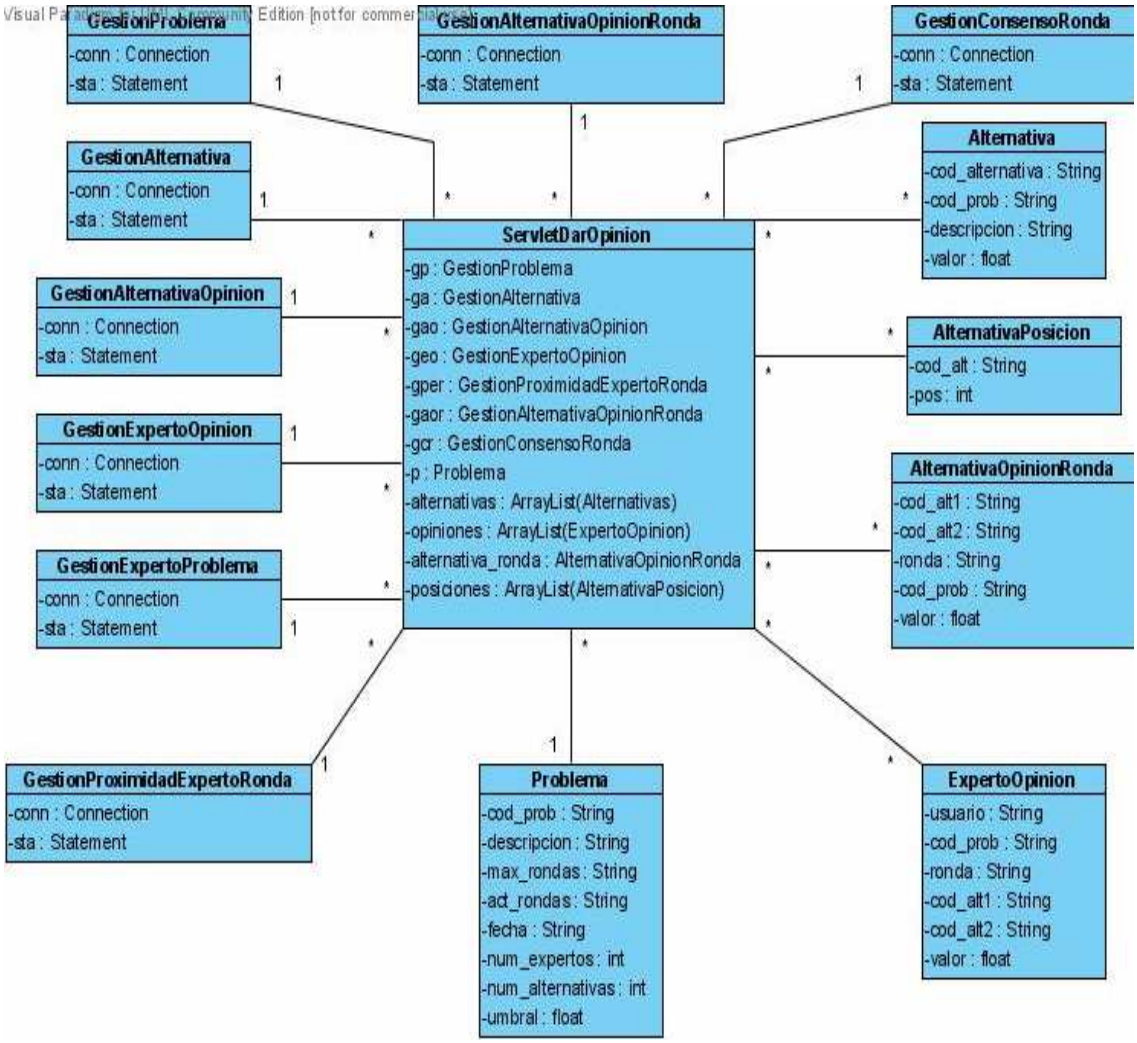


Figura: Diagrama de clases de ServletDarOpinion.

3.4.2. Diseño de los datos.

En esta fase del diseño software, vamos a determinar las estructuras de datos sobre las que vamos a trabajar. En nuestro sistema, intervienen los siguientes elementos de información:

- Un conjunto de administradores que crean y siguen el estado de los problemas.

- Un conjunto de expertos. Son aquellas personas que opinan sobre los distintos problemas en los que estén asignados.

- Un conjunto de problemas. De cada uno de ellos necesitamos saber su fecha de creación, alternativas de solución, expertos asignados, umbral de consenso, máximo de rondas y una breve descripción.

- La opinión de cada uno de los expertos acerca de los problemas en los que estén asignados en cada una de las rondas.

- El grado de proximidad de cada experto a la opinión colectiva.

- El número de cambios que debe realizar cada experto en cada ronda del PC del problema.

Una vez hemos identificado la información que va a manejar nuestra aplicación, y visto el volumen de información que puede llegar a manejar la aplicación, nos hemos decantado por diseñar una base de datos potente que nos permita el acceso a la información de una manera rápida y eficaz. Para ello, necesitamos primero hacer un diseño conceptual de la Base de Datos, que nos permita posteriormente obtener las tablas necesarias. Para este diseño conceptual vamos a emplear el modelo Entidad-Relación.

Modelo Entidad-Relación (E-R).

El modelo Entidad-Relación es una técnica de modelado de datos, que utiliza diagramas entidad-relación. Es la técnica de modelado de datos más extendida, aunque no la única.

Un diagrama entidad-relación está compuesto básicamente de tres elementos básicos:

- Entidades → Son aquellos objetos sobre los que tenemos información. Pueden ser cosas, personas, etc... En el diagrama se representan mediante rectángulos, y cada una ha de tener un nombre que la identifique del resto de entidades.

- Relaciones → Son dependencias entre entidades. Se relacionan mediante flechas y rombos. Cada una de ellas está etiquetada para diferenciarlas del resto. Estas relaciones pueden ser entre una sola entidad (reflexiva) o varias (binarias, ternarias, etc...).

- Atributos → Son las características de una entidad o relación. Se representan mediante elipses con su nombre en el interior.

En un diagrama entidad-relación también hay que tener en cuenta una serie de consideraciones:

- Clave → Es aquella combinación de atributos de una entidad, que permite distinguir a cada instancia del resto.

- Clave foránea → Es aquella combinación de atributos de una entidad, que es clave en otra entidad distinta.

- Entidades débiles → Son aquellas entidades que no tienen clave. Es decir, son aquellas cuyos atributos no permiten diferenciar unas instancias de otras. Necesitan por tanto relacionarse con otras Entidades para poder existir. Se representan con dos rectángulos concéntricos.

- Atributos de relación → Son aquellos que no están asociados a una entidad en concreto, sino a la relación entre una o varias entidades.

- Cardinalidad de la relación→ Es el número de instancias que pueden relacionarse en una relación entre entidades. Ésta puede ser:

    - 1 a 1 → Una instancia de la entidad X se relaciona con una y solo una instancia de la entidad Y.

    - 1 a N → Una instancia de la entidad X puede relacionarse con una o varias instancias de la entidad Y.

    - M a N → Cada instancia de la entidad X puede relacionarse con una o varias instancias de la entidad Y y viceversa.

Una vez descrito el modelo Entidad-Relación, vamos a pasar a modelar nuestra Base de Datos. Para ello vamos primero a construir el Esquema Conceptual (EC), luego continuaremos modelando el Esquema Conceptual Modificado (ECM) a partir del anterior, y por último vamos a obtener las tablas.

Esquema Conceptual (EC).

A partir de los elementos de información identificados, vamos a construir el Esquema Conceptual. Cada elemento se convierte en una entidad. El diagrama es el siguiente:

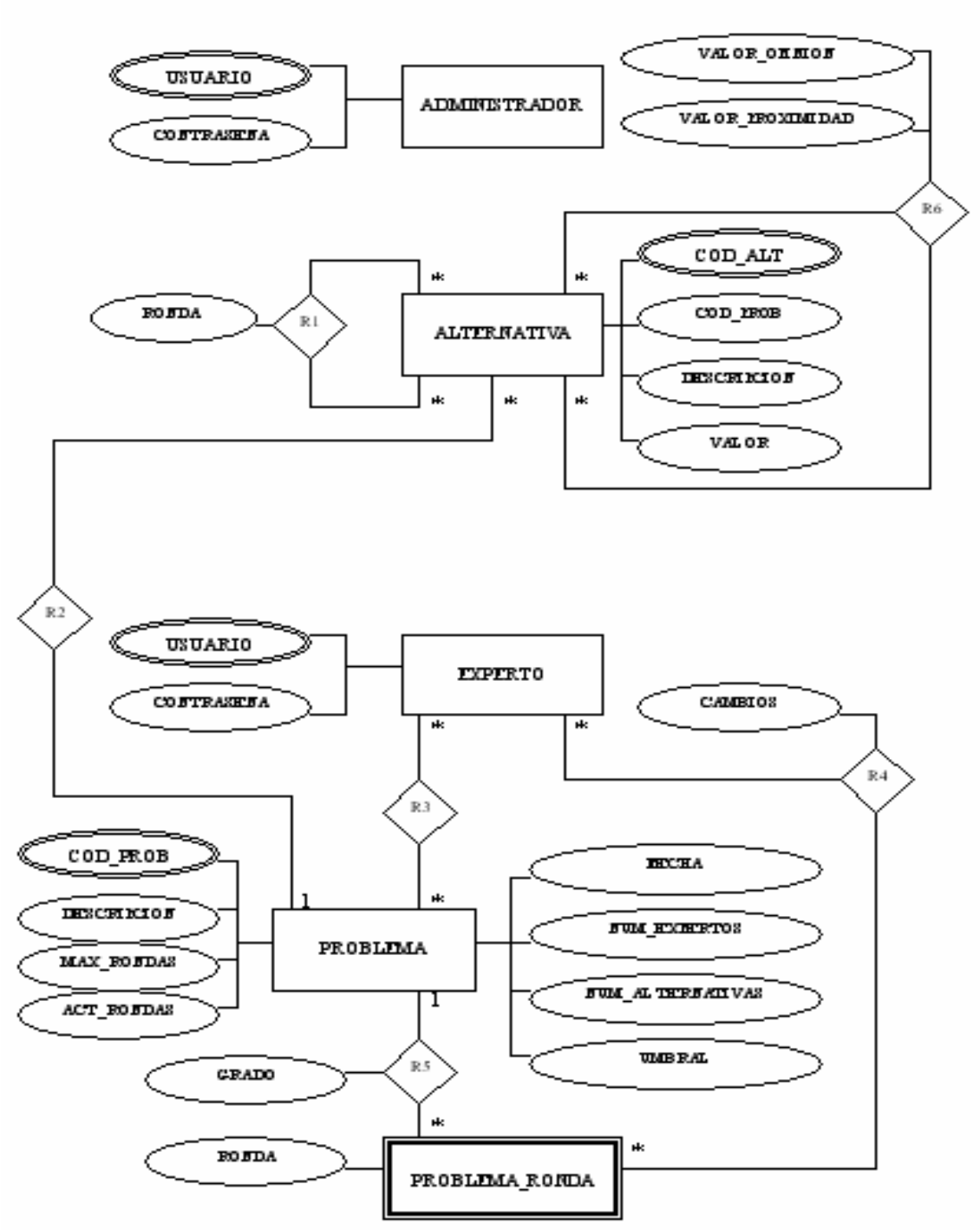


Figura: Esquema conceptual.

R1: Valor.

R2: Contiene.

R3: Es asignado.

R4: Debe cambiar.

R5: Consenso.

R6: Opina.

Esquema Conceptual Modificado (ECM).

Una vez obtenido el EC, pasamos a obtener el ECM. El ECM se obtiene a partir del EC haciendo lo siguiente:

- Eliminar entidades débiles. En nuestro caso tenemos la entidad PROBLEMA_RONDA.

- Sustituir las relaciones M a N por relaciones 1 a N.

- Eliminar los atributos de relación.

Por tanto el ECM de nuestra Base de Datos quedaría como sigue:

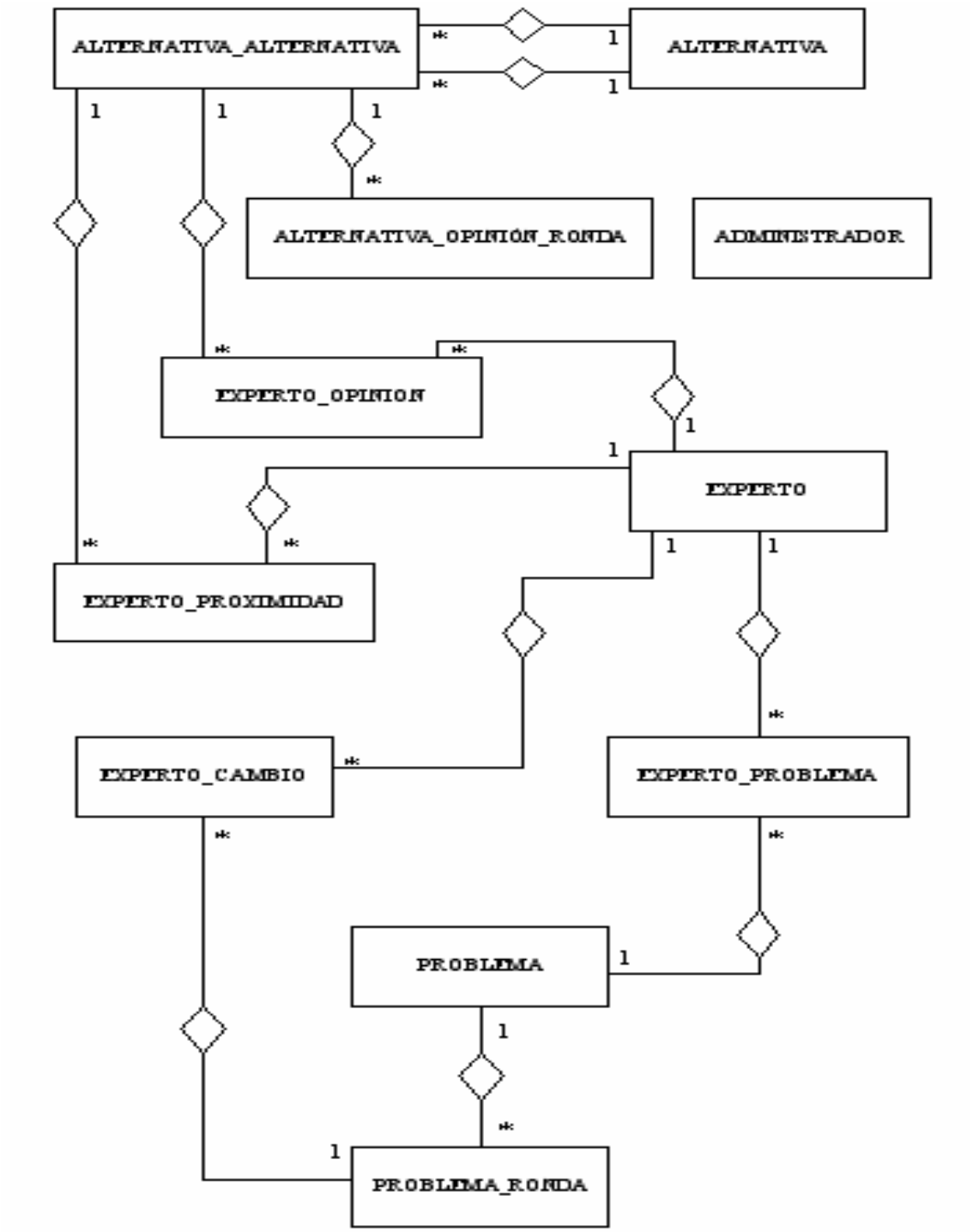


Figura: Esquema Conceptual Modificado.

Los atributos de las entidades en el ECM es el mostrado en la siguiente tabla:

Entidad	Atributos
ADMINISTRADOR	usuario, contrasena.
EXPERTO	usuario, contrasena.
PROBLEMA	cod_prob, descripcion, max_rondas, act_rondas, fecha, num_expertos,

	num_alternativas, umbral.
EXPERTO_PROBLEMA	cod_prob, usuario.
PROBLEMA_RONDA	cod_prob, ronda, grado.
ALTERNATIVA	cod_alt, cod_prob, descripcion, valor.
ALTERNATIVA_ALTERNATIVA	cod_alt1 , cod_alt2.
EXPERTO_OPINION	usuario, cod_prob, ronda, cod_alt1, cod_alt2, valor.
EXPERTO_PROXIMIDAD	usuario, cod_prob, cod_alt1, cod_alt2, ronda, valor.
EXPERTO_CAMBIO	usuario, cod_prob, ronda, cambios.
ALTERNATIVA_OPINION_RONDA	cod_alt1, cod_alt2, ronda, cod_prob, valor.

Tabla: Campos de las entidades del ECM anterior.

Tablas.

El último paso del diseño de nuestra Base de Datos, es obtener las tablas que la modelizan. Estas tablas son obtenidas a partir del ECM con las siguientes reglas:

- Cada entidad del ECM es una tabla.

- Cada atributo de una entidad es un campo de la tabla del mismo nombre.

Por tanto, tenemos las siguientes tablas (Tercera Forma Normal):

A) ADMINISTRADOR.

Tabla que contiene el conjunto de administradores del sistema. Es decir, aquellos usuarios que crean los problemas del sistema, asignan recursos y controlan el proceso de consenso. La tabla es la siguiente:

Columna	Tipo	Descripción
usuario (K)	VARCHAR(10)	Es el identificador de usuario administrador.
contrasena	VARCHAR(15)	Es la contraseña del usuario administrador.

Tabla: Campos de la tabla ADMINISTRADOR.

La clave es el campo usuario.

B) EXPERTO.

Tabla que contiene el conjunto de expertos del sistema. Estos expertos son los usuarios que dan sus opiniones sobre los problemas en los que están asignados. La tabla es la siguiente:

Columna	Tipo	Descripción
usuario (K)	VARCHAR(10)	Es el identificador de usuario experto.
contrasena	VARCHAR(15)	Es la contraseña del usuario experto.

Tabla: Campos de la tabla EXPERTO.

La clave es el campo usuario.

C) PROBLEMA.

Tabla que contiene la información de los distintos problemas que son debatidos en el sistema. La tabla es la siguiente:

Columna	Tipo	Descripción
cod_prob(K)	VARCHAR(3)	Identificador de problema.
descripcion	VARCHAR(60)	Descripción del problema.
max_rondas	INTEGER	Número de rondas máximo del problema.
act_rondas	INTEGER	Número de ronda actual del problema.
fecha	VARCHAR(18)	Fecha de creación del problema.
num_expertos	INTEGER	Número de expertos asignados al problema.
num_alternativas	INTEGER	Número de alternativas del problema.
umbral	FLOAT	Valor del umbral de consenso.

Tabla: Campos de la tabla PROBLEMA.

La clave es el campo cod_prob.

D) EXPERTO_PROBLEMA.

Tabla que relaciona cada experto con los problemas que tiene asignados en el sistema. La tabla es la siguiente:

Columna	Tipo	Descripción
cod_prob(K,F)	VARCHAR(10)	Identificador de problema.

usuario(K,F)	VARCHAR(45)	Identificador de usuario.

Tabla: Campos de la tabla EXPERTO_PROBLEMA.

La clave son los campos cod_prob y usuario. Ambos además son clave foránea de PROBLEMA (cod_prob) y EXPERTO (usuario).

E) PROBLEMA_RONDA.

Tabla que registra el grado de consenso alcanzado en cada una de las rondas del PC del problema. Su tabla es la siguiente:

Columna	Tipo	Descripción
cod_prob(K,F)	VARCHAR(3)	Identificador de problema.
ronda(K)	INTEGER	Número de ronda.
grado	FLOAT	Grado alcanzado en ronda.

Tabla: Campos de la tabla PROBLEMA_RONDA.

La clave son los campos cod_prob y ronda. Además, cod_prob es clave foránea de la tabla PROBLEMA.

F) ALTERNATIVA.

Tabla que contiene el conjunto de alternativas de los distintos problemas del sistema. Su tabla es la siguiente:

Columna	Tipo	Descripción
cod_alt(K)	VARCHAR(4)	Es el identificador de alternativa.
cod_prob(F)	VARCHAR(3)	Es el problema al que pertenece la alternativa.
descripcion	VARCHAR(40)	Descripción de la alternativa.
valor	FLOAT	Valor por defecto de la alternativa.

Tabla: Campos de la tabla ALTERNATIVA.

La clave es el campo cod_alt. Además, cod_prob es clave foránea de PROBLEMA.

G) EXPERTO_OPINION.

Tabla que registra las preferencias de los expertos para cada uno de los problemas en los que esté asignado y en cada una de sus rondas. Su tabla es la siguiente:

Columna	Tipo	Descripción
usuario(K,F)	VARCHAR(10)	Identificador de usuario.
cod_prob(K,F)	VARCHAR(3)	Identificador de problema.
ronda(K)	VARCHAR(2)	Ronda.
cod_alt1(K,F)	VARCHAR(4)	Identificador de alternativa 1.
cod_alt2(K,F)	VARCHAR(4)	Identificador de alternativa 2.
valor	FLOAT	Valor de la opinión para la ronda.

Tabla: Campos de la tabla EXPERTO_OPINION.

La clave son los campos usuario, cod_prob, ronda, cod_alt1 y cod_alt2. Además, usuario es clave foránea de EXPERTO y cod_prob, cod_alt1, cod_alt2 de ALTERNATIVA.

H) EXPERTO_PROXIMIDAD.

Tabla que registra la proximidad de la opinión del experto a la opinión colectiva de la ronda. Su tabla es la siguiente:

Columna	Tipo	Descripción
usuario(K,F)	VARCHAR(10)	Identificador de usuario.
cod_prob(K,F)	VARCHAR(3)	Identificador de problema.
cod_alt1(K,F)	VARCHAR(4)	Identificador de alternativa 1.
cod_alt2(K,F)	VARCHAR(4)	Identificador de alternativa 2.
ronda(K)	INTEGER	Número de ronda.
valor	FLOAT	Valor de la proximidad.

Tabla: Campos de la tabla EXPERTO_PROXIMIDAD.

La clave la forman todos los campos menos valor. Además, usuario es clave foránea de EXPERTO, y cod_prob, cod_alt1 y cod_alt2 de ALTERNATIVA.

I) EXPERTO_CAMBIO.

Tabla que almacena el número de cambios que debe realizar cada experto, en cada una de las rondas del PC de aquellos problemas en los que esté asignado. Su tabla es la siguiente:

Columna	Tipo	Descripción
usuario(K,F)	VARCHAR(15)	Identificador de usuario.
cod_prob(K,F)	VARCHAR(3)	Identificador de problema.
ronda(K)	VARCHAR(2)	Ronda.
cambios	VARCHAR(2)	Número de cambios.

Tabla: Campos de la tabla EXPERTO_CAMBIO.

La clave la forman los campos usuario, cod_prob y ronda. Además, usuario es clave foránea de EXPERTO y cod_prob a su vez de PROBLEMA.

J) ALTERNATIVA_OPINION_RONDA.

Tabla que almacena las opiniones colectivas de los expertos, para cada una de las rondas, para cada uno de los problemas del sistema en los que están asignados. Su tabla es la siguiente:

Columna	Tipo	Descripción
cod_alt1(K,F)	VARCHAR(4)	Identificador de la alternativa 1.
cod_alt2(K,F)	VARCHAR(4)	Identificador de la alternativa 2.
ronda(K)	VARCHAR(2)	Ronda a la que pertenece.
cod_prob(K,F)	VARCHAR(3)	Problema al que pertenece.
valor	FLOAT	Valor de la preferencia.

Tabla: Campos de la tabla ALTERNATIVA_OPINIÓN_RONDA.

La clave de esta tabla la forman los campos cod_alt1, cod_alt2, ronda y cod_prob. Además, cod_alt1, cod_alt2 y cod_prob son clave foránea de ALTERNATIVA.

3.4.3. Diseño de la interfaz.

En esta fase del Diseño del Sistema, vamos a definir la apariencia visual de la aplicación, es decir, se define la interfaz visual entre el usuario y la aplicación. Sin duda, realizar un buen diseño de la interfaz resulta primordial, ya que ésta debe presentarse atractiva al usuario de la aplicación pero a la vez le debe resultar fácil de entender y trabajar sobre ella. Buscamos por tanto, una interfaz de usuario atractiva y que haga muy sencilla la interacción del usuario con la aplicación.

Guías de estilo.

Antes de ponerse a diseñar una interfaz de usuario, se debe definir el estilo de la misma. Esto es de vital importancia cuando el diseño va a ser compartido entre varios diseñadores, ya que ayuda a mantener la coherencia interna de la interfaz.

Si embargo, en contra de lo que pueda parecer en un principio, también es de mucha utilidad definir una guía de estilo cuando hay un diseñador encargado de la interfaz. Esto se debe a varias razones:

- A veces es posible que mantener la coherencia y consistencia de una interfaz, si esta es muy grande o muy ambiciosa, sea algo complicado incluso si solo hay un diseñador. Necesita una base.

- El diseñador primitivo puede, por las más diversas razones abandonar el diseño y es de utilidad para su sustituto contar con una guía de estilo predefinida para no tener que empezar de cero otra vez.

Metáforas.

Una metáfora es el empleo de un objeto con un significado o dentro de un contexto diferente al habitual. Al diseñar una interfaz gráfica, la utilización de metáforas resulta muy útil ya que permiten al usuario, por comparación con otro objeto o concepto, comprender de una manera más intuitiva las diversas tareas que la interfaz permite desarrollar.

Al igual que pasa en el ámbito de la literatura o la lingüística, para que una metáfora cumpla con su cometido, el desarrollador de la aplicación y el usuario final de ésta, deben tener una base cultural similar. Es muy posible que el uso de un icono de manera metafórica sea entendido de una manera por el usuario occidental y de otra bien distinta por un usuario de oriente. Hay que intentar, por lo tanto, que las metáforas empleadas sean lo más universales posibles para que así sean comprendidas a la perfección por la mayor parte del público potencial de la aplicación.

Las aplicaciones de escritorio de Windows suelen seguir la Guía de Estilo XP y utilizan una serie de metáforas con las que el usuario está plenamente familiarizado (por ejemplo, una lupa con un signo '+' en su interior establece que la función del icono es, inequívocamente, la de realizar un aumento de zoom). En el mundo de las aplicaciones web, también existen una cantidad de metáforas de amplia difusión como puede ser, por ejemplo el celebre carrito de la compra que emplean casi todos los comercios online.

Pero las metáforas no solo dependen del tipo de aplicación (escritorio o web) sino también del ámbito de la misma. Por ejemplo, el carrito de la compra es una metáfora conocida por todos pero si nuestra aplicación no va a vender nada al usuario no resulta conveniente utilizarla ya que puede confundir.

StoryBoards.

Los **Storyboards** son ilustraciones mostradas en secuencia con el objetivo de servir de guía para entender una historia, previsualizar una animación o seguir la estructura de una película, aplicación, etc... antes de realizarse o filmarse.

Al tener la aplicación una orientación web, la interfaz de usuario de ella será una interfaz web. A la hora de definir una interfaz web, son muchas las guías o plantillas de estilo que pueden emplearse. En este tema no hay un estándar definido como ocurre en otros tipos de aplicaciones. Cada autor emplea una hoja de estilo personal para sus contenidos e interfaces web.

A la hora de decantarnos por una hoja de estilos concreta, debemos tener en cuenta, que ciertas tonalidades de la pantalla o de los elementos que aparecen en ella pueden ser más cómodos para unas personas que para otras. Un ejemplo es el tema del color. Hay ciertas culturas, en las que el color amarillo es síntoma de mala suerte,

etc... Para evitar este tipo de situaciones, no vamos a seguir una hoja de estilo determinada, sino que vamos a usar la que viene por defecto. De esta manera si un usuario se siente más identificado con una que con otra, podrá aplicarle la que más desee. Veamos ahora un listado de las consideraciones de diseño de la interfaz que hemos seguido.

- Los mensajes de error que aparezcan en la propia pantalla, tales como introducir un valor incorrecto, los mostraremos en rojo.

- Mostrar en pantalla un cuadro con las preferencias solo de la ronda anterior. Si mostramos las de todas las rondas, puede llevar al usuario a confusión.

- Las recomendaciones mostradas al usuario vendrán en tres tonalidades distintas:

    - Rojo → Debe incrementar.

    - Verde → Debe decrementar.

    - Azul → En consenso.

- Los botones de aceptar o pasar al menú siguiente siempre aparecerán en el lado derecho de la pantalla.

- Mostrar la información del problema arriba de la ventana y de forma resumida.

Que el administrador pueda consultar el estado del problema en cada una de sus rondas a través de botones. De esta manera evitamos llenar la pantalla de datos.

3.5. Implementación.

Una vez visto el Diseño del Sistema, la siguiente etapa de la Ingeniería del Software es la implementación. La implementación por tanto es la actividad final de la Ingeniería del Software. Consiste en transformar las actividades anteriores (Análisis y Diseño del Sistema) en código fuente. Para ello hay que elegir el lenguaje de programación que mejor se adapte al sistema que vamos a desarrollar.

Comenzaremos describiendo los lenguajes de programación, así como las tecnologías utilizadas en esta implementación. Estas tecnologías son la arquitectura Cliente-Servidor y la arquitectura Servlet.

3.5.1. Arquitectura.

Para la implementación del Modelo de Consenso propuesto, nos hemos basado en la arquitectura cliente-servidor. El concepto de la arquitectura cliente-servidor, proporciona una forma eficiente de utilizar todos estos recursos de máquina de tal forma que la seguridad y fiabilidad que proporcionan los entornos mainframe a la red de área local.

La arquitectura cliente-servidor, es un modelo para el desarrollo de sistemas de información en el que las transacciones se dividen en procesos independientes. Estos procesos a su vez cooperan entre sí para intercambiar información, servicios o recursos.

El cliente es aquel que inicia el diálogo o solicita los recursos, mientras que el servidor es quien responde a las solicitudes de los distintos clientes.

En este modelo las aplicaciones se dividen de forma que el servidor contiene la parte que debe ser compartida por varios usuarios, y en el cliente permanece sólo lo particular de cada usuario.

Los clientes realizan generalmente las siguientes funciones:

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

Por su parte, los servidores realizan las siguientes funciones:

- Gestión de periféricos compartidos.
- Control de accesos concurrentes a bases de datos compartidas.
- Enlaces de comunicaciones con otras redes LAN o WAN.

Cuando un cliente requiere un servicio, éste lo solicita al servidor correspondiente. El servidor le responde proporcionándoselo.

Normalmente, pero no necesariamente, el cliente y el servidor están ubicados en distintos procesadores.

Los clientes se suelen situar en ordenadores personales y/o estaciones de trabajo y los servidores en procesadores departamentales o de grupo.

Entre las principales características esta arquitectura, podemos destacar las siguientes:

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

3.5.1.1. Arquitectura Básica.

Esta arquitectura se puede estructurar en cinco niveles, según la funcionalidad que asumen el cliente y el servidor. Esta estructura la podemos ver en el siguiente diagrama:

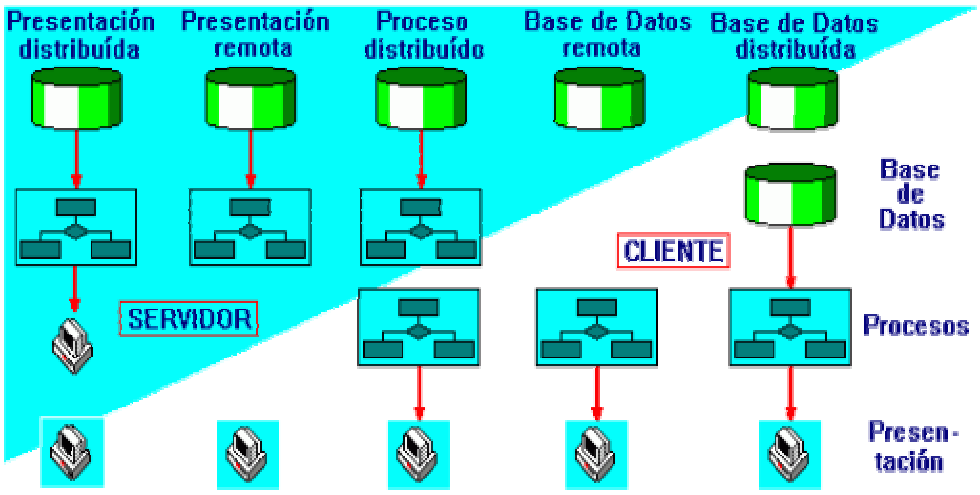


Figura: Variantes de la arquitectura cliente-servidor

El primer nivel es conocido como Presentación distribuida. En él, el cliente asume parte de las funciones de presentación de la aplicación, ya que siguen existiendo programas en el servidor dedicados a esta tarea. Esta técnica no exige cambios en las aplicaciones orientadas a los terminales de cliente, pero dificulta su mantenimiento. El servidor almacena la totalidad de los datos.

En el segundo nivel, conocido como Presentación remota, la aplicación está soportada directamente por el servidor. Sin embargo, la presentación es totalmente remota y reside en el cliente. Los terminales del cliente soportan la captura de datos, incluyendo una validación parcial de los mismos y una presentación de las consultas.

En el tercer nivel, conocido como Proceso distribuido, la lógica de los procesos se divide entre los distintos componentes del cliente y del servidor. El diseñador de la aplicación debe definir los servicios y las interfaces del sistema de información de forma que los papeles de cliente y servidor sean intercambiables, excepto en el control de los datos que es responsabilidad exclusiva del servidor.

En el cuarto nivel, conocido como Base de Datos remota, el cliente realiza tanto las funciones de presentación como los procesos. Por su parte, el servidor almacena y gestiona los datos que permanecen en una base de datos centralizada.

En el quinto y último nivel, conocido como Base de Datos distribuida, el reparto de tareas es similar al del nivel anterior. El gestor de base de datos divide sus componentes entre el cliente y el servidor. Las interfaces entre ambos están dentro de las funciones del gestor de datos y, por lo tanto, no tienen impacto en el desarrollo de las aplicaciones.

3.5.1.2. Estructura física.

La idea principal de la arquitectura cliente-servidor consiste en aprovechar la potencia de los ordenadores personales para realizar sobre todo los servicios de presentación y, según el nivel, algunos procesos o incluso algún acceso a datos locales. De esta forma, estamos liberando al servidor de ciertas tareas para que pueda realizar otras más rápidamente.

También existe una plataforma de servidores que sustituye al ordenador central tradicional y que da servicio a los clientes autorizados. Incluso a veces el antiguo ordenador central se integra en dicha plataforma como un servidor más. Estos servidores suelen estar especializados por funciones (seguridad, cálculo, bases de

datos, comunicaciones, etc.), aunque, dependiendo de las dimensiones de la instalación se pueden reunir en un servidor una o varias de estas funciones.

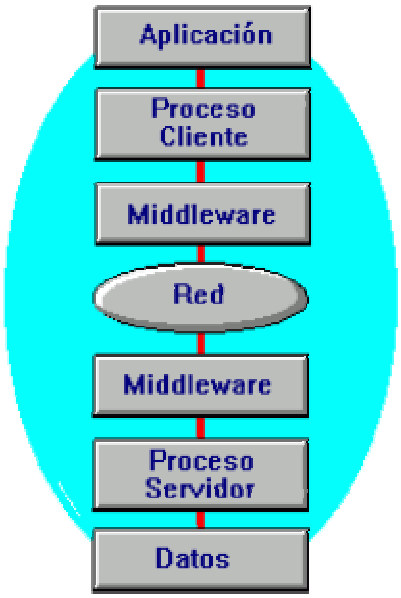


Figura: Capas de una aplicación cliente-servidor.

Las unidades de almacenamiento masivo en esta arquitectura se caracterizan por incorporar elementos de protección que evitan la pérdida de datos y permiten multitud de accesos simultáneos (alta velocidad, niveles RAID, etc.).

Para la comunicación de todos estos elementos se emplea un sistema de red que se encarga de transmitir la información entre clientes y servidores. Físicamente consiste en un cableado (coaxial, par trenzado, fibra óptica, etc.) o en conexiones mediante señales de radio o infrarrojas, dependiendo de que la red sea local (LAN), metropolitana (MAN) o de área extensa (WAN).

Para la comunicación de los procesos con la red se emplea un tipo de equipo lógico denominado *middleware* que controla las conversaciones. Su función es independizar ambos procesos (cliente y servidor).

3.5.1.3. Estructura lógica.

Una de las principales aportaciones de esta arquitectura a los sistemas de información es la interfaz gráfica de usuario. Gracias a ella se dispone de un manejo más fácil e intuitivo de las aplicaciones mediante el uso de un dispositivo tipo ratón.

En cuanto a los datos, cabe señalar que en la arquitectura cliente-servidor las duplicidades (copias y comparaciones de datos), teniendo siempre una imagen única y correcta de los mismos disponible en línea para su uso inmediato.

Todo esto tiene como fin que el usuario de un sistema de información soportado por una arquitectura cliente-servidor trabaje desde su estación de trabajo con distintos datos y aplicaciones, sin importarle dónde están o dónde se ejecuta cada uno de ellos.

3.5.1.4. Ventajas e inconvenientes.

La arquitectura cliente-servidor tiene las siguientes ventajas:

A) Aumento de la productividad.

- Los usuarios pueden utilizar herramientas que le son familiares, como hojas de cálculo y herramientas de acceso a bases de datos.

- Los usuarios pueden construir soluciones particularizadas que se ajusten a sus necesidades cambiantes.

- Una interfaz gráfica de usuario consistente reduce el tiempo de aprendizaje de las aplicaciones.

B) Menores costes de operación.

- Permiten un mejor aprovechamiento de los sistemas existentes, protegiendo la inversión. Mejora el rendimiento de uso de los dispositivos.

- Proporcionan un mejor acceso a los datos. La interfaz de usuario ofrece una forma homogénea de ver el sistema, independientemente de los cambios o actualizaciones que se produzcan en él y de la ubicación de la información.

- El movimiento de funciones desde un ordenador central hacia servidores o clientes locales origina el desplazamiento de los costes de ese proceso hacia máquinas más pequeñas y por tanto, más baratas.

C) Mejora en el rendimiento de la red.

- La arquitectura cliente-servidor la necesidad de mover grandes bloques de información por la red hacia los ordenadores personales o estaciones de trabajo para su proceso.

- Los servidores controlan los datos, procesan peticiones y después transfieren sólo los datos requeridos a la máquina cliente.

- Tanto el cliente como el servidor pueden escalarse para ajustarse a las necesidades de las aplicaciones.

- La existencia de varias CPUs proporciona una red más fiable: un fallo en uno de los equipos no significa necesariamente que el sistema deje de funcionar.

- Tanto los clientes como el servidor pueden renovarse para aumentar sus funciones y capacidad de forma independiente, sin afectar al resto del sistema.

Por el contrario, la arquitectura cliente-servidor tiene los siguientes inconvenientes:

- Hay una alta complejidad tecnológica al tener que integrar una gran variedad de productos.

- Requiere un fuerte rediseño de todos los elementos involucrados en los sistemas de información (modelos de datos, procesos, interfaces, comunicaciones, almacenamiento de datos, etc.).

- Es más difícil asegurar un elevado grado de seguridad en una red de clientes y servidores que en un sistema con un único ordenador centralizado.

- A veces, los problemas de congestión de la red pueden degradar el rendimiento del sistema por debajo de lo que se obtendría con una única máquina (arquitectura centralizada).

- La interfaz gráfica de usuario puede a veces ralentizar el funcionamiento de la aplicación.

- El quinto nivel de esta arquitectura (bases de datos distribuidas) es técnicamente muy complejo y en la actualidad hay muy pocas implantaciones que garanticen un funcionamiento totalmente eficiente.

Existen multitud de costes ocultos (formación en nuevas tecnologías, licencias, cambios organizativos, etc.) que encarecen su implantación.

3.5.2. Lenguajes de programación.

Dado el enfoque web de nuestra aplicación, el lenguaje HTML (Hyper Text Markup Lenguage) se antoja vital para la comunicación entre usuarios y servidor. Pero con este único lenguaje no basta.

Para la comprobación de las preferencias introducidas, necesitamos un lenguaje que valide los datos, en la máquina cliente, antes de ser enviarlos al servidor de Base de Datos. Por ello debemos emplear en situaciones muy puntuales JavaScript. Este metalenguaje, se ejecuta en el lado del cliente, por lo que dichos datos serán validados en el cliente antes de ser enviados al servidor. Con esto conseguimos un ahorro en el volumen de datos enviados a través de la red. Utiliza una sintaxis similar al lenguaje C, y puede ir incrustado en el código HTML usando etiquetas de tipo *<script type=text/javascript>.*

Para la implementación del núcleo central de la aplicación, y el módulo de procesamiento, hemos usado el lenguaje Java. De este lenguaje hemos usado la extensión J2EE, ya que la versión J2SE no tiene soporte para los Servlets. Existen otras tecnologías con la misma funcionalidad que los Servlets de Java. Son los cgi y los jsp. Al final nos hemos decantado por los Servlets por estar más extendidos.

En el servidor de la Base de Datos, vamos a usar el lenguaje SQL (Structured Query Language). Es el lenguaje más extendido entre los distintos Sistemas de Gestión de Bases de Datos (SGPD).

3.5.3. Herramientas de desarrollo.

Para el desarrollo de las páginas web, hemos empleado el Microsoft FrontPage. Existen en el mercado multitud de herramientas para la creación y diseño de páginas web mucho más completos que éste que hemos empleado. Pero con este programa,

que viene en el paquete de Microsoft Office, tenemos acceso a toda la funcionalidad de la que queremos dotar a estas páginas web.

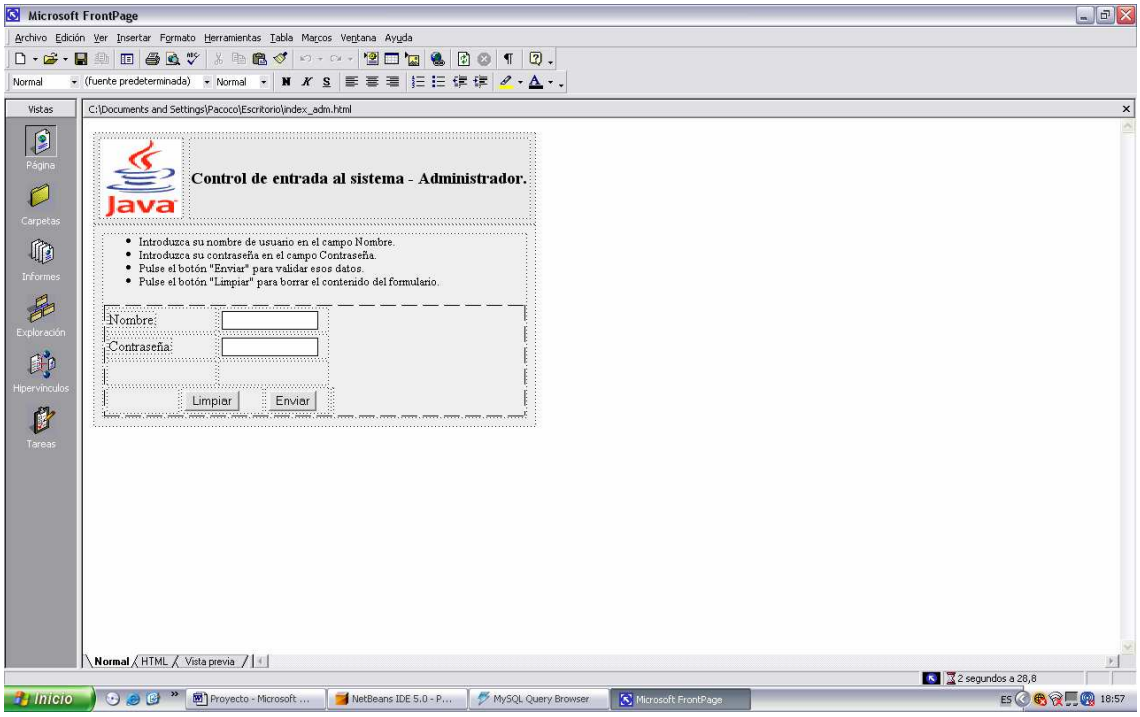


Figura: Captura de pantalla de la aplicación Microsoft FrontPage.

Para el desarrollo de la aplicación, hemos usado el entorno de desarrollo (IDE) Netbeans. Existen otros IDEs muy conocidos y extendidos como son Eclipse, JBuilder, etc...Todos tienen una funcionalidad similar. Este IDE viene ya con la JVM (Java Virtual Machine) incluida en el paquete. Si no viene, tenemos que instalarnos Java en nuestra máquina por separado.

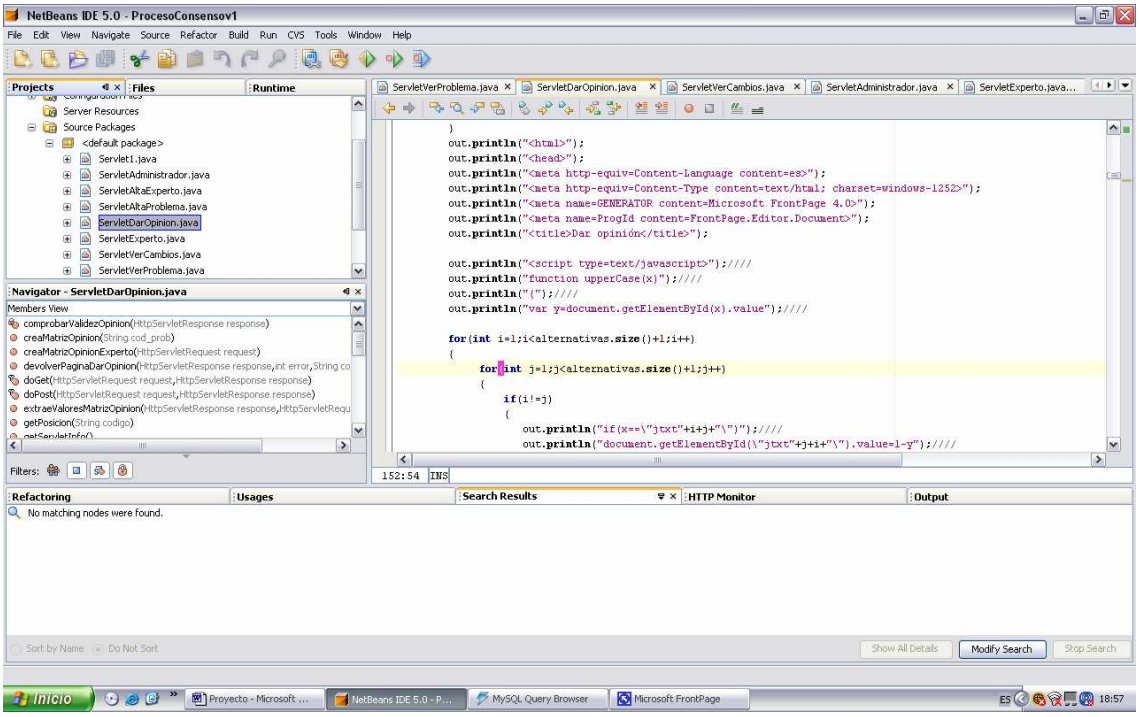


Figura: Captura del entorno de desarrollo Java Netbeans.

Para el desarrollo de la Base de Datos, hemos empleado una Base de Datos MySQL. Este cliente de Base de Datos funcione realmente bien y es de libre distribución. Otros clientes de Bases de Datos más consistentes como por ejemplo DB2 o Oracle son de pago. Con MySql tenemos más que suficiente potencia para nuestra Base de Datos.

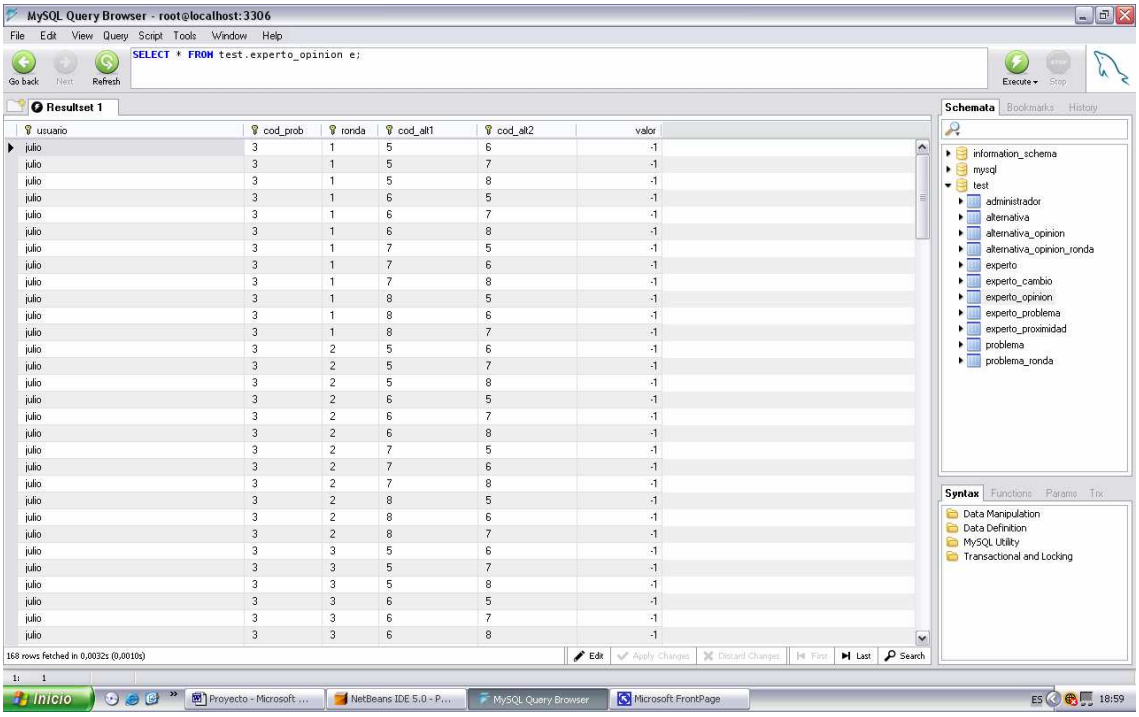


Imagen: Captura de la ventana principal del cliente MySql.

Para el manejo de servidores de aplicaciones, vamos a emplear un servidor Apache Tomcat. De nuevo como indicamos antes, existen servidores de aplicaciones más potentes y seguros como son WebLogic o WebSphere, pero son servidores de pago, y con Tomcat tenemos toda la funcionalidad que buscamos.

En el Anexo I puede consultar la instalación de todas estas aplicaciones.

CAPITULO 4: CONCLUSIONES.

ANEXO I:
MANUAL DE INSTALACIÓN DEL SERVIDOR.

Consideraciones previas y material necesario.

Las únicas consideraciones previas que han de tenerse en cuenta, son que durante todos este manual de instalación se ha supuesto que la unidad principal de disco duro es C:, y que la unidad de disco óptico es D:.

Todo el material necesario para instalar y dejar operativo el servidor se encuentra disponible en el CD que acompaña a esta memoria. Vaya al directorio D:\ProcesoConsenso y compruebe que en el directorio se encuentran los siguientes archivos:

- apache-tomcat-5.5.12.exe
- jdk-6u1-nb-5_5_1-win-ml.exe
- mysql-5.0.17-win32.zip
- mysql-administrator-1.1.6-win.ins
- mysql-query-browser-1.1.18-win.ins
- ProcesoConsensov1.zip
- sql.cmd
- tablas.sql

Si todos estos ficheros están contenidos en el CD-ROM que acompaña a la memoria, podemos comenzar la instalación de nuestro servidor inmediatamente. En caso contrario, póngase en contacto con el responsable de la aplicación para subsanar el percance.

Paso 1: Instalar la jdk y el entorno de desarrollo.

El primero de todos los pasos es instalar la jdk, es decir la JVM (Java Virtual Machine). A continuación habría que instalar el entorno de desarrollo Netbeans. Existe un paquete que puede ser descargado desde el web www.netbeans.org que contiene ambos programas. De esta manera, la instalación es más sencilla. El archivo que contiene tanto la jdk como el Netbeans es jdk-6u1-nb-5_5_1-win-ml.exe.

Ejecute este fichero y le saldrá la siguiente ventana:

Figura: Ventana de entrada del asistente de instalación del jdk y Netbeans.

Pulse el botón Next. A continuación el instalador le mostrará la siguiente ventana:

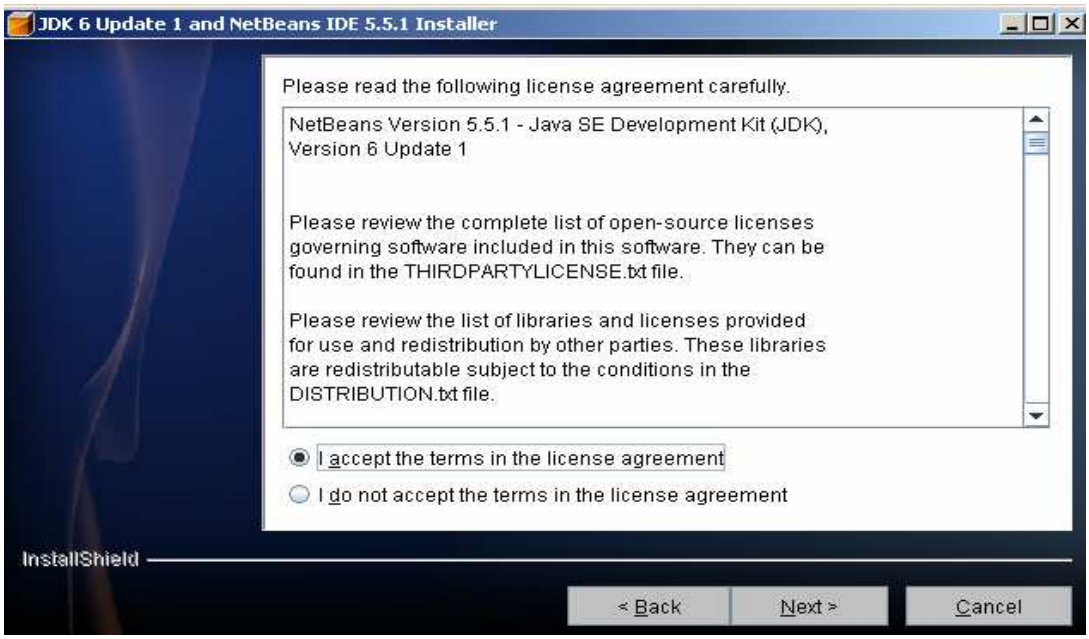


Figura: Ventana de conformidad con la licencia de Sun.

Pinche en la opción “I accept the terms un the license agreement”, y a continuación pulse el botón Next. A continuación, el programa de instalación le pedirá en que ruta queremos instalar el entorno de desarrollo Netbeans y la jdk.

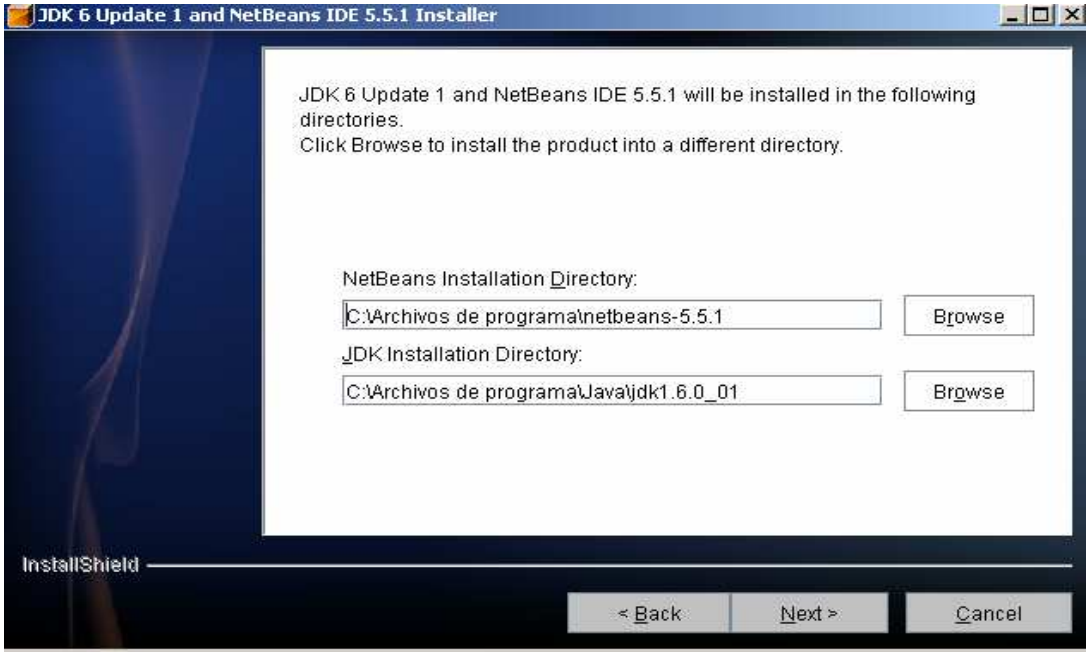
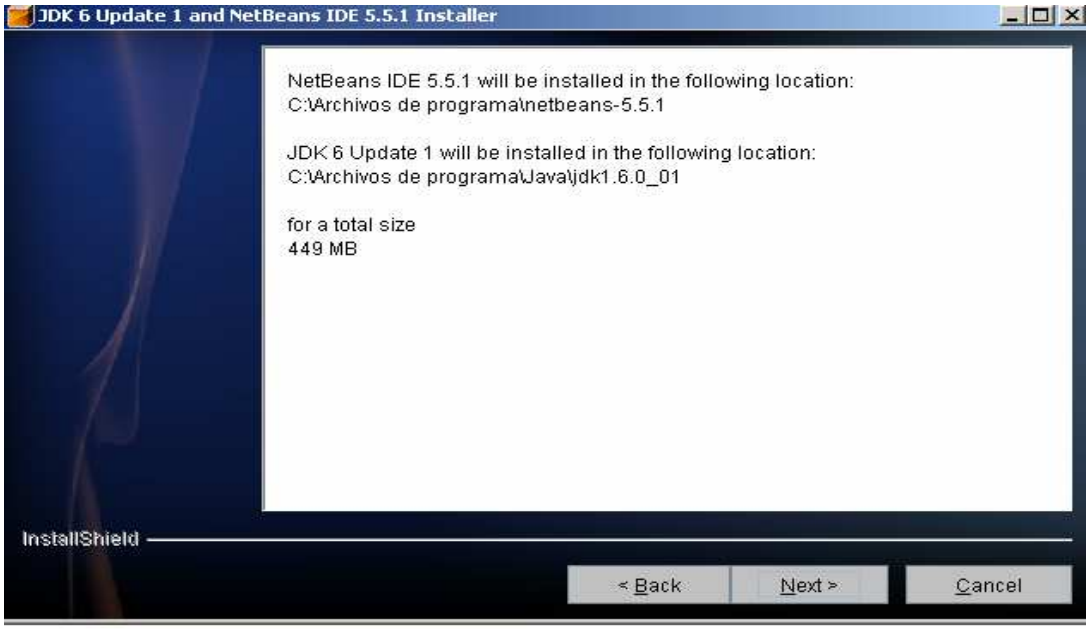


Figura: Ventana para introducir las rutas de instalación del jdk y del Netbeans

Si ya tenemos instalada la jdk en el servidor, solo aparecerá en pantalla el cuadro con la ruta de instalación del entorno Netbeans. Introduzca la ruta deseada y pulse en botón Next.

A continuación, una ventana de confirmación le indicará la ruta de instalación de ambos programas y el tamaño total. Si está de acuerdo pulse el botón Next. En caso contrario pulse el botón Back.



Si ha pulsado el botón Next, la instalación habrá comenzado. Esta operación puede durar varios minutos. Cuando haya finalizado, el programa de instalación devolverá una ventana con el siguiente mensaje:

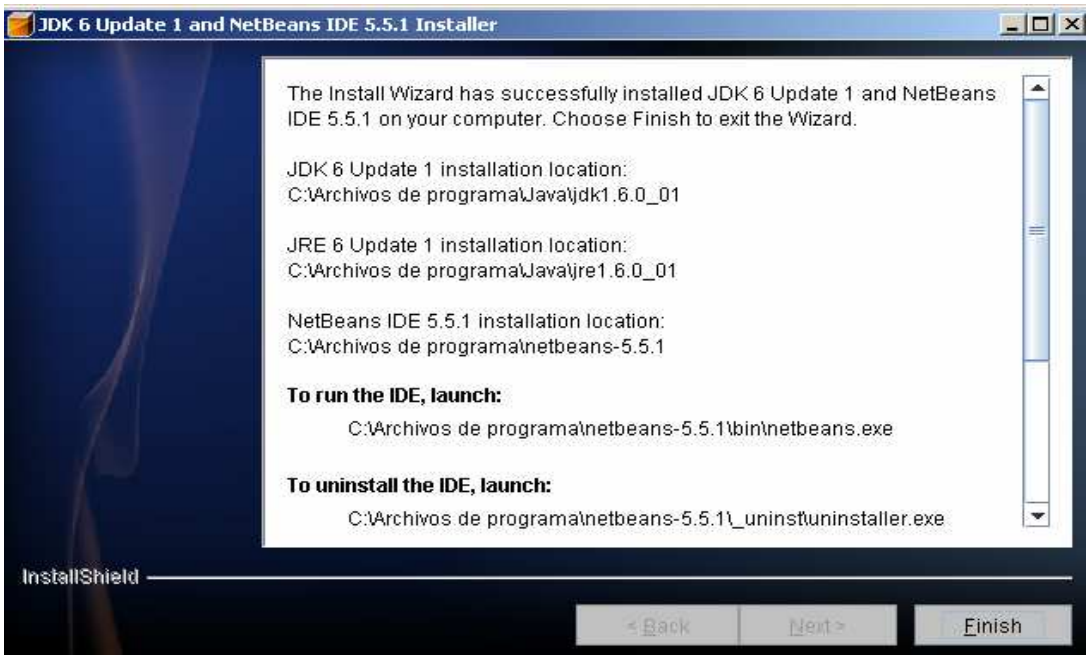


Figura: Ventana de confirmación de instalación del jdk y Netbeans.

Por último, pulse el botón Finish para terminar la instalación de estos dos programas.

Paso 2: Instalar el servidor Apache Tomcat.

El siguiente paso en la instalación es el de la instalación del servidor Apache Tomcat. Apache es un servidor http de código abierto y multiplataforma desarrollado por la Apache Software Foundation, en cuya web (www.apache.org) se pueden conseguir tanto la última versión, como sus múltiples módulos de desarrollo y mucha documentación.

Para instalar el servidor Apache, ejecute el fichero de instalación (*apache-tomcat-5.5.12.exe*) que acompaña al CD-ROM de la memoria. Éste le mostrará la siguiente ventana:

Figura: Ventana principal de instalación de Apache Tomcat.

Pulse el botón Next para comenzar el asistente de instalación del Apache Tomcat. Acto seguido, éste le mostrará la ventana de conformidad con la licencia de la Apache Software Foundation.

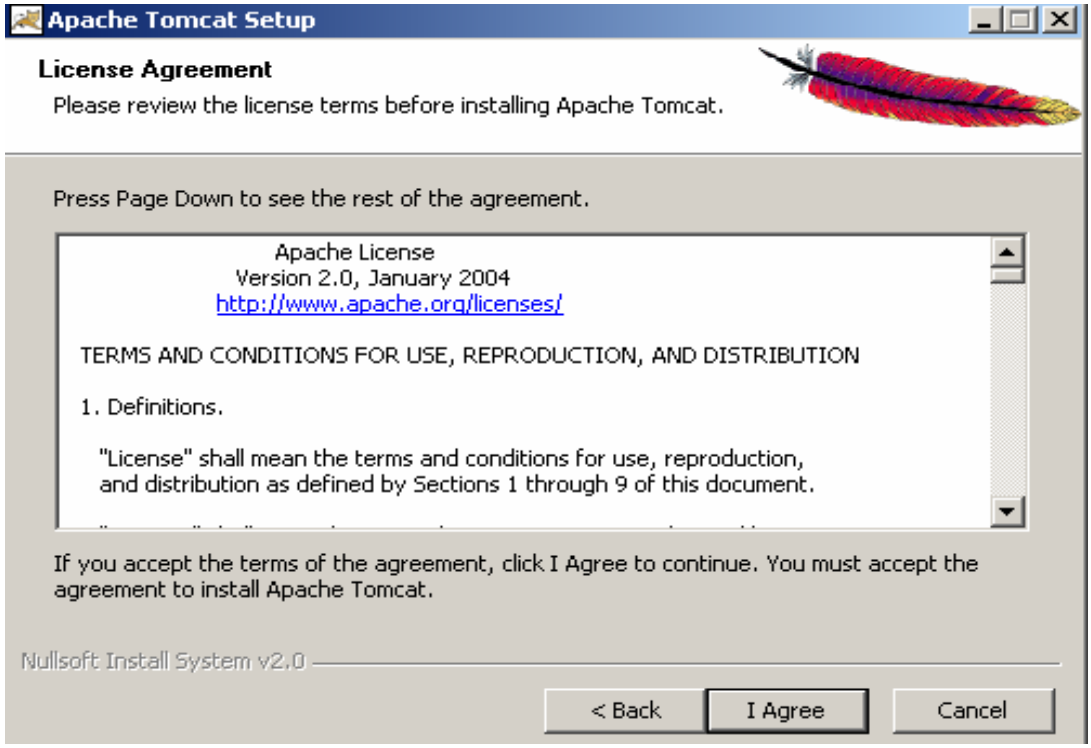


Figura: Ventana de conformidad con la licencia de la Apache Software Foundation.

Pulse el botón I Agree para continuar con el asistente de instalación. A continuación el asistente le pedirá el tipo de instalación deseada. La opción normal es la nuestra. La completa contiene algunos ejemplos, pero es salvo esto la misma.

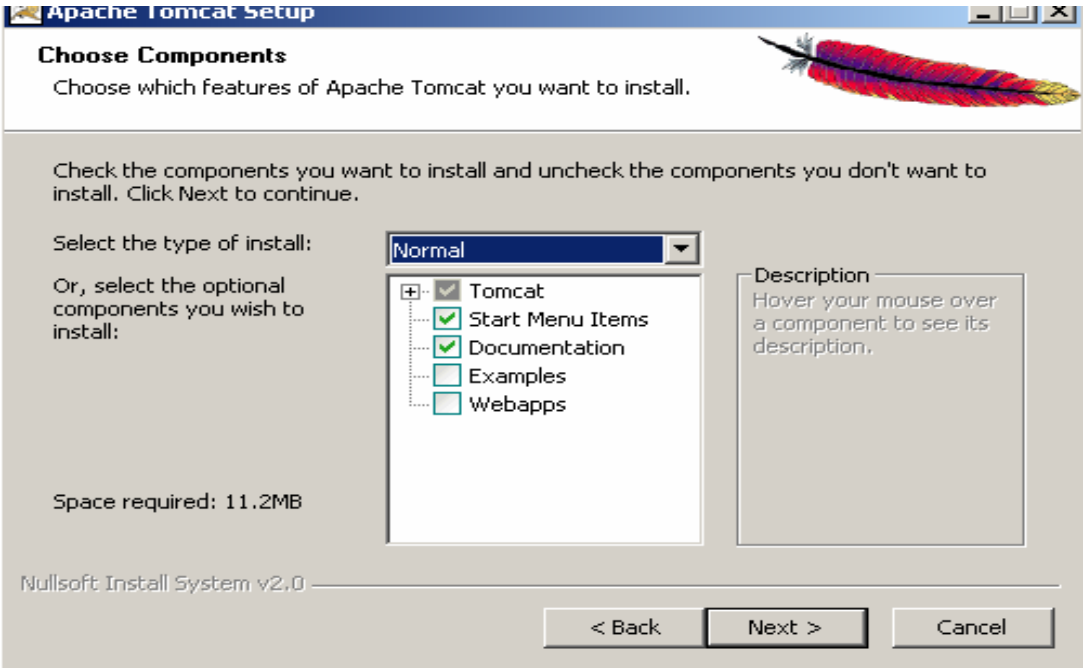


Figura: Ventana de tipo de instalación del Apache Tomcat.

Pulse en el botón Next para continuar con la instalación. El asistente mostrará la ventana para indicar la ruta en la que queremos instalar el servidor Apache Tomcat.

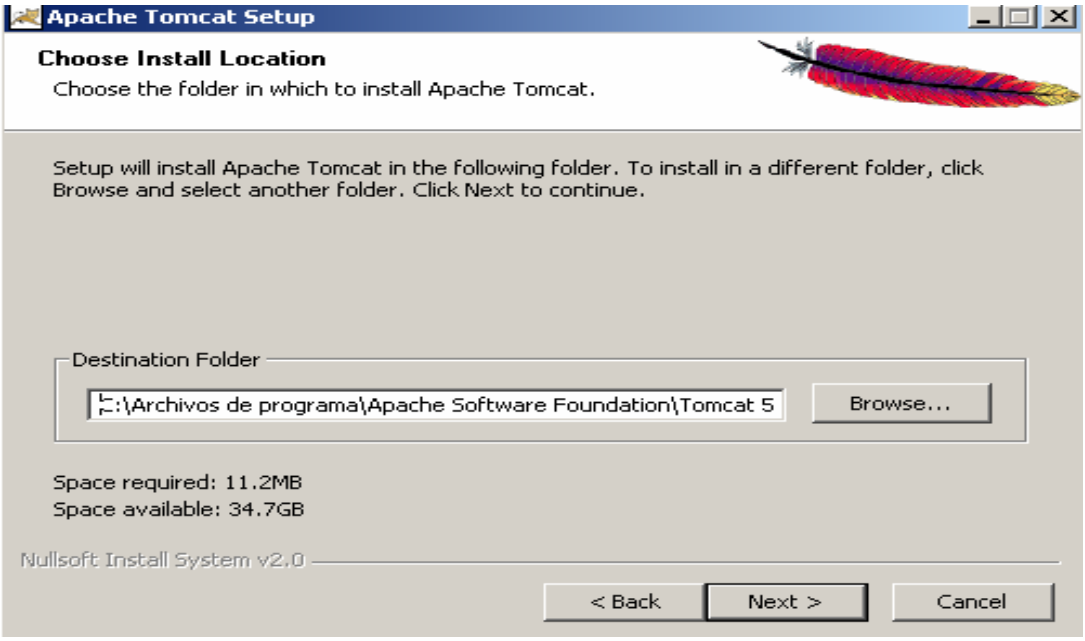


Figura: Ventana de confirmar ruta de instalación del servidor Apache Tomcat.

Pulse el botón Next y el asistente mostrará la siguiente ventana de instalación. En ella se le pedirá el puerto para las conexiones http y el identificador y password del administrador del servidor.

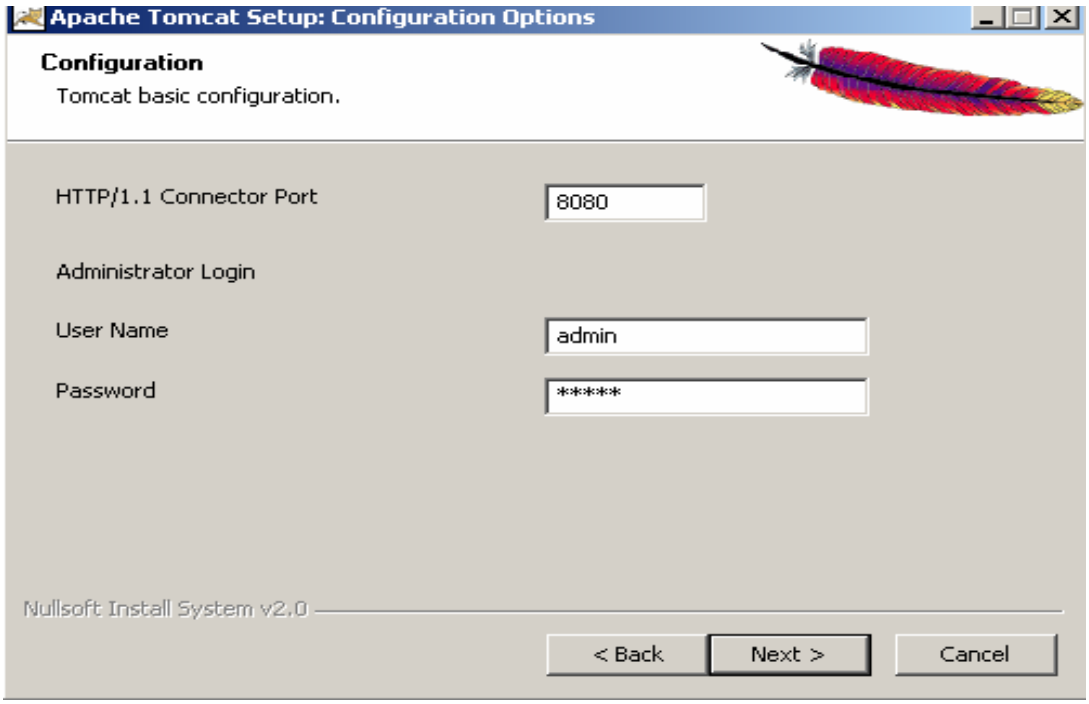


Figura: Ventana de puertos y clave de acceso del Administrador del servidor.

Introduzca el puerto deseado, 8080 por defecto, y la contraseña del administrador. A continuación pulse el botón Next. El asistente nos pedirá ahora la ruta donde tenemos instalada la jre (instalada en el Paso 1).

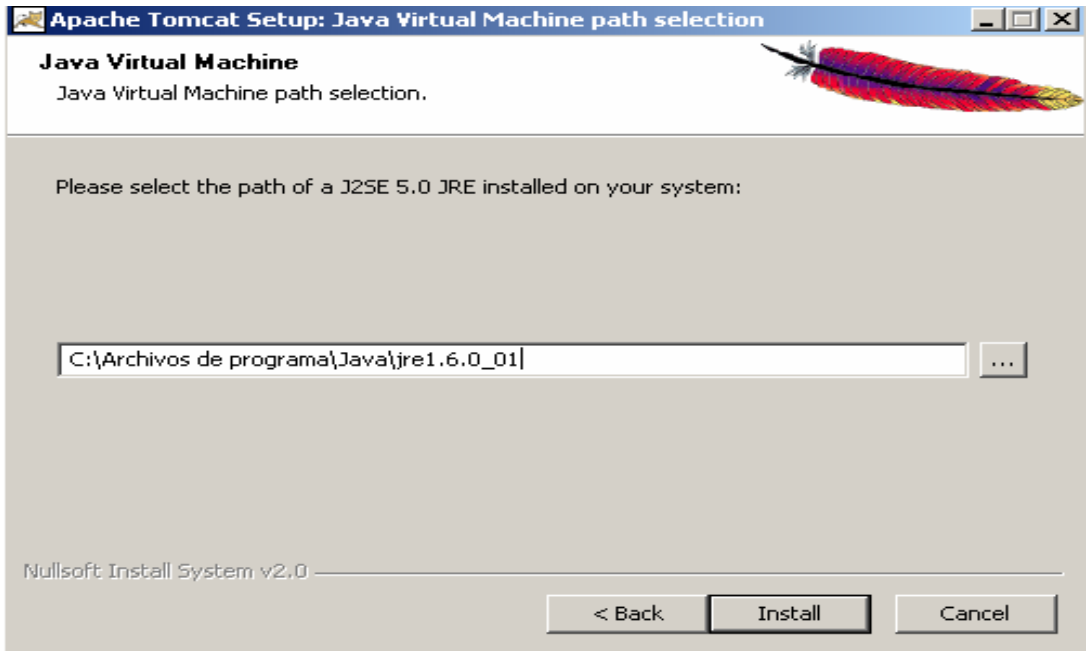


Figura: Ventana de introducir ruta de la jre.

Una vez introducida, pulse el botón Install, y la instalación dará comienzo. Finalizada la instalación, el asistente mostrará la siguiente ventana:

Figura: Ventana de confirmación de la instalación del servidor Apache Tomcat.

Pulse el botón Finish para terminar la instalación. El servidor Apache Tomcat ya está instalado.

Paso 3: Instalar la base de datos MySql.

El tercer paso de la instalación de la aplicación es el de instalar tanto el Administrador como el cliente de la Base de Datos en el servidor. En primer lugar, instalamos el servidor de la Base de Datos MySql. Para ello ejecutamos el fichero Setup.exe contenido en el directorio D:\ ProcesoConsenso\mysql-5.0.17-win32 del CD-ROM que acompaña a la memoria.

Nada más ejecutar este fichero, el asistente nos mostrará la siguiente ventana:

Figura: Ventana principal del asistente de instalación del servidor MySQL.

Si pulsamos el botón Next, el asistente nos preguntará por el tipo de instalación deseada. Seleccionamos la opción Typical y pulsamos el botón Next.

Figura: Ventana de selección de tipo de instalación.

A continuación, el asistente nos informará del tipo seleccionado y de la ruta de instalación del servidor MySQL. Una vez conforme, pulse el botón Install para comenzar la instalación.

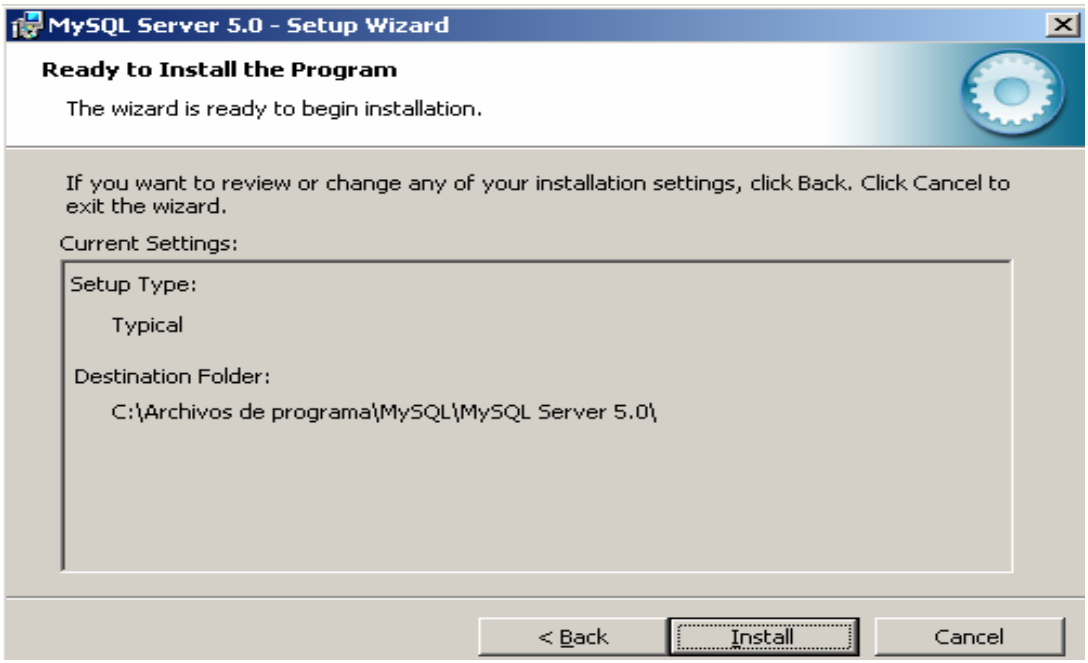


Figura: Ventana de comienzo de la instalación del servidor MySQL.

Una vez concluida la instalación del servidor de Base de Datos MySQL, el asistente nos pedirá que creemos una cuenta de MySQL.com. Podemos obviar este paso si pinchamos en la opción Skip Sign-Up y a continuación pulsamos el botón Next.

Finalmente, la siguiente ventana nos informará de que el servidor MySQL ha sido instalado satisfactoriamente.

Figura:Ventana de final de la instalación del servidor MySQL.

Una vez instalado el servidor de Base de Datos MySQL, vamos a configurarlo. En la siguiente ventana pulsamos el botón Next para iniciar el asistente de configuración del servidor MySQL.

Figura: Ventana principal del asistente de configuración del servidor MySQL.

Seleccionamos la opción Server Machine y pulsamos el botón Next.

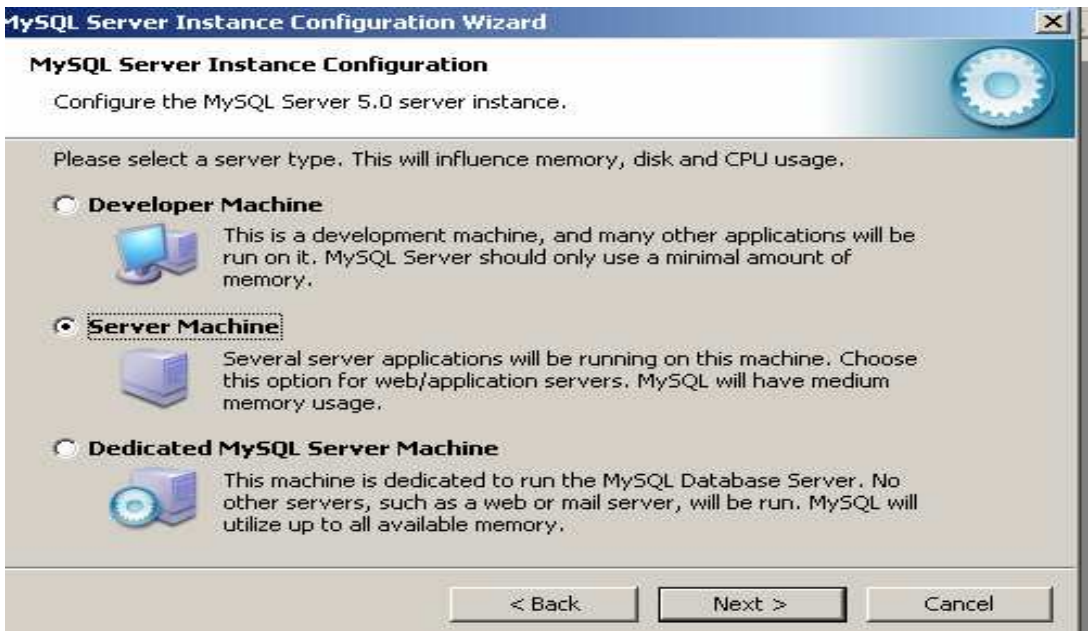


Figura:Ventana de selección de tipo de servidor de Base de Datos.

A continuación el asistente nos pregunta qué tipo de base de datos queremos tener. Seleccionamos Multifunctional Database, para que nos permita tener accesos concurrentes a la Base de Datos.

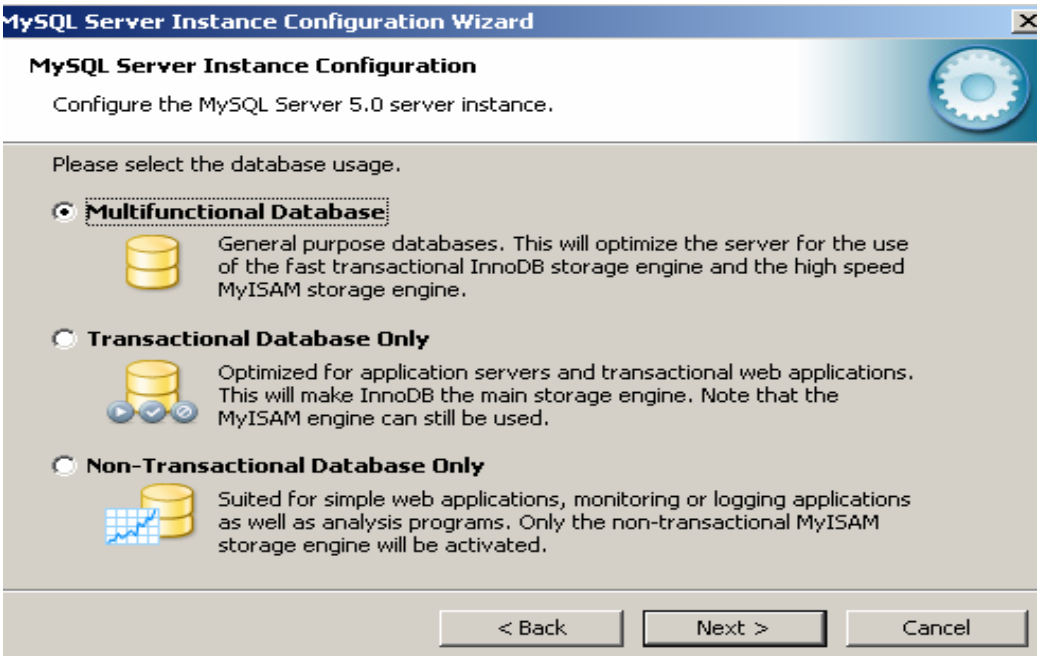


Figura: Ventana de selección de tipo de Base de Datos.

Pulsamos en el botón Next, para acceder a otra ventana en la que nos pide el asistente donde van a residir las tablas en el servidor. Le decimos que en mismo directorio que el de instalación de MySQL.

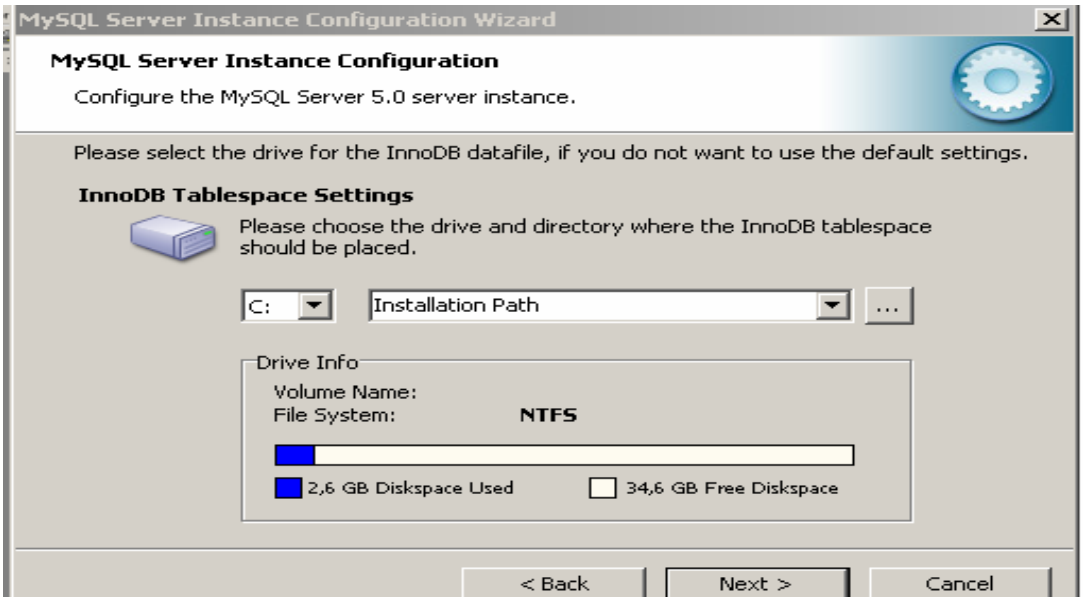


Figura: Ventana de selección de directorio de la Base de Datos.

Ahora debemos seleccionar el número máximo de conexiones simultáneas al servidor. Podemos elegir la primera opción o la tercera. Una vez elegido, pulsamos el botón Next.

Figura: Ventana para seleccionar el máximo de conexiones simultáneas.

A continuación, debemos indicar cual va a ser el puerto usado para las conexiones al servidor de la Base de Datos MySQL. Dejamos el que viene por defecto:

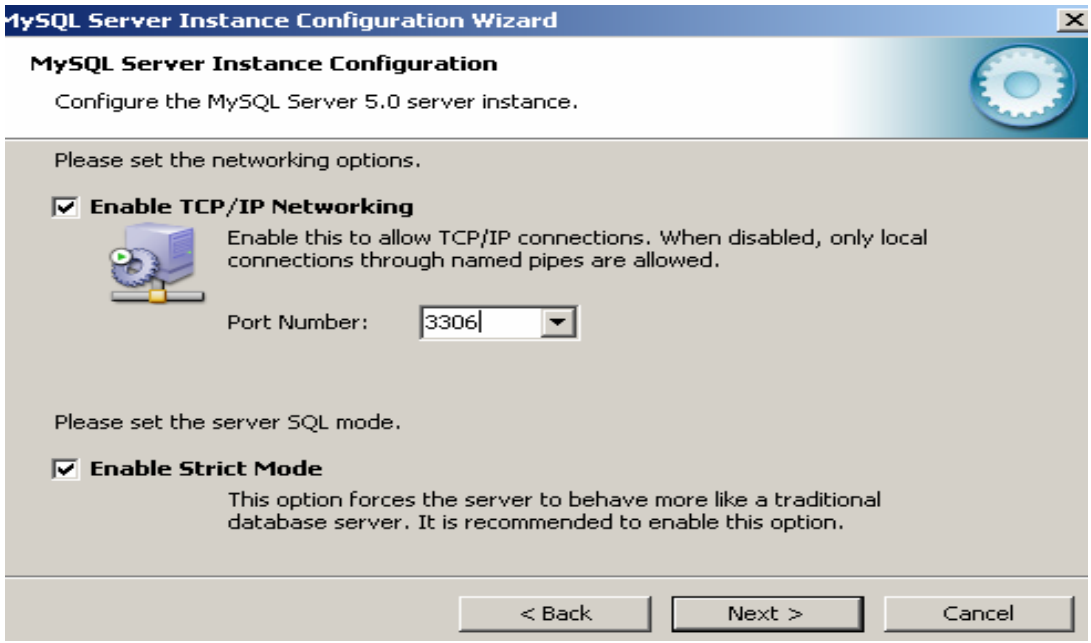


Figura: Selección del puerto de conexión al servidor MySQL.

Ahora es el turno de elegir el conjunto de caracteres para representar la información de la Base de Datos. Seleccionamos el que viene por defecto.

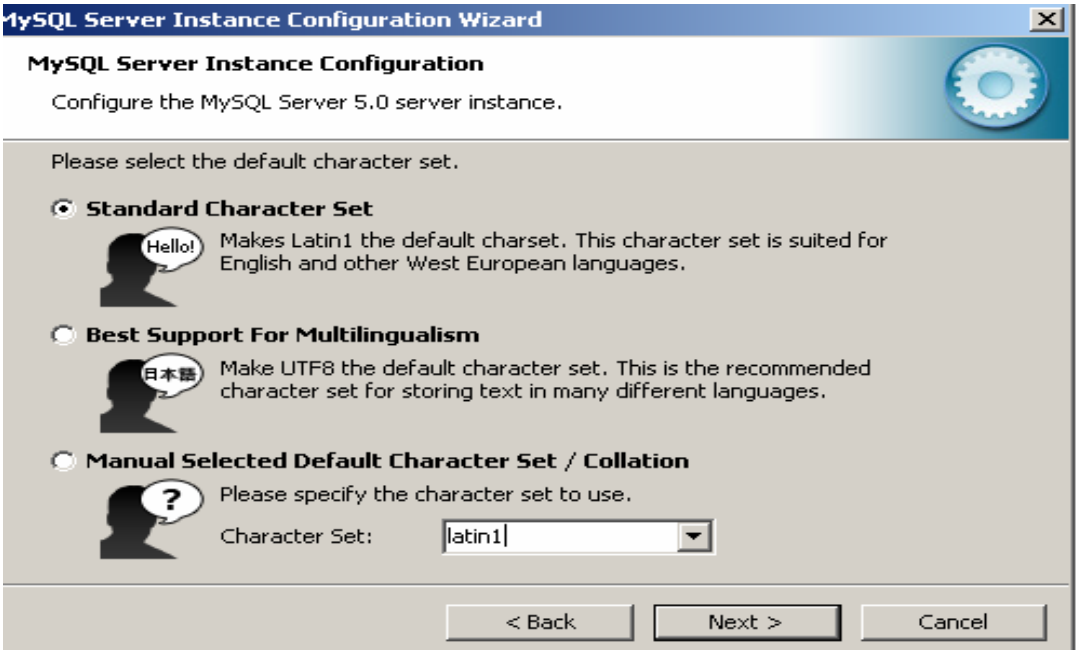


Figura: Ventana de selección de juego de caracteres.

A continuación, el asistente nos preguntará cómo queremos que se instale el servicio MySQL. Las dos opciones son como servicio de Windows o servicio desde la línea de comandos. Seleccionamos la primera opción.

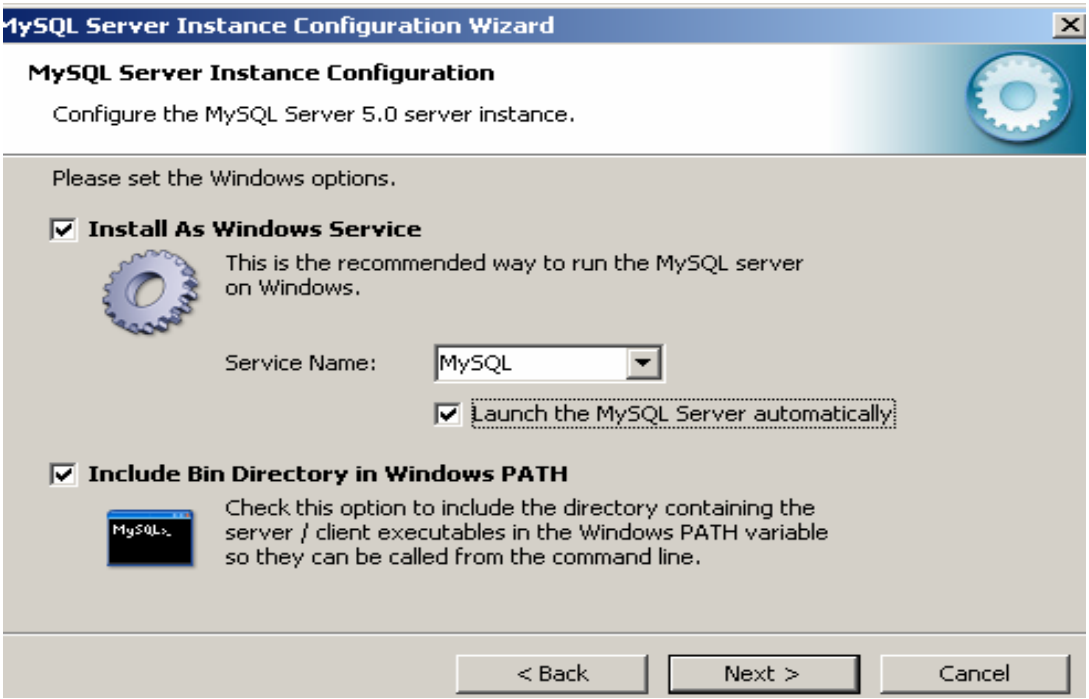


Figura: Ventana de selección de servicio MySQL.

El penúltimo paso de la configuración del servidor MySQL es el de especificar la clave del usuario root del servidor. Teclee la contraseña como muestra el siguiente cuadro y pulse el botón Next.

Figura: Ventana para introducir contraseña de root.

Por último pulse el botón Execute, y la configuración del servidor MySQL habrá finalizado con éxito.

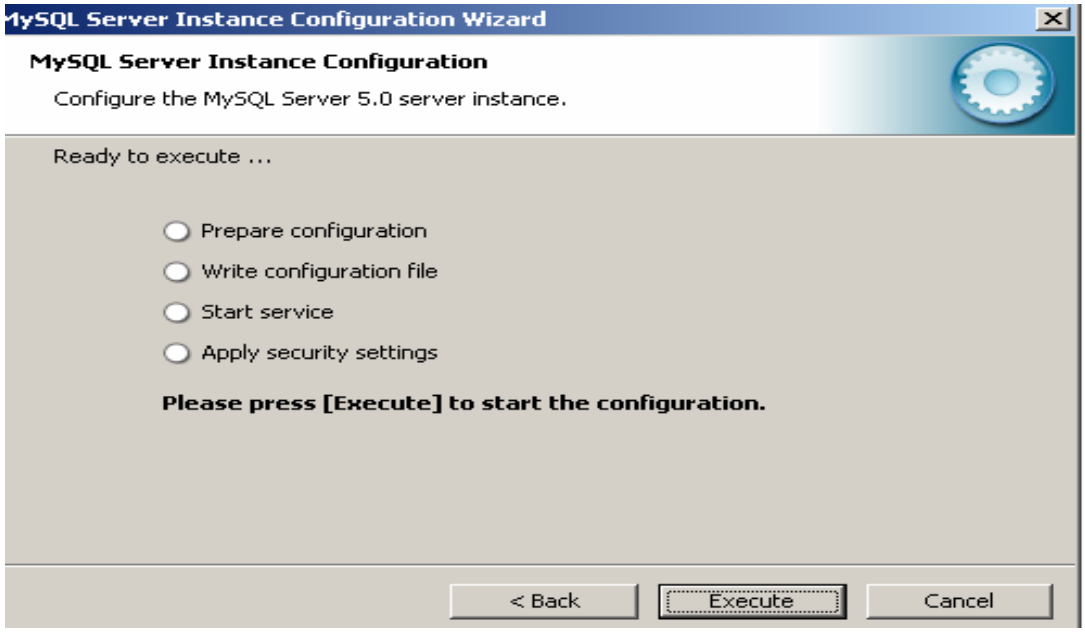


Figura: Ventana de lanzar la configuración del servidor MySQL.

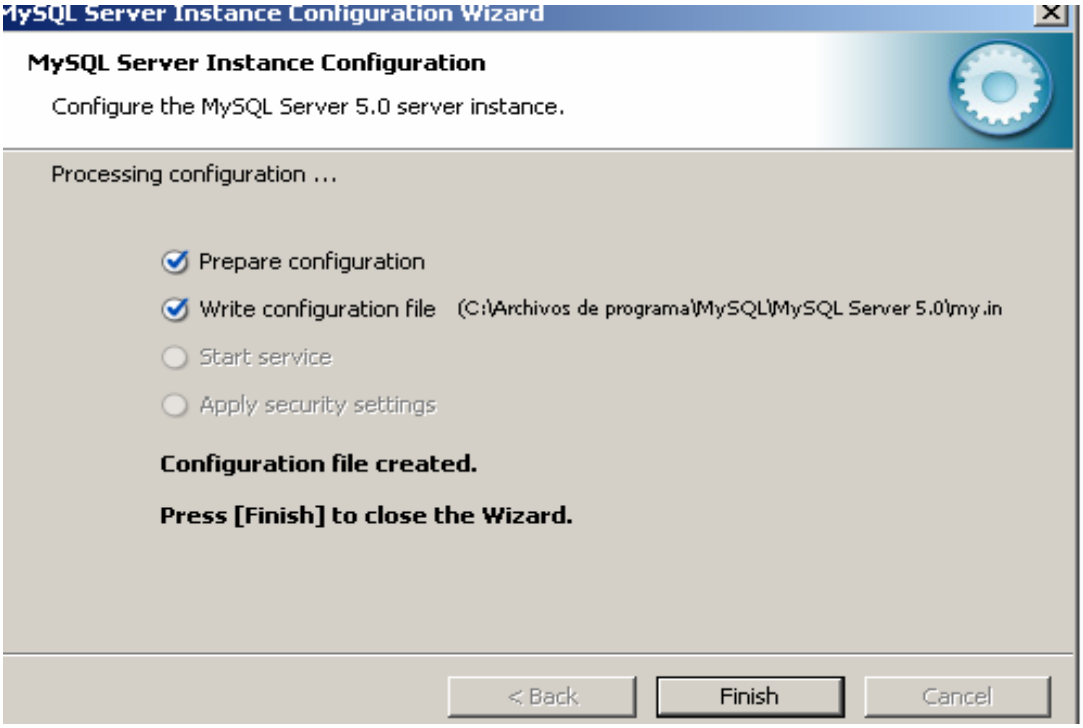


Figura: Ventana de confirmación de la configuración.

Una vez instalado y configurado el servidor MySQL, es el turno de instalar tanto el administrador como el cliente MySQL.

Paso 4: Instalar el administrador MySQL.

Para instalar el administrador MySQL, ejecute el fichero *mysql-administrator-1.1.6-win.ins* incluido en el CD-ROM que acompaña a esta memoria. Al ejecutar este fichero, se inicia el asistente de instalación:

Figura: Ventana de entrada al asistente de instalación del administrador MySQL.

Pulsamos el botón Next y aparece la ventana de conformidad con la licencia de MySQL.

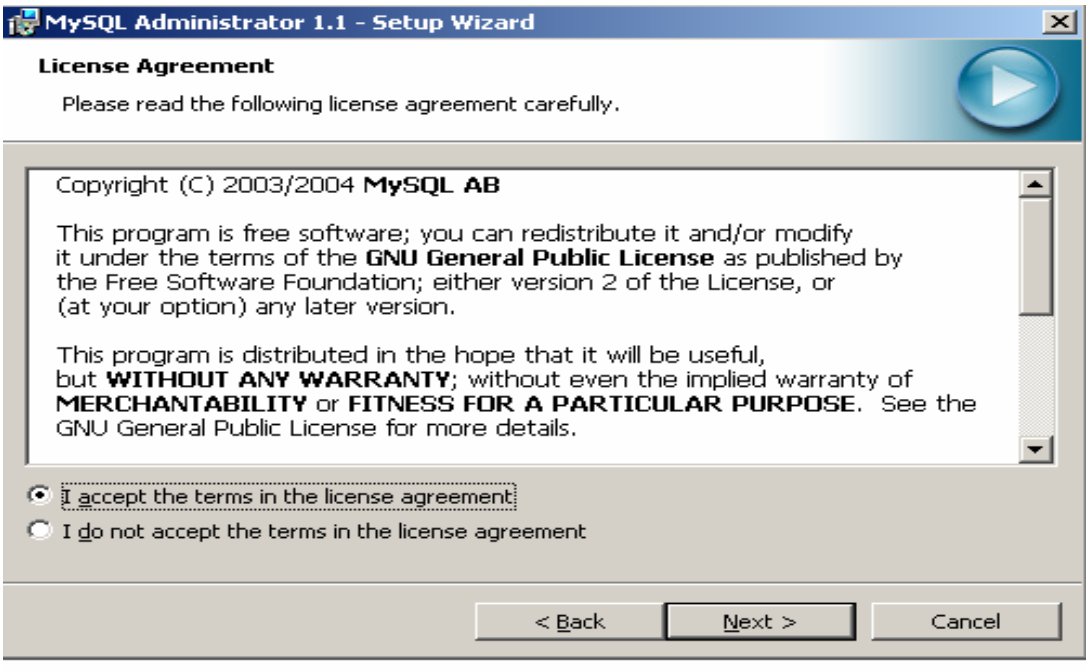


Figura: Ventana de conformidad con la licencia de MySQL.

Pinchamos la opción “I accept the terms in the license agreement” y a continuación pulsamosel botón Next. Acto seguido aparecerá la ventana de selección del directorio de instalación del administrador MySQL.

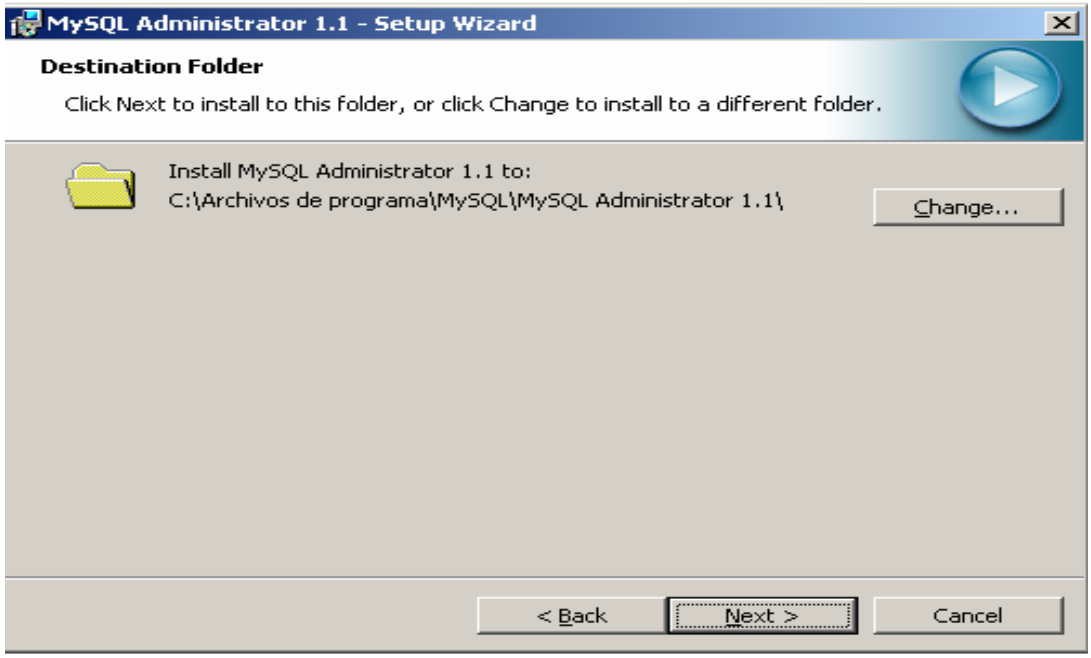


Figura: Ventana de selección del directorio de instalación.

Seleccionamos la ruta deseada y a continuación pulsamos el botón Next. Ahora es turno de seleccionar el tipo de instalación deseada. Elegimos la opción completa. Pulsamos en Next.

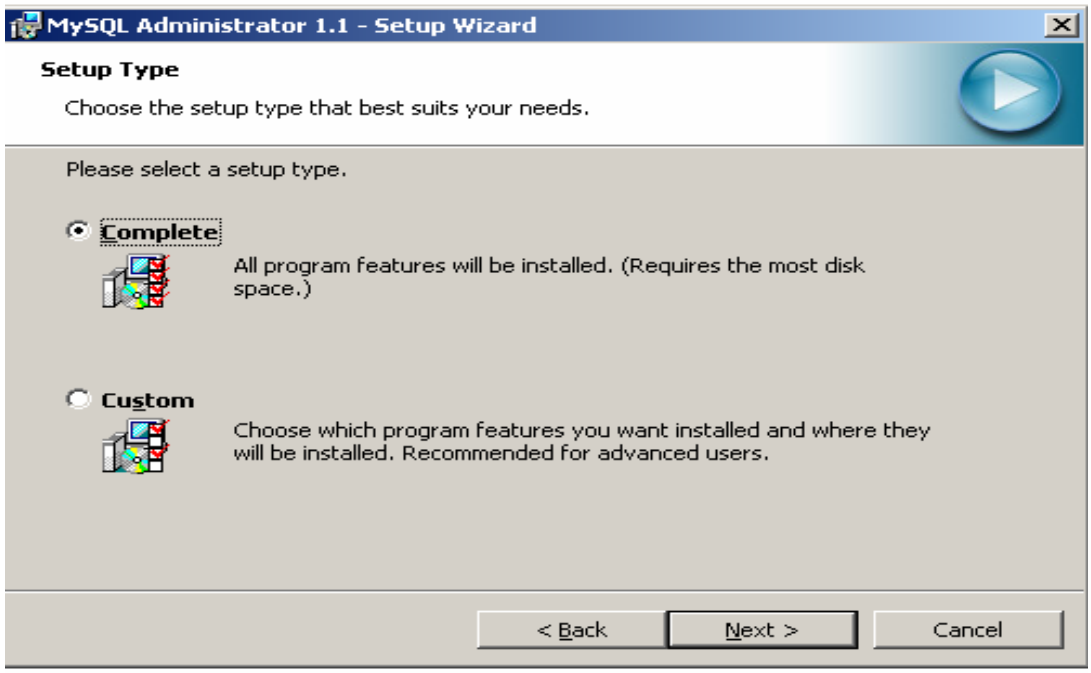


Figura: Ventana de selección del tipo de instalación.

Ahora el asistente nos informará del tipo de instalación y el directorio seleccionado. Si estamos conformes pulsamos el botón Install y la instalación comenzará.

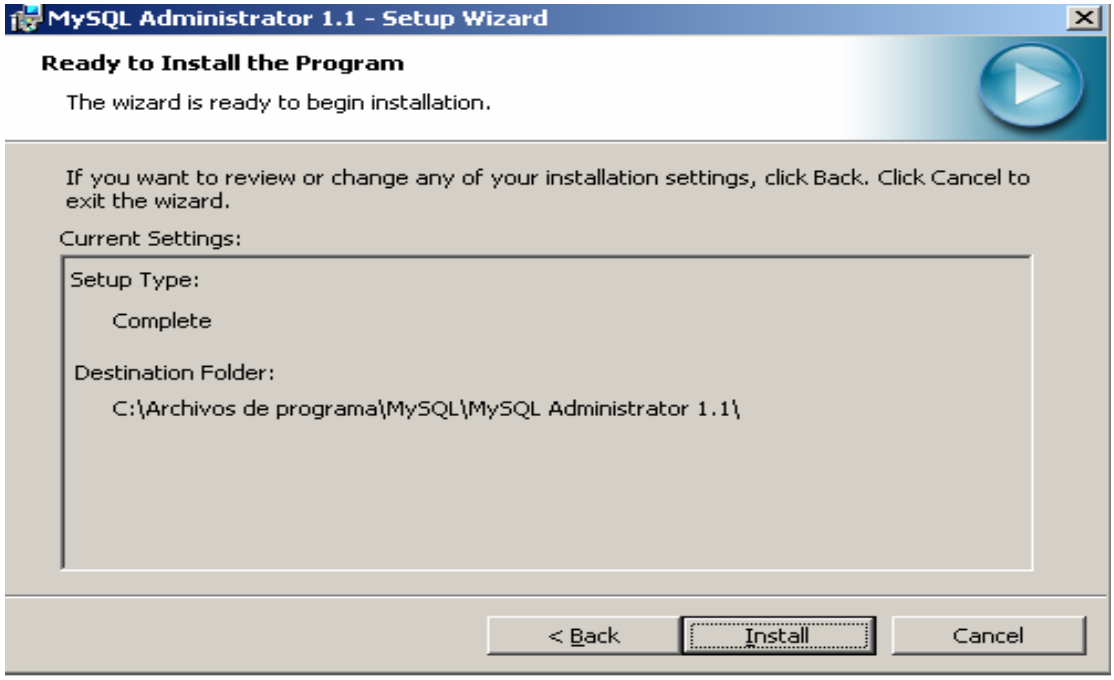


Figura: Ventana de confirmación de tipo y directorio de instalación.

Finalizada la instalación, el asistente nos mostrará la siguiente ventana:

Figura: Ventana de instalación satisfactoria.

Ya hemos instalado el administrador de la Base de Datos MySQL. Ahora es el turno del cliente.

Paso 5: Instalar el cliente MySQL.

El siguiente paso es el de instalar el cliente MySQL. A través de este cliente, el Administrador de la Base de Datos puede realizar consultas y modificaciones de los datos sin tener que arrancar la aplicación. Para iniciar la instalación, ejecute el fichero *mysql-query-browser-1.1.18-win.ins* adjuntado en el CD-ROM de la memoria del proyecto.

Los pasos que hay que realizar para instalar este cliente MySQL y las ventanas mostradas por este asistente son las mismas que en el del administrador. Mire el paso anterior para instalar este cliente MySQL.

Paso 6: Crear la Base de Datos de la aplicación.

Una vez que tenemos instalados los programas que forman la Base de Datos MySQL, el siguiente paso consiste en crear las tablas que formarán la base de datos de la aplicación ProcesoConsensov1.

Para ello hay dos alternativas. Una de ellas es crear las tablas manualmente desde el administrador de MySQL, y la otra más cómoda es lanzar el script *sql.cmd* desde la línea de comandos.

En este script hay que editar una serie de parámetros. Para ello abra el fichero con su editor preferido.

a) mysql_home.

Introduzca aquí el path del directorio bin de la instalación de MySQL.

set mysql_home=C:/Archivos de Programa/MySQL/MySQL Server 5.0/bin

b) script_tablas.

Introduzca el path del fichero tablas.sql que viene en el CD-ROM de la

memoria.

set script_tablas=C:/tablas.sql

c) fichero_salida.

Introduzca el path del fichero de salida del script.

set fichero_salida=D:/salida.out

d) user.

Introduzca el identificador del superusuario de la Base de Datos. Es el administrador de la Base de Datos.

set user=root

e) pass.

Por último edite la contraseña del usuario de arriba.

set pass=mysql

Una vez editados los cambios, guarde dichos cambios y lance el script desde la línea de comandos. Para ello ejecute una ventana de la línea de comandos:

Menú Inicio → Ejecutar → cmd

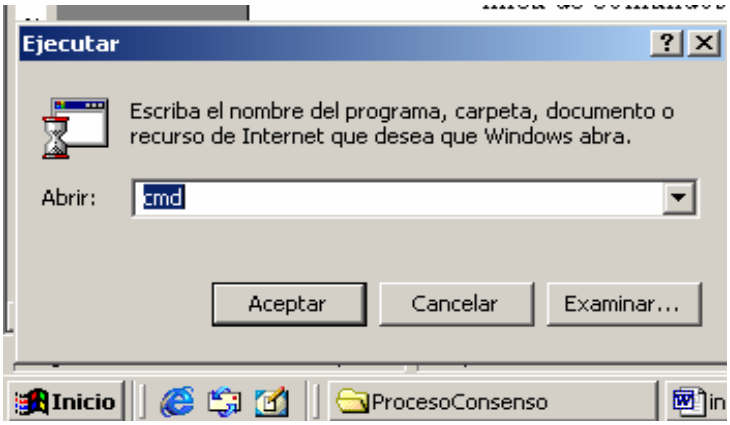


Figura: Ejecutar aplicación en inicio.

En esta ventana, dirijase al directorio donde se encuentra el fichero sql.cmd y ejecute el comando:

sql.cmd

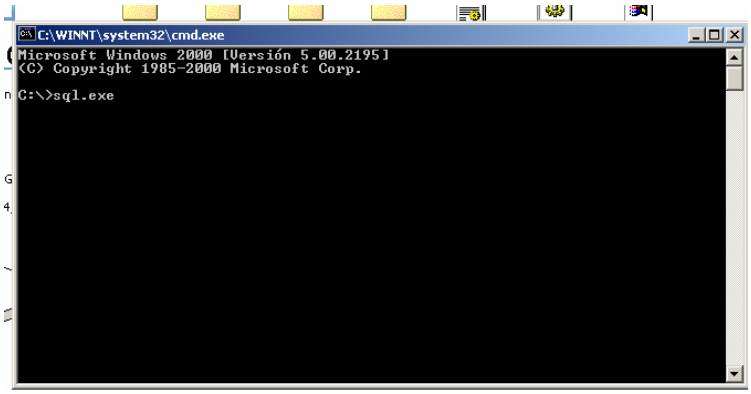


Figura: Ejecución del script que crea la Base de Datos.

Paso 7: Instalar la aplicación.

Una vez instalados todos los programas que forman parte del núcleo de nuestra aplicación, es el turno de instalar ProcesoConsensov1.

Para ello, descomprima el fichero ProcesoConsensov1.zip incluido en el CD-ROM que acompaña a esta memoria en el directorio que desee.

A continuación arranque el entorno de desarrollo Netbeans y abra el proyecto de desarrollo ProcesoConsensov1.

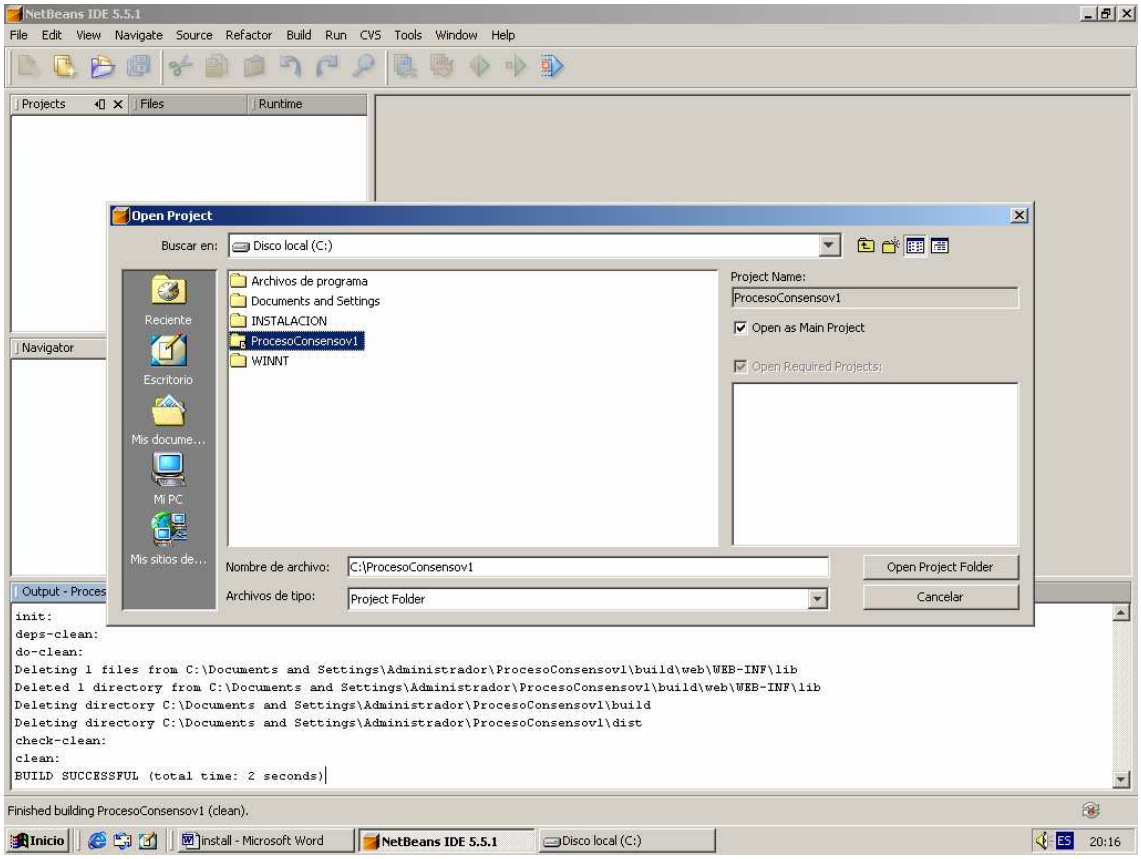


Figura: Añadir el directorio ProcesoConsensov1.

Una vez abierto el proyecto, hay que ponerle una referencia a la librería mysql. Esta referencia se realiza de la siguiente manera:

Pinche con el botón derecho del ratón sobre el proyecto ProcesoConsensov1.

A continuación pinche en Properties.

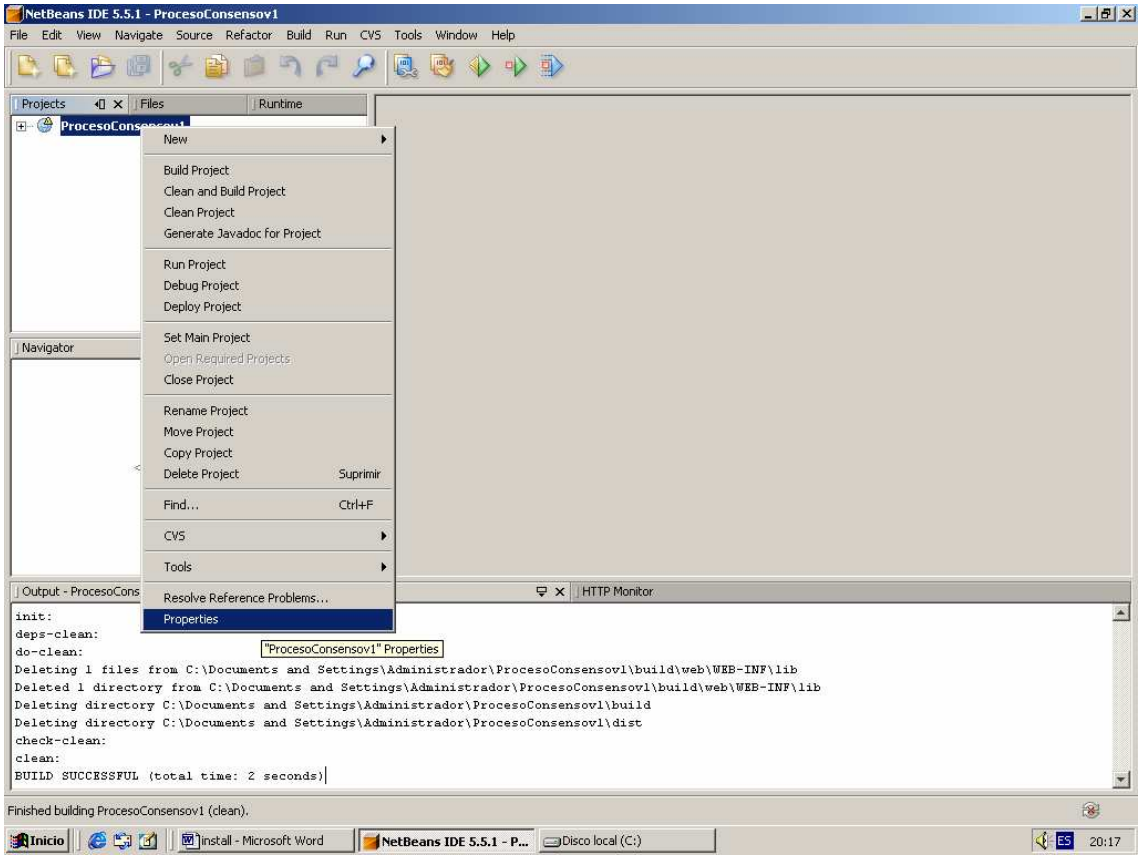


Figura: Menú Properties del proyecto ProcesoConsensov1

Una vez esté en Properties, vaya a la sección Libraries y pulse el botón Add JAR/Folder. Seleccione el fichero mysql.jar incluido en el directorio lib del proyecto ProcesoConsensov1 y dele a OK.

El proyecto se construirá automáticamente. Ya está instalada la aplicación.

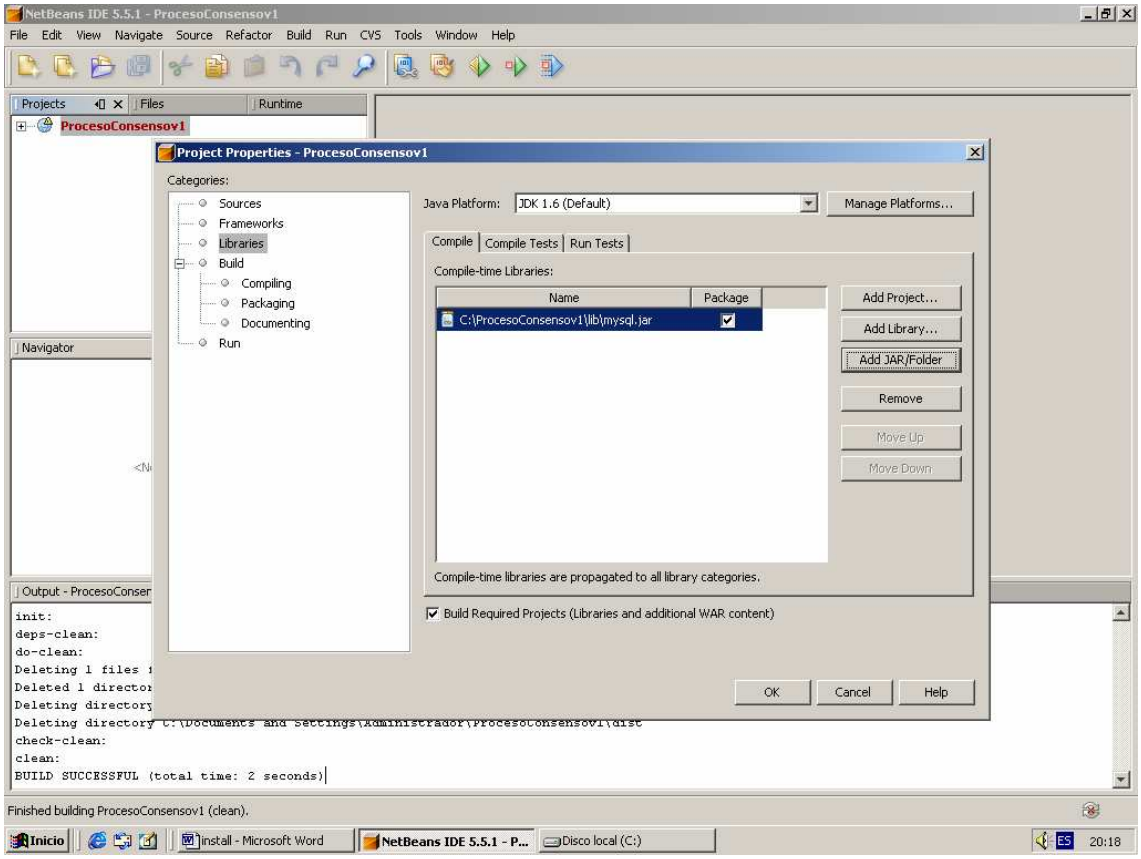


Figura:Añadir jar al proyecto ProcesoConsensov1

ANEXO II:
MANUAL DE USUARIO.

1 Introducción.

El manual de usuario es una documento técnico que permite dar asistencia a los usuarios de un sistema. En nuestro caso, nuestro manual de usuario explicará de manera pormenorizada cómo realizar las distintas funcionalidades de nuestro sistema. Además, informa de los posibles casos de error que puede cometer el usuario y como resolverlos.

Este manual de usuario está organizado en tres partes, una de ellas por cada uno de los tipos de usuario del sistema. Tendremos un bloque para los expertos, otro para los administradores y finalmente otro para los administradores de la base de datos.

Como hemos explicado en el párrafo anterior, tenemos tres tipos de usuarios del sistema:

- Experto → Es aquel usuario que da sus opiniones sobre los problemas. No tiene funciones de administración de problemas y de base de datos.

- Administrador → Es aquel usuario que define los problemas y asigna expertos a su discusión. Tiene también funciones de consulta del estado del problema y creación de nuevos expertos.

- Administrador de la Base de Datos → Es aquel usuario del sistema que interactúa de manera directa con la Base de Datos. Es el único usuario que puede modificar los datos almacenados en la Base de Datos.

Antes de realizar una visita por la aplicación a través de este manual de usuario, es conveniente que el usuario tenga claro algunos aspectos:

- Toda la aplicación está optimizada para su visualización en un navegador Mozilla Firefox con una resolución de 1280x800 píxeles. Si se utiliza otro navegador o una resolución distinta a la recomendada, pueden producirse fallos en la visualización de los elementos de la pantalla.

- La aplicación tiene dos tipos de usuarios distintos: Expertos y Administradores, amén del Administrador global del sistema o Gestor de la Base de Datos. Para

ello hemos dividido este manual de usuario en tres secciones: Manual de usuario para el Administrador de la Base de Datos, Manual de usuario para el usuario Administrador; y por último, Manual de usuario para el usuario Experto.

2 Manual de usuario del experto.

En esta sección, vamos a explicar el funcionamiento de la aplicación desde el punto de vista del Experto. Los expertos de nuestra aplicación, dan su opinión acerca de los problemas en los que se encuentran asignados. El Experto no tiene más funcionalidad que la que acabamos de explicar.

Para iniciar una sesión del Experto en el sistema, debe de ejecutar en el navegador el fichero *index_exp.html*. Es la página de inicio del Experto en el sistema.

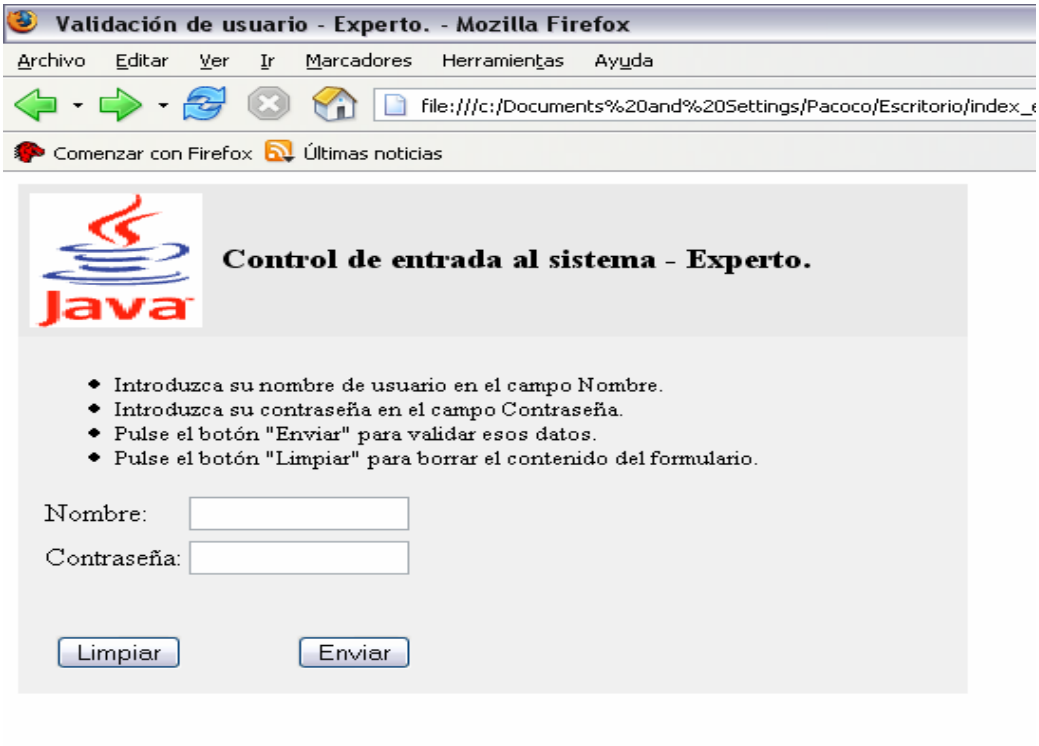


Figura: Página de inicio de Experto (*index_exp.html*)

2.1 Entrada al sistema.

Para poder autentificarse el Experto en el sistema, tiene que introducir en el formulario de index_exp.html su identificador de usuario y contraseña que tiene en el sistema.

Figura: Página de inicio de sesión para el Experto.

Una vez introducidos ambos datos (identificador de experto y contraseña), tiene dos opciones:

- Limpiar la pantalla → Si los datos introducidos no son correctos, pueden ser modificados o borrados del formulario antes de ser enviados. Para ello pulse el botón *Limpiar*.

- Enviar los datos → Si los datos introducidos cree que son válidos, envíelos al sistema pulsando el botón *Enviar.*

Si los datos son correctos, el sistema dará la bienvenida al sistema al Experto, y mostrará el listado de problemas en los que participa.

También se pueden producir las siguientes situaciones de error. Si el identificador de usuario no existe en la Base de Datos, o alguno de los datos introducidos no son válidos, la aplicación mostrará la siguiente ventana:

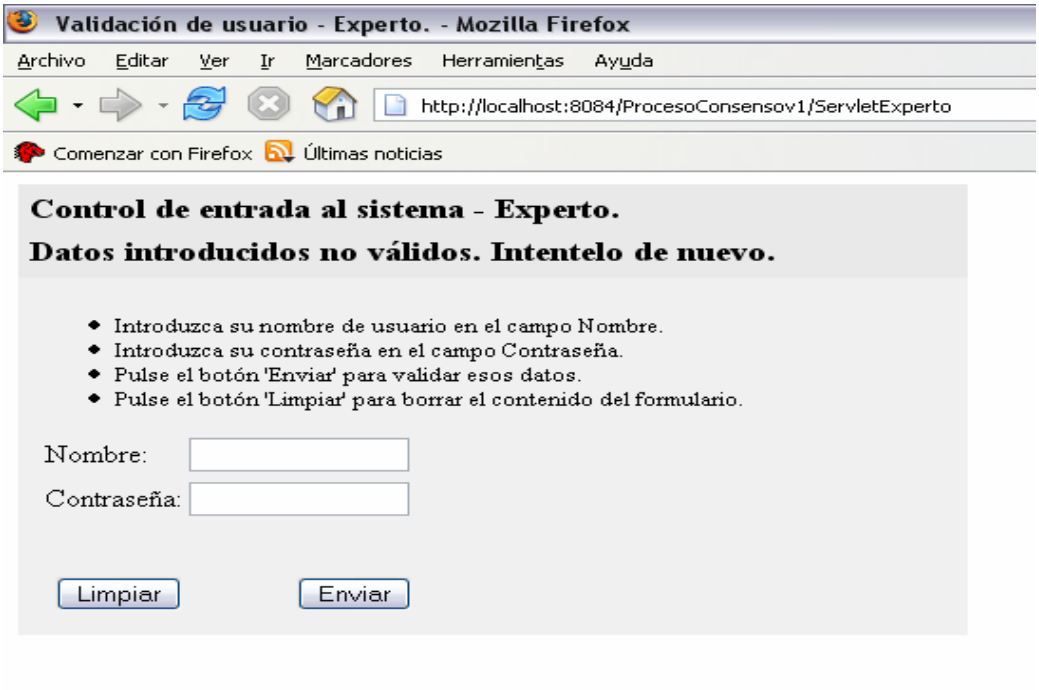


Figura: Página de inicio de sesión del Experto (2º intento o posterior).

Finalmente, si el usuario Experto existe en la Base de Datos, y sus datos introducidos en el sistema son válidos, podrá acceder a su página principal de usuario. En ella se mostrará el listado de problemas en los que participa el Experto:

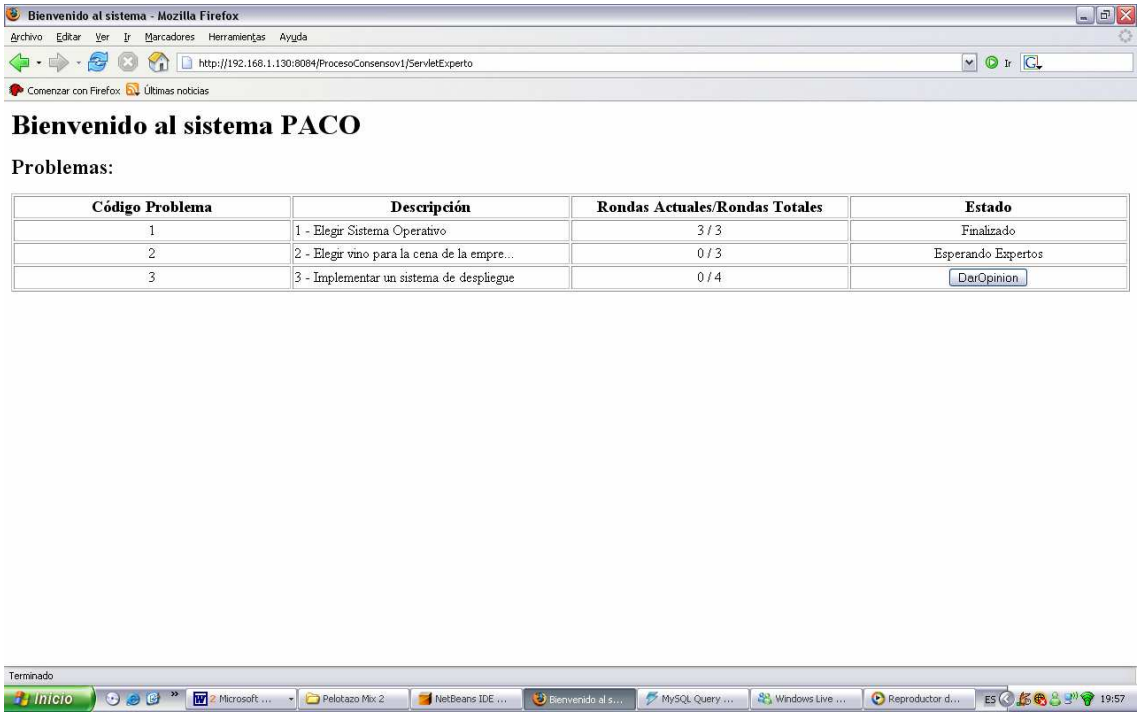


Figura: Listado de problemas asignados del Experto paco.

2.2 Seleccionar un problema.

En la pantalla de problemas asignados al Experto (Imagen anterior), tenemos el listado de problemas asignados al Experto. Éstos pueden tener tres posibles estados:

- Finalizado → El PC ya ha finalizado. El Experto no puede opinar ya sobre el problema en cuestión.

- Esperando Expertos → El Experto ya ha dado su opinión para la ronda actual. El sistema espera aún la opinión de alguno de los Expertos asignados al problema. El Experto debe esperar a que el resto de Expertos den sus opiniones para poder continuar el PC.

- Dar Opinión → El sistema está esperando la opinión del Experto para el problema en concreto y para la ronda actual.

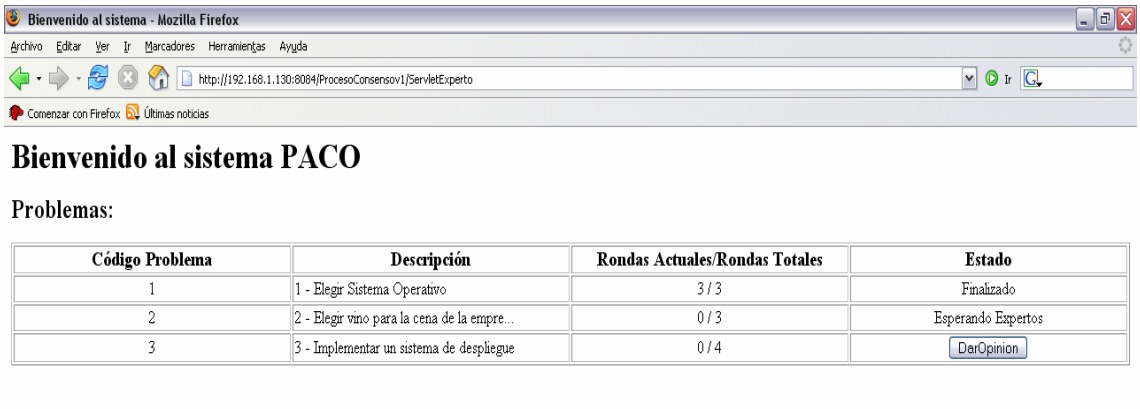


Figura: Listado de problemas y sus posibles estados.

2.3 Dar opinión de un problema.

Esta es la funcionalidad principal del Experto en el sistema. A partir del listado de problemas asignados al Experto, éste debe seleccionar aquel del que desea dar su opinión.

Para que el Experto pueda seleccionar un problema para dar su opinión, éste ha tener el estado Dar Opinión. Si no tiene este estado, el Experto no puede dar su opinión, bien porque haya finalizado o porque el sistema aún esté esperando opiniones del resto de expertos.

Para dar la opinión de un problema, el Experto pulsará el botón Dar Opinión del problema deseado. Tras pulsar dicho botón, el sistema mostrará la siguiente ventana:

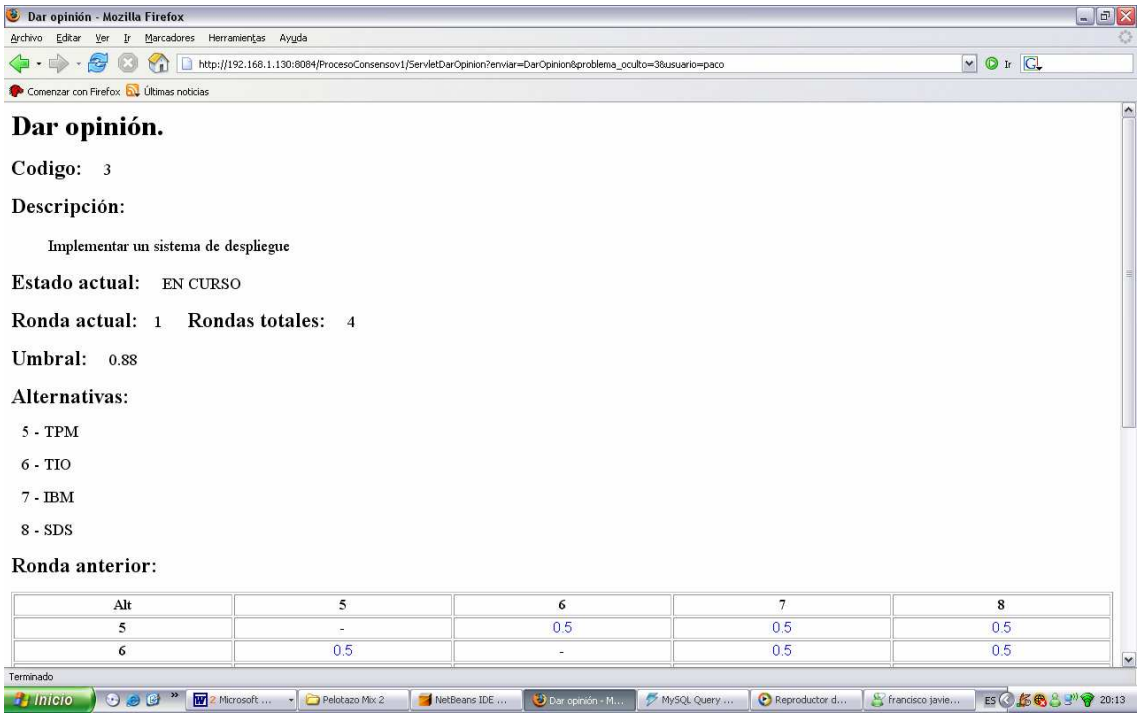


Figura: Página principal del problema a dar opinión (Parte 1).

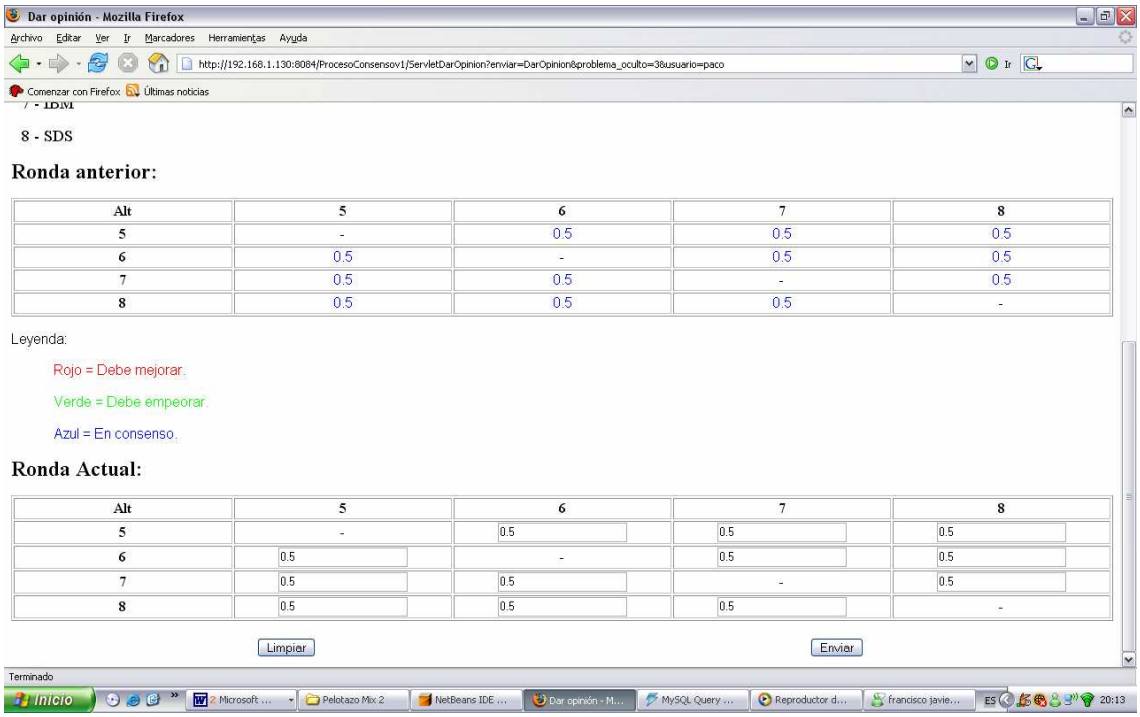


Figura: Página principal del problema a dar opinión (Parte 2).

En esta página, tenemos tres secciones bien diferenciadas:

- Parte 1 → Descripción del problema, valor de sus parámetros básicos, alternativas de solución y estado actual.

- Parte 2 → Ronda anterior. Indica el valor de las preferencias suministradas por el Experto en la ronda anterior, así como las recomendaciones de dicha ronda. De esta manera, se facilita la labor del Experto. En el caso de que ese par de alternativas esté en posiciones de consenso, el valor de la ronda anterior aparecerá de color azul. En caso de que haya que incrementar aparecerá en color rojo y por último, si debe empeorar aparecerá en color verde.

- Parte 3 → Ronda actual. Es una tabla en la que el Experto introduce sus preferencias de la ronda actual.

Una vez haya introducido sus preferencias el Experto, pulse el botón Enviar para enviar sus preferencias al sistema.

Si alguna de las preferencias no es un numérica o es vacía, el sistema mostrará de nuevo la ventana con el mensaje de error:

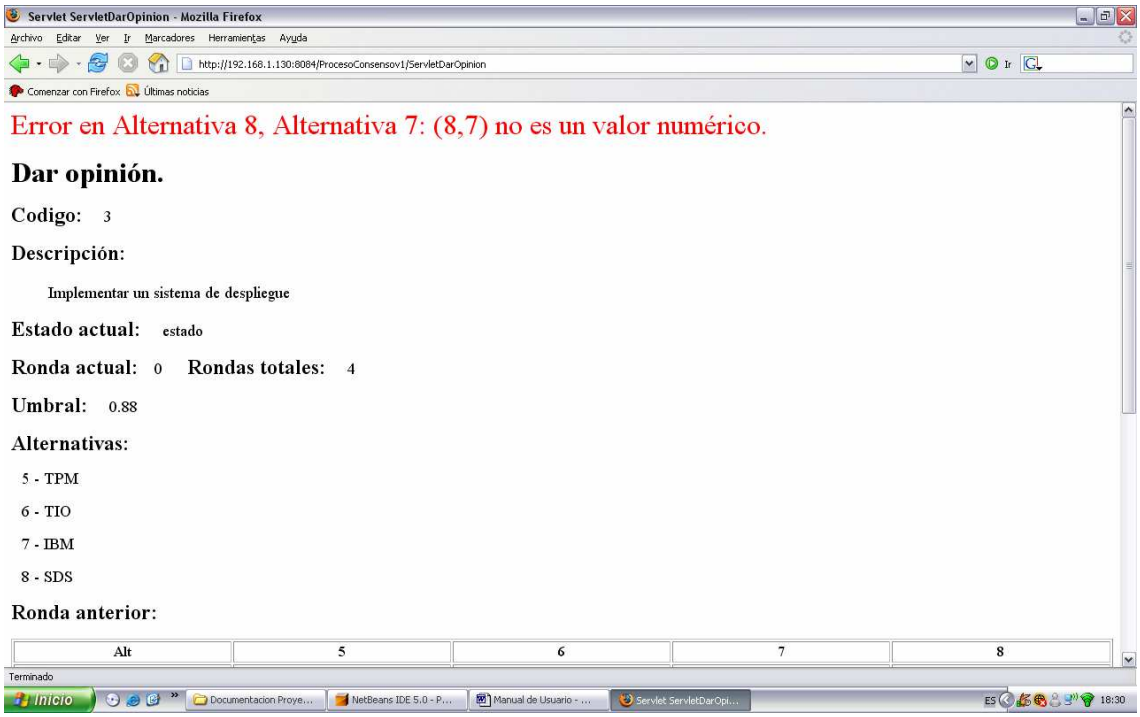


Figura: Error al introducir el valor de una preferencia.

3 Manual de usuario del Administrador.

En esta sección, vamos a explicar el funcionamiento de la aplicación desde el punto de vista del Administrador. Los administradores de nuestra aplicación, crean los problemas de toma de decisiones, asignan alternativas al problema y eligen el conjunto de expertos que discuten sobre el problema. Además, pueden consultar el estado actual e histórico del Proceso de Consenso.

Para iniciar una sesión de Administrador en el sistema, debe de ejecutar en el navegador el fichero *index_adm.html*. Es la página de inicio del Administrador en el sistema.

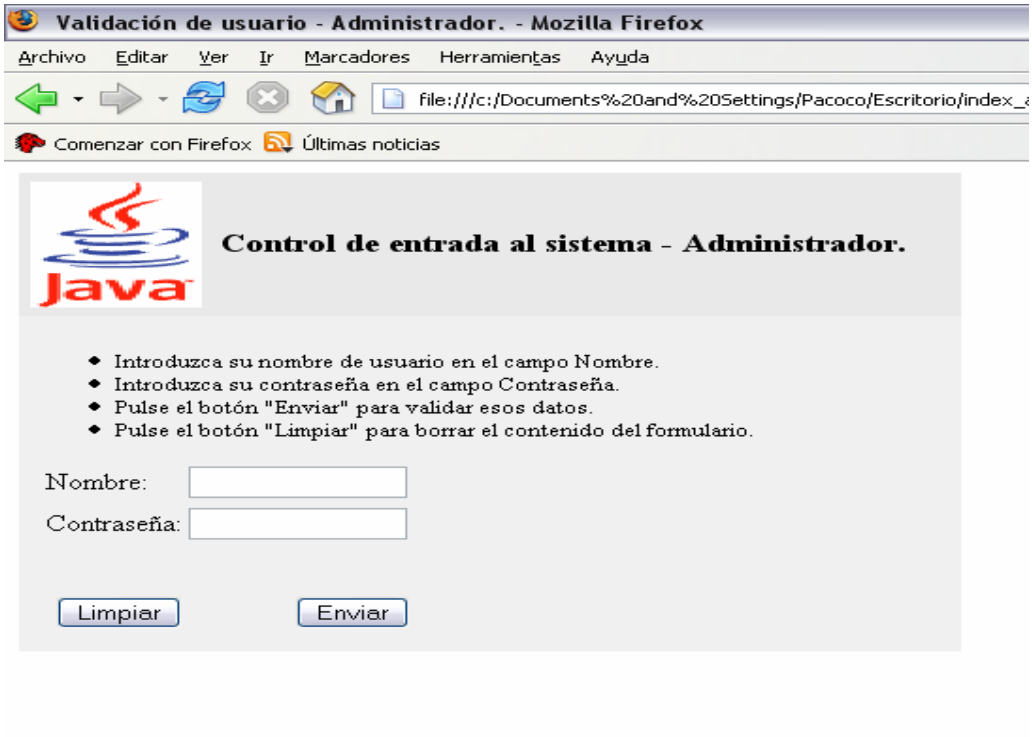


Figura: Página de inicio de Administrador (index_adm.html).

3.1 Entrada al sistema.

Para poder autentificarse el Administrador en el sistema, tiene que introducir en el formulario de index_adm.html su identificador de usuario y contraseña que tiene en el sistema.

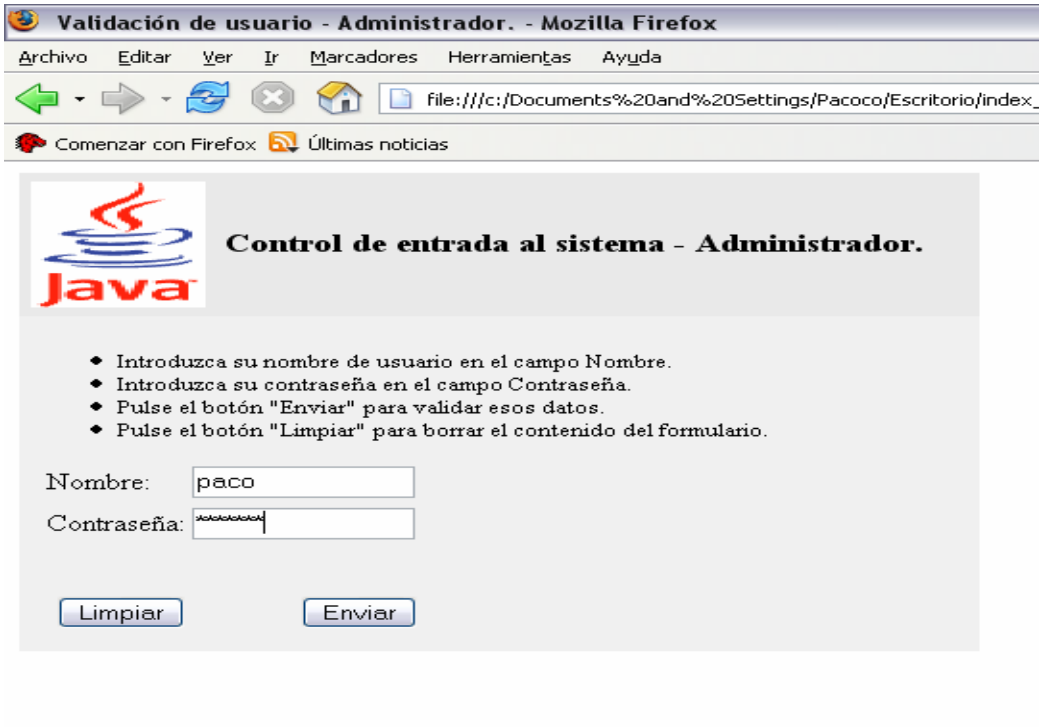


Figura: Página de inicio de sesión para el Administrador.

Una vez introducidos ambos datos (identificador de administrador y contraseña), tiene dos opciones:

- Limpiar la pantalla → Si los datos introducidos no son correctos, pueden ser modificados o borrados del formulario antes de ser enviados. Para ello pulse el botón *Limpiar*.

- Enviar los datos → Si los datos introducidos cree que son válidos, envíelos al sistema pulsando el botón *Enviar.*

Si los datos son correctos, el sistema dará la bienvenida al sistema al Administrador, y mostrará el menú principal del Administrador.

También se pueden producir las siguientes situaciones de error. Si el identificador de Administrador no existe en la Base de Datos, o alguno de los datos introducidos no son válidos, la aplicación mostrará la siguiente ventana:

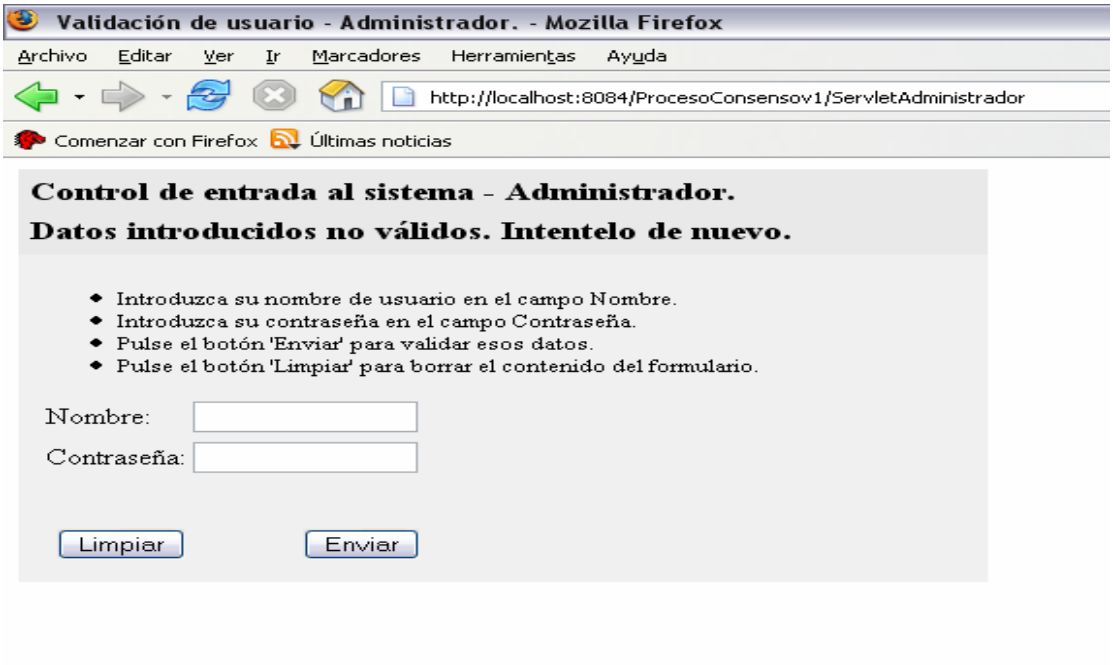


Figura: Página de inicio de sesión del Experto (2º intento o posterior).

Finalmente, si el usuario Administrador existe en la Base de Datos, y sus datos introducidos en el sistema son válidos, podrá acceder a su página principal de usuario. En ella se mostrará el siguiente menú de opciones:

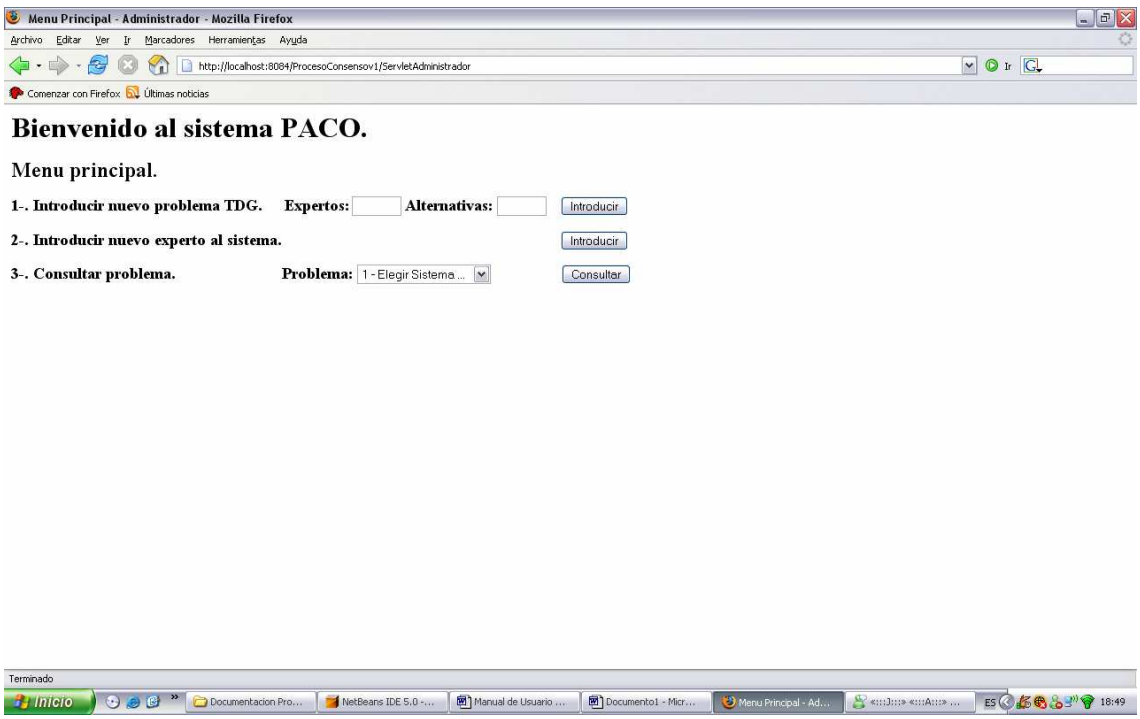


Figura: Menú principal del Administrador paco.

En este menú de principal tenemos las siguientes opciones:

- Crear un nuevo problema de Toma de Decisiones.

- Crear un nuevo experto del sistema.

- Consultar el estado del problema seleccionado.

3.2 Crear un nuevo problema de TDG.

La función más importante que desempeña un Administrador en nuestra aplicación es sin duda la de crear un nuevo problema de Toma de Decisiones en Grupo. Esta función consta de las siguientes subfunciones:

- Describir el problema y especificar sus parámetros.

- Describir las alternativas del problema.

- Asignar expertos al PC del problema.

Todas estas funciones son llevadas a cabo en la opción 1 del menú principal del Administrador.

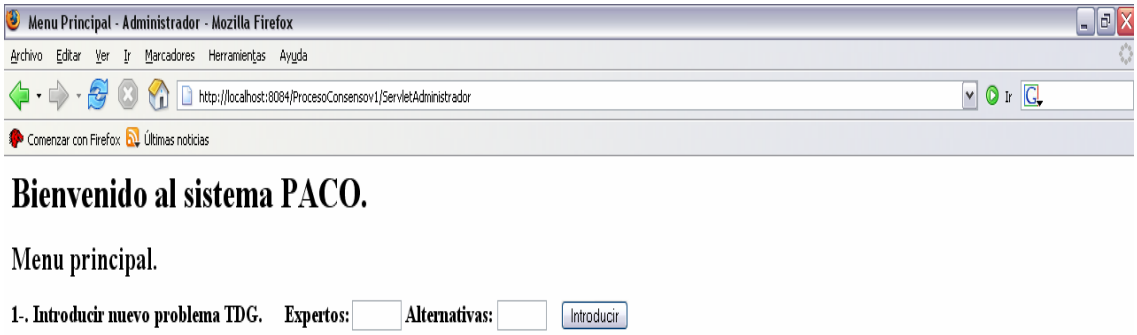


Figura: Crear un nuevo problema de TDG.

En este formulario, hay que introducir el número de expertos y de alternativas que tendrá el PC. Ambos datos deben ser valores enteros positivos. Si alguno de los valores es omitido, o no es correcto, el sistema devolverá de nuevo la pantalla principal del Administrador junto al mensaje de error:

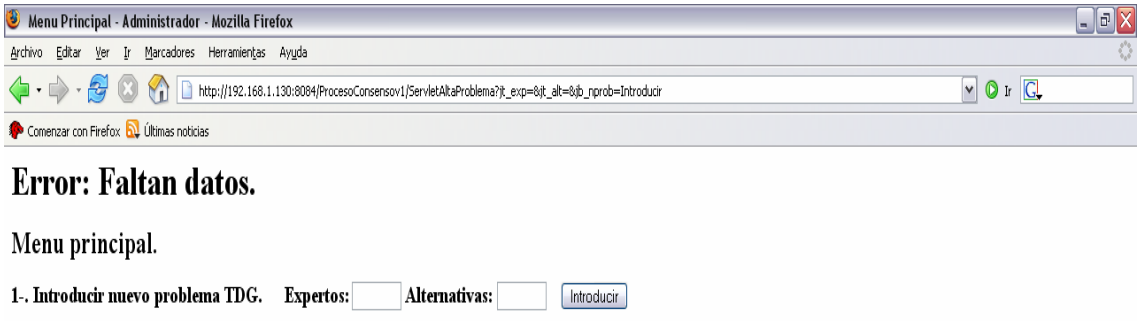


Figura: Datos del problema no son correctos.

Una vez introducidos los valores correctamente, pulse el botón Introducir para iniciar la creación del problema.

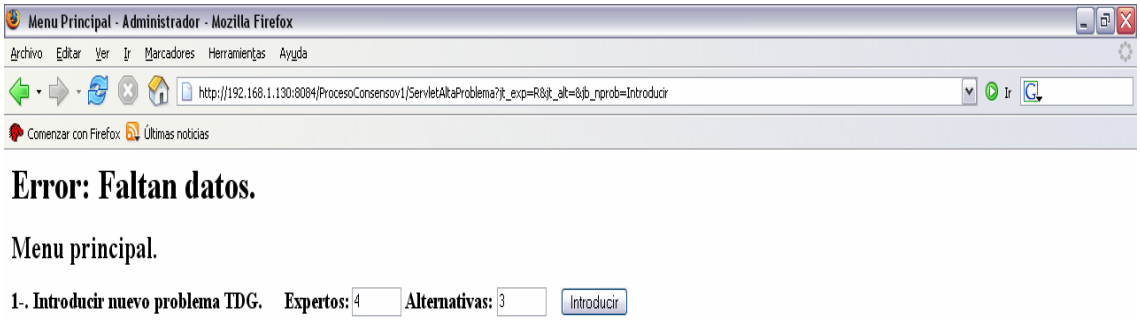


Figura: Ejemplo de datos correctos en la creación de un problema de TDG.

Una vez introducidos los datos y validados por el sistema, este devolverá la siguiente ventana:

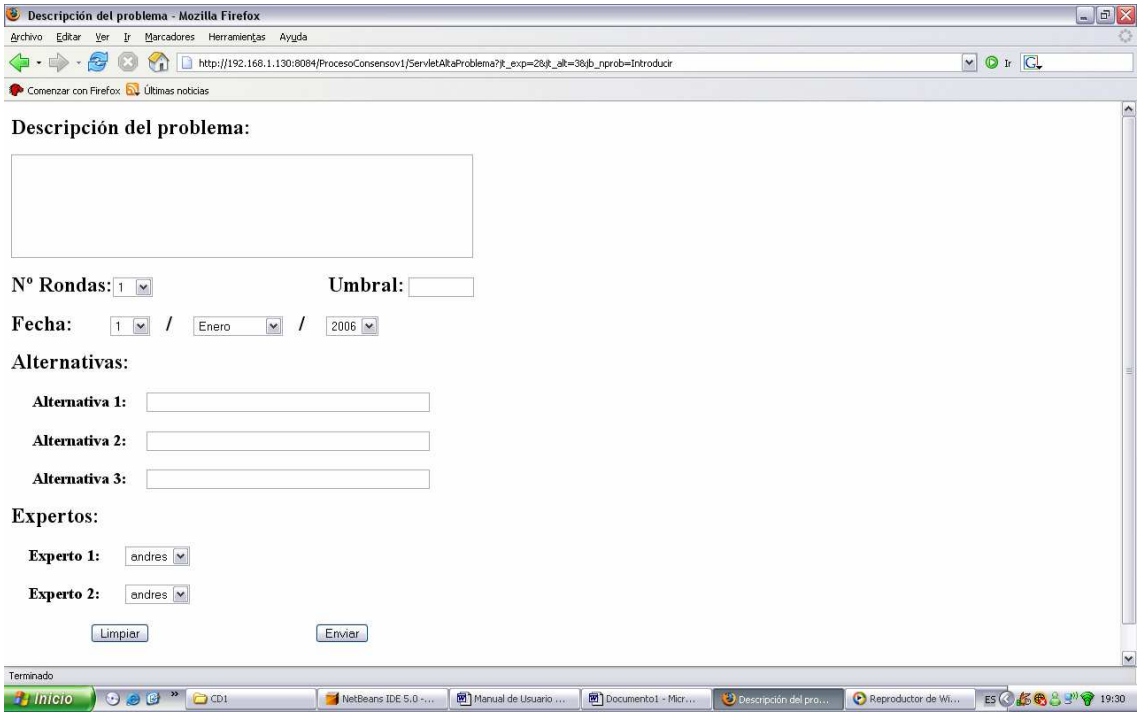


Figura: Pantalla principal de creación de un problema.

En esta pantalla tenemos los siguientes datos:

- Descripción del problema → Introduzca aquí una breve descripción del problema, su contexto, etc...Si no especifica la descripción del problema, la aplicación volverá a mostrar la ventana principal con el mensaje de error:

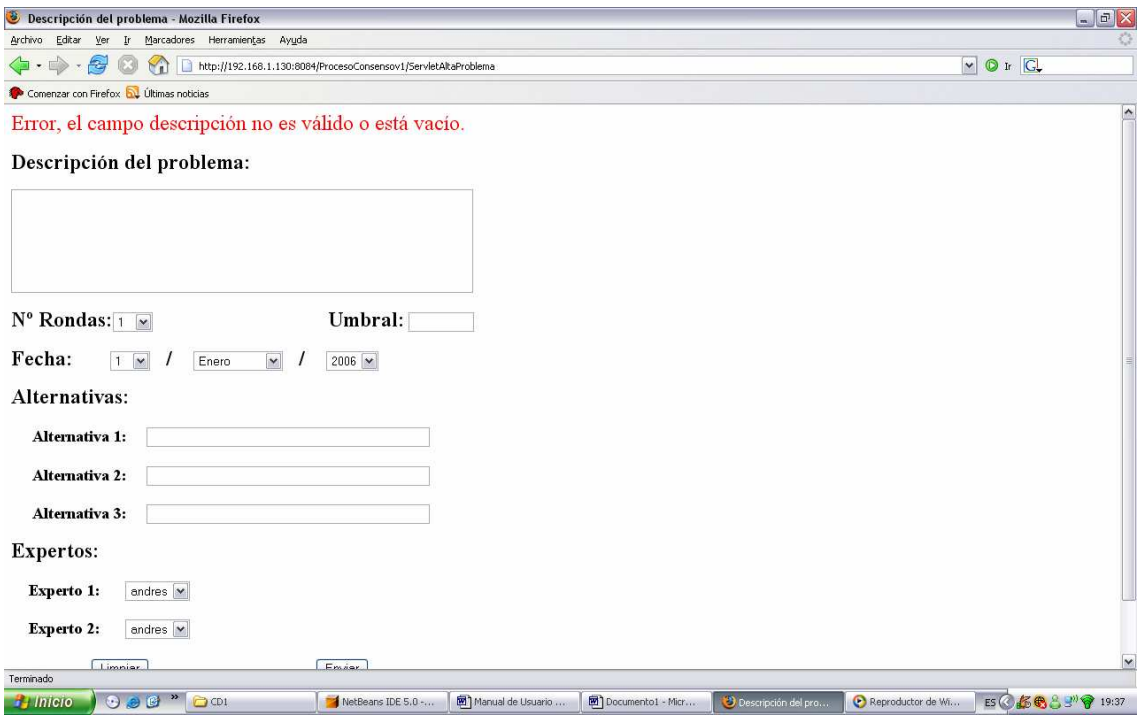


Figura: No ha especificado una descripción del problema.

- Nº Rondas → Seleccione el número máximo de rondas que puede durar como mucho el Proceso de Consenso.

- Umbral → Especifique el umbral de consenso a alcanzar. Debe ser un valor comprendido entre 0 y 1. Si el valor introducido es vacío o fuera del rango, el sistema devolverá de nuevo la ventana de crear problema indicando el error cometido:

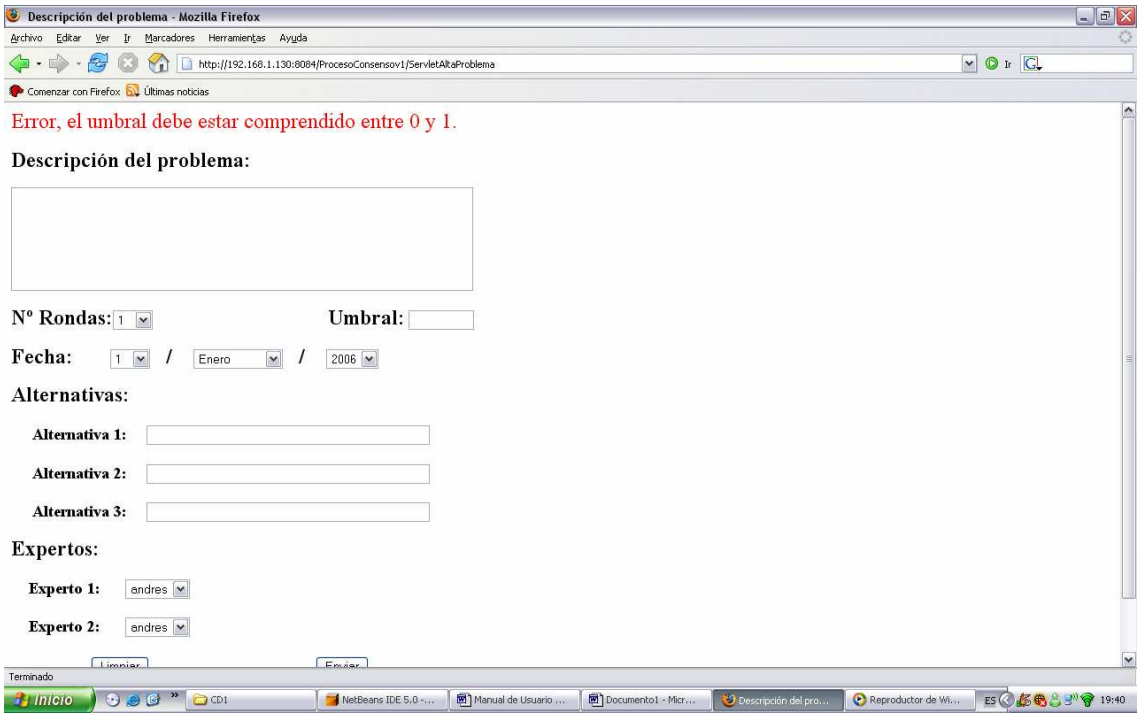


Figura: El umbral de consenso especificado no está comprendido entre 0 y 1.

- Fecha → Seleccione la fecha de creación del problema de TDG.

- Alternativas → Escriba la descripción de cada una de las alternativas del problema. Si alguna de las alternativas es vacía, el sistema volverá a mostrar la ventana de crear problema junto al mensaje de error:

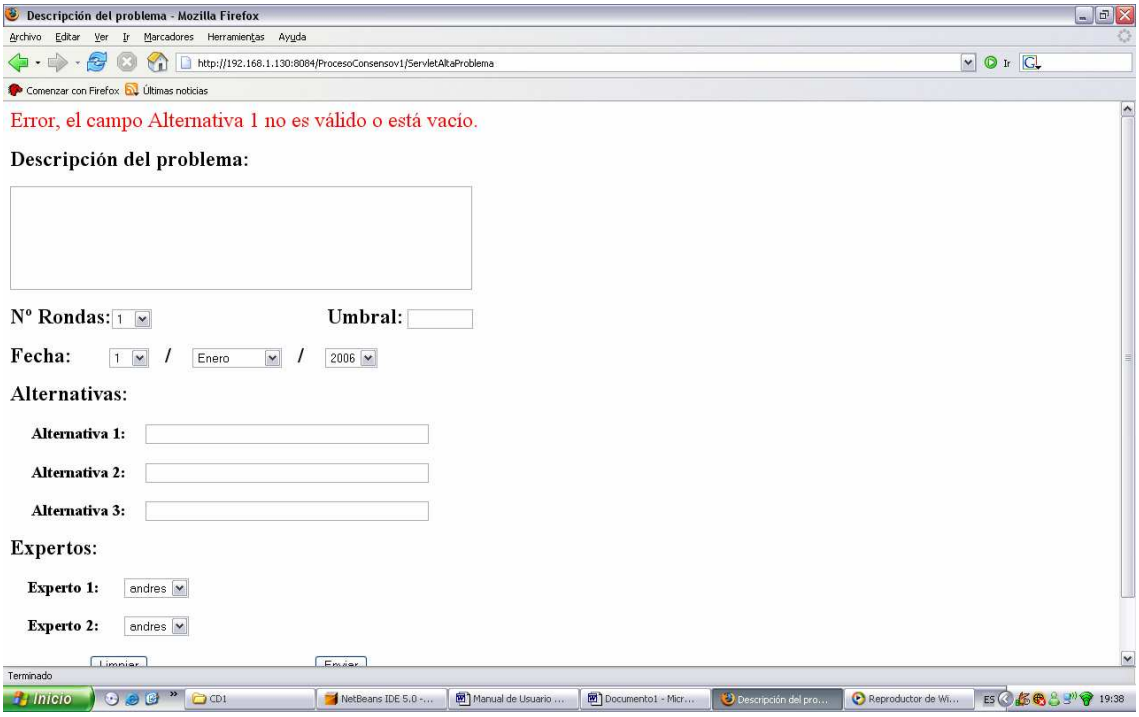


Figura: La alternativa 1 no ha sido descrita.

- Expertos → Seleccione el conjunto de expertos asignados al problema.

Una vez estén todos los datos introducidos, pulse el botón Enviar, y el sistema tras validarlos introducirá el nuevo problema en la Base de Datos. A partir de este momento, este nuevo problema será accesible tanto para el Administrador como para los Expertos asignados.

Veamos un ejemplo de problema correcto:

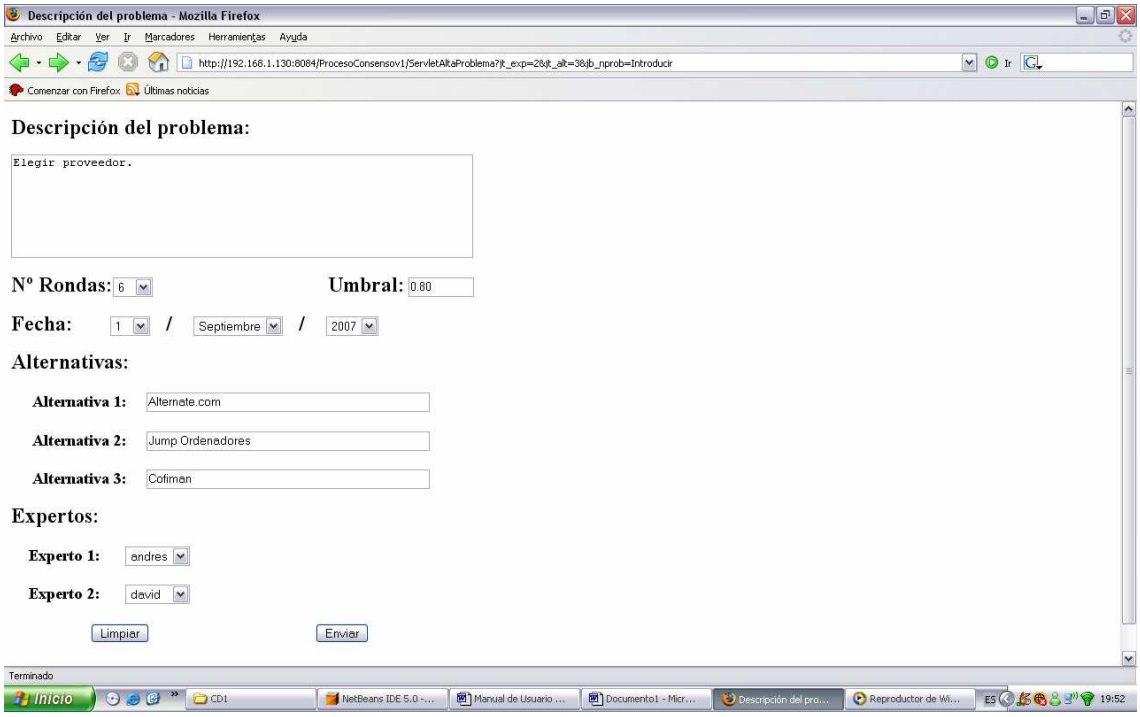


Figura: Ejemplo de problema válido.

Este problema es accesible para el Administrador nada más ser almacenado:

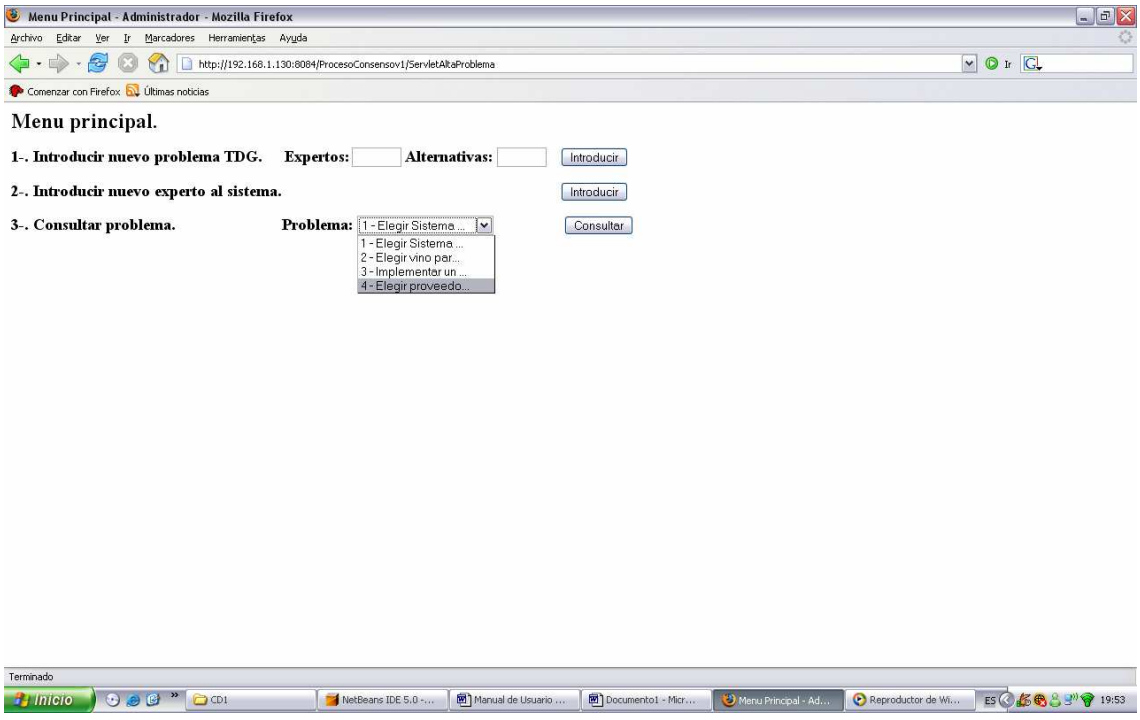


Figura: El Administrador que ha creado el problema seleccionándolo.

Y además, también es accesible para los Expertos asignados a dicho problema. El resto de Expertos no asignados al problema, no tendrán conocimiento de dicho problema.

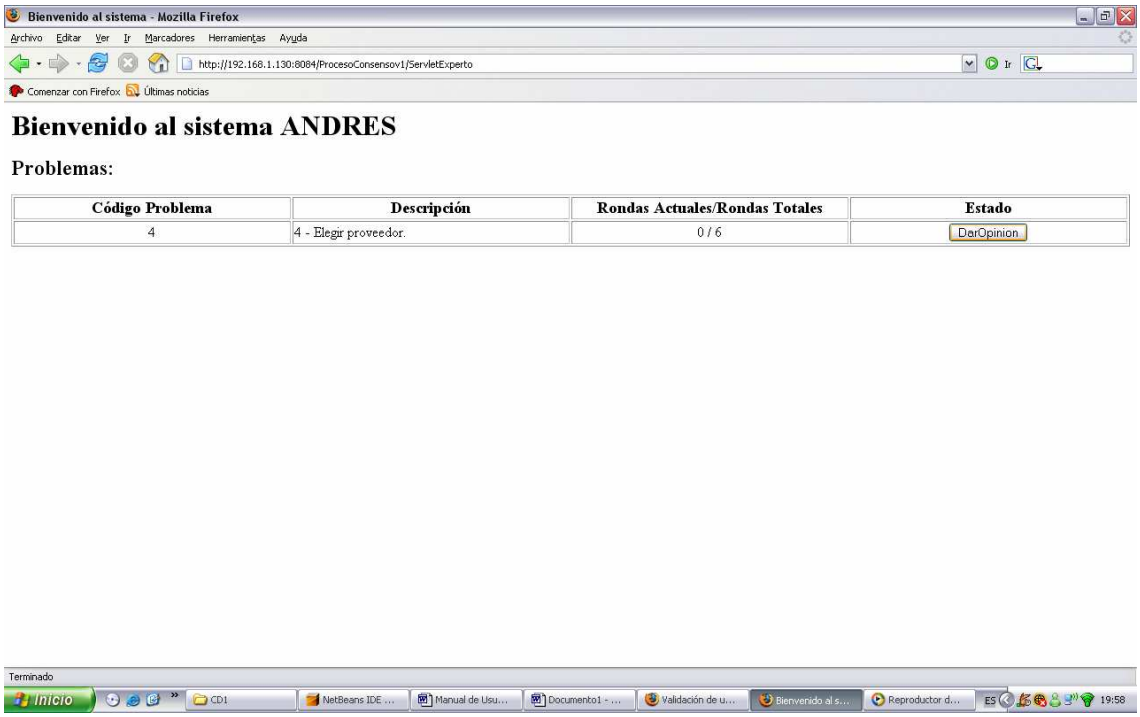


Figura: El experto andres ya puede opinar del problema.

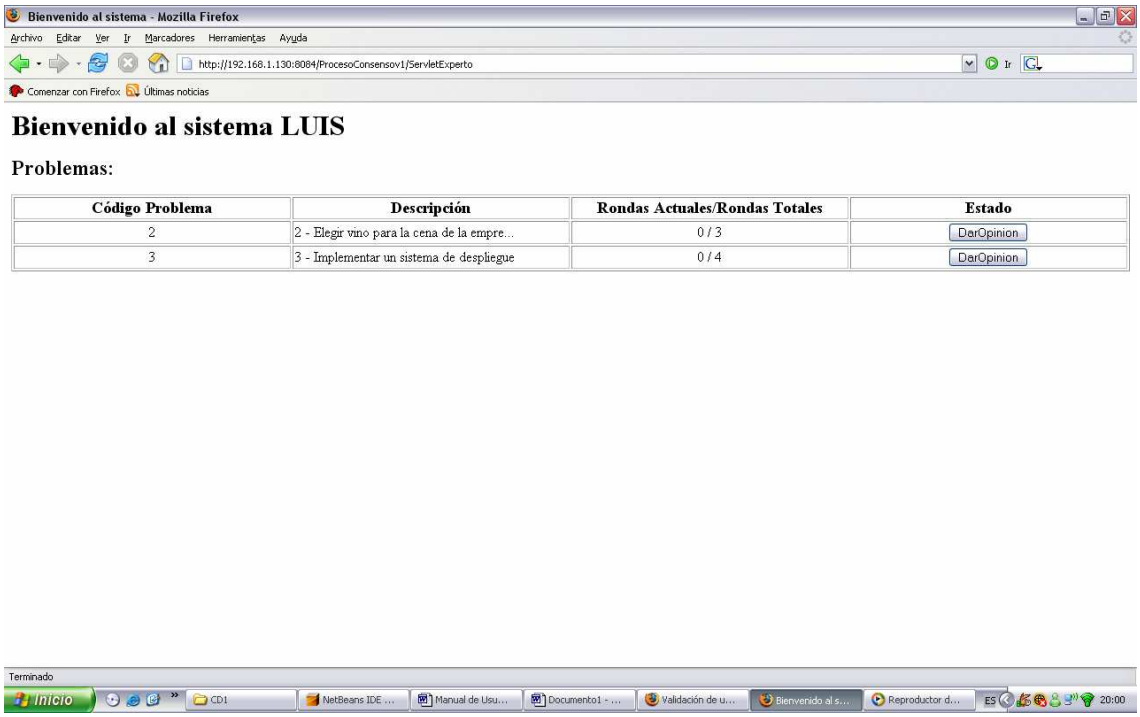


Figura: El experto luis no está al tanto de la existencia del problema.

3.3 Crear un nuevo experto.

Otra de las funcionalidades que puede llevar a cabo un Administrador es la de crear nuevos expertos para el sistema. Los expertos creados por un Administrador pueden se asignados a problemas de ese mismo Administrador o de otros.

La creación de un nuevo Experto también puede realizarla el Administrador de la BBDD, pero de manera manual.

Para crear un nuevo Experto en el sistema, el Administrador debe seleccionar la opción dos dentro del menú principal. Para ello debe pulsar el botón Introducir de la opción dos.

Figura: Botón de iniciar el alta de un nuevo experto.

Una vez pulsado el botón de Introducir, el sistema devuelve la pantalla principal para introducir un nuevo Experto al sistema:

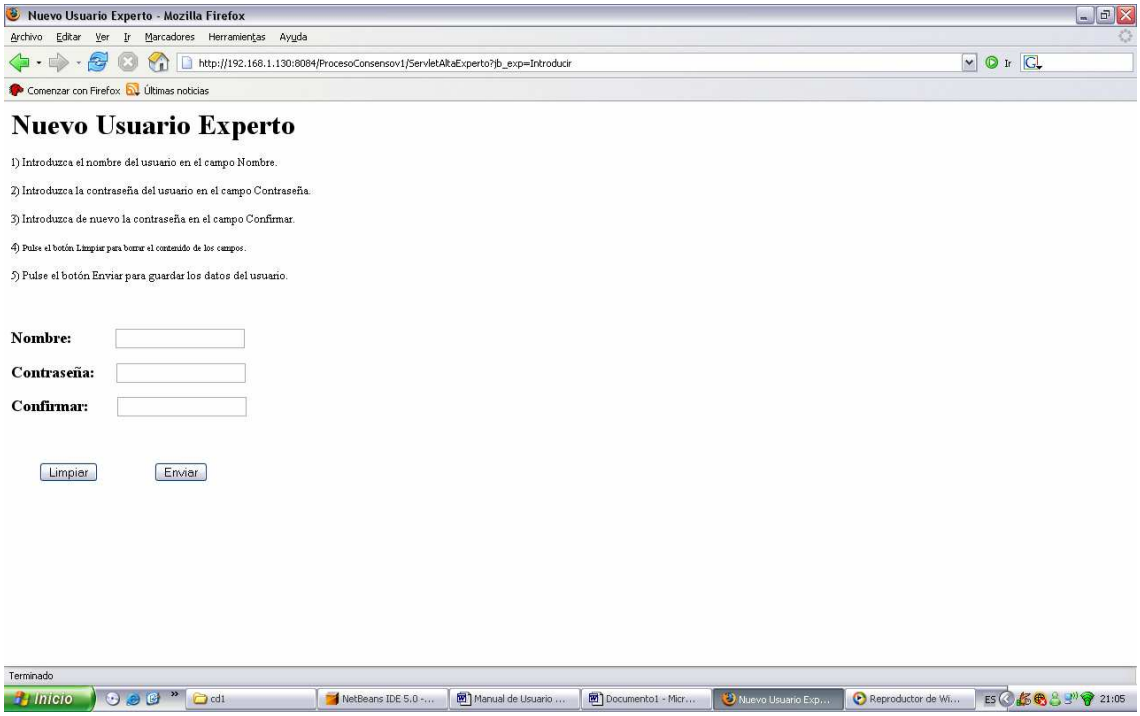


Figura: Página principal de alta de nuevo Experto.

En este formulario introduzca el nombre del usuario, su contraseña y la contraseña de nuevo. Cuando estén, pulse el botón Enviar para mandar los datos a la Base de Datos.

Si alguno de los datos requeridos no existe, el sistema vuelve a mostrar la ventana anterior con el mensaje de error:

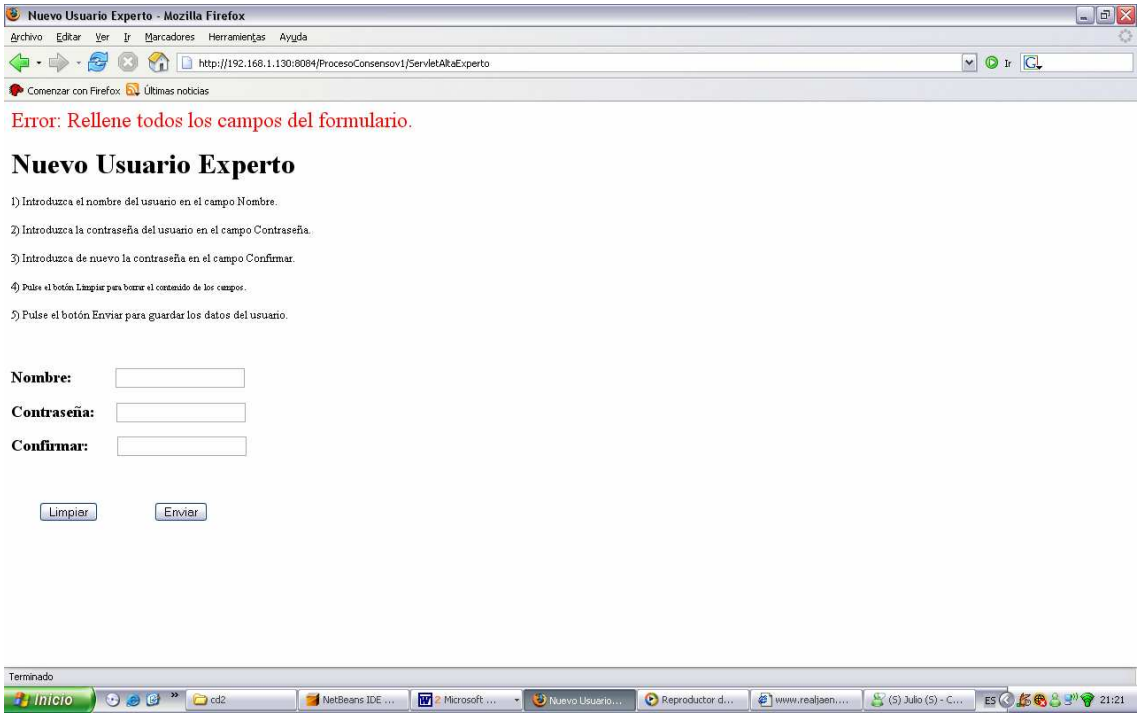


Figura: Faltan datos del usuario nuevo.

Si por el contrario, contraseña y confirmación de contraseña no existen, la aplicación de nuevo devolverá la ventana para introducir los datos. Además, indicará el error cometido.

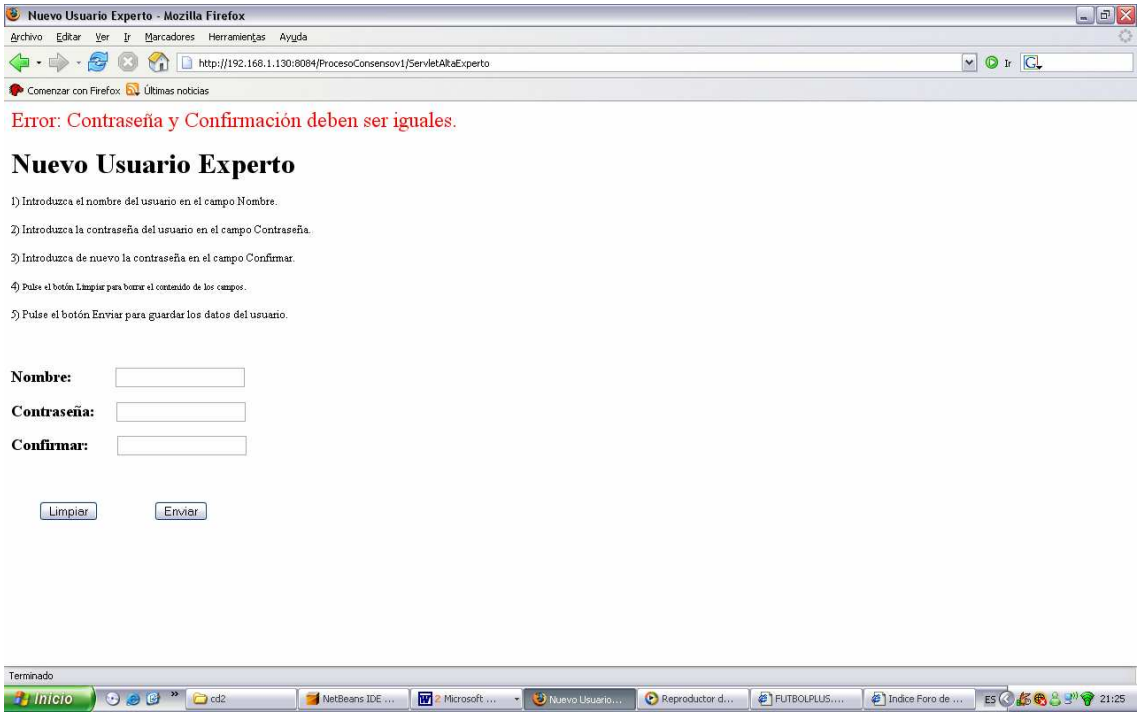


Figura: La contraseña y la confirmación no son iguales.

Si todos los datos son correctos, al pulsar el botón Enviar los datos son mandados a la Base de Datos.

3.4 Consultar un problema.

Una última funcionalidad del Administrador de un problema, es la de poder consultar su estado a lo largo de su existencia. Para ello ha de elegir la opción tres del menú principal del Administrador.

Para ello, debe en primer lugar seleccionar un problema del listado que se le muestra en esta pantalla, y a continuación pulsar el botón Consultar.

Figura: Administrador paco selecciona el problema 3.

Al pulsar el botón Consultar, el sistema devuelve la pantalla de estado del problema. Esta ventana está dividida en las siguientes partes:

- Parte 1→ Descripción del problema. Se realiza una descripción del problema indicando sus parámetros.

- Parte 2→ Alternativas. Se indican las alternativas del problema y el código de cada una en las tablas de más abajo.

- Parte 3→ Expertos. Se indica en una tabla el histórico de cambios a realizar por cada experto en las distintas rondas del proceso.

- Parte 4→ Grado de Consenso. En esta tabla se indica el grado de consenso alcanzado en cada una de las rondas del Proceso de Consenso.

- Parte 5→ Opinión colectiva. Se muestra la matriz de opinión colectiva de la última ronda del Proceso de Consenso.

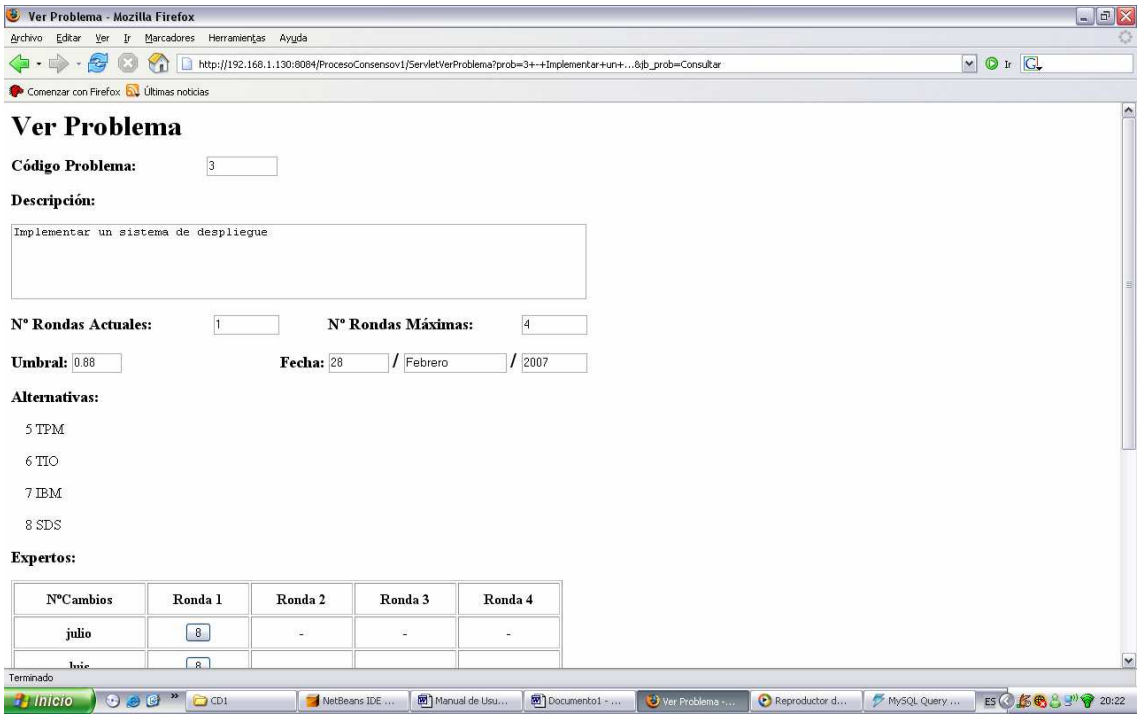


Figura: Pantalla de consultar problema (Parte 1).

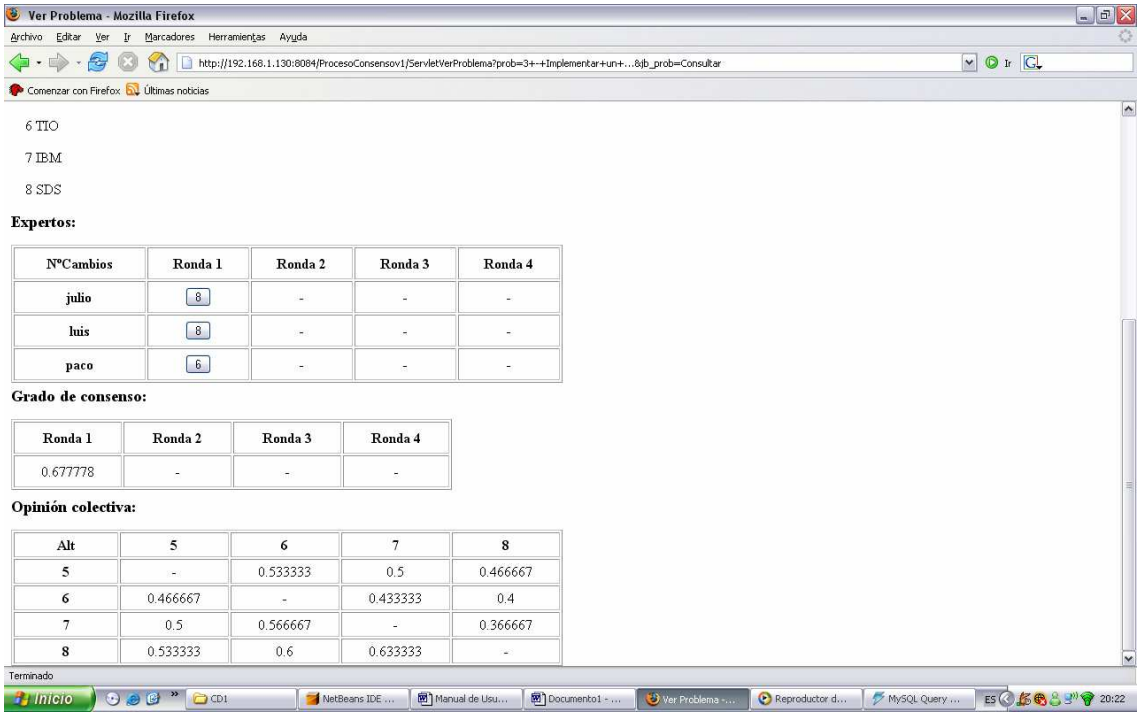


Figura: Pantalla de consultar problema (Parte 2).

Si pulsamos sobre el cualquier botón de la matriz de expertos, podemos ver detalladamente las recomendaciones enviadas al Experto seleccionado en la ronda especificada.

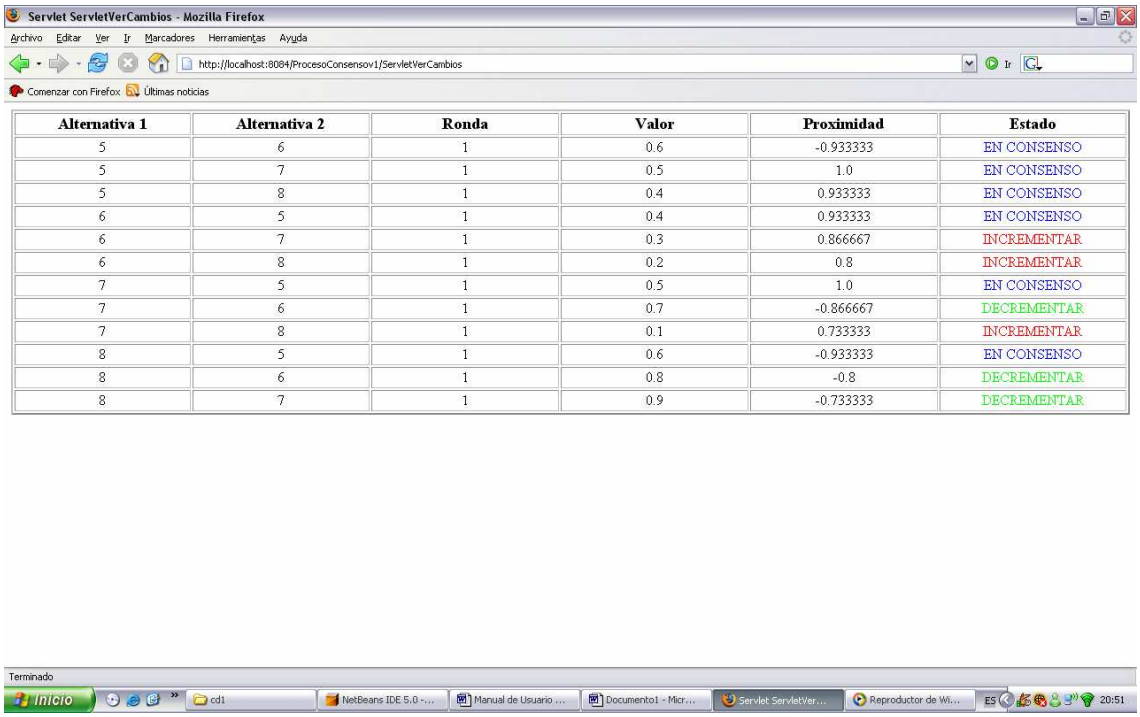


Figura: Recomendaciones para un experto.

4 Manual de usuario del Administrador de la Base de Datos.

El Administrador de la Base de Datos no es un usuario del sistema. Es decir, ni crea problemas ni participa en el debate. Tan solo gestiona la Base de Datos y lanza y mantienen la aplicación en el servidor.

4.1 Lanzar la aplicación.

El entorno de desarrollo NetBeans incorpora el servidor Apache Tomcat. Al tenerlo incorporado, basta con lanzar la aplicación desde el entorno para que éste llame a su vez a Tomcat. Para arrancar la aplicación web en el servidor usando NetBeans, siga los siguientes pasos:

1) Arranque el entorno de desarrollo NetBeans 5.0.

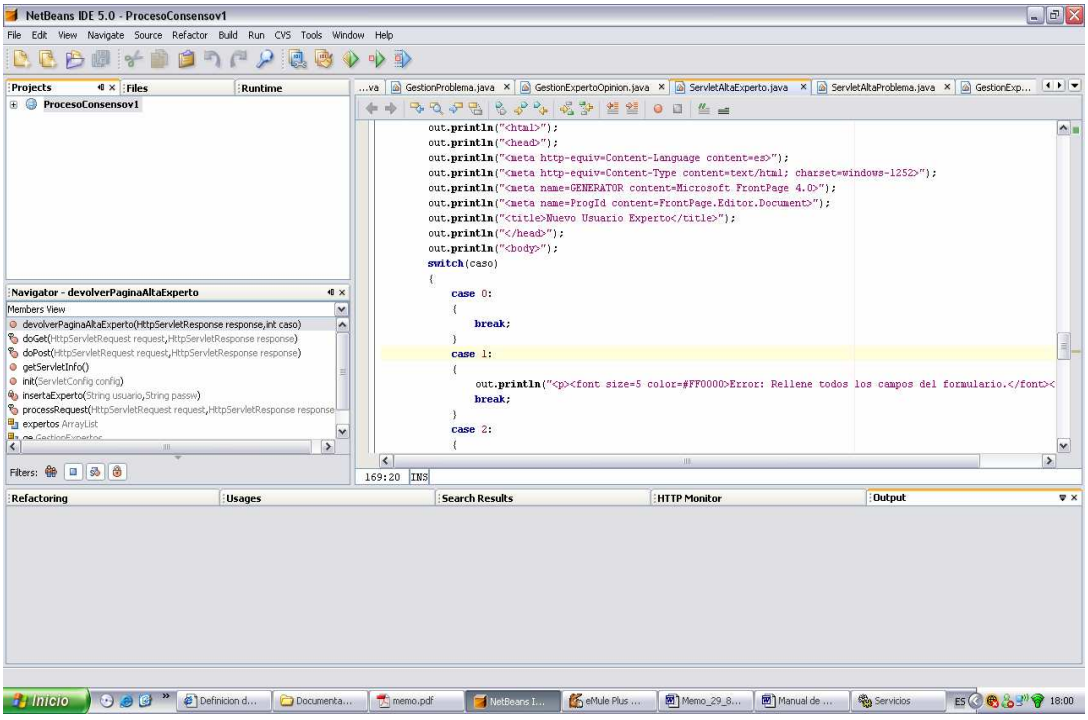


Figura: Ventana principal del entorno de desarrollo NetBeans.

2) Una vez arrancado, pulse en la barra de herramientas del entorno en el menú:

File → Open Project.

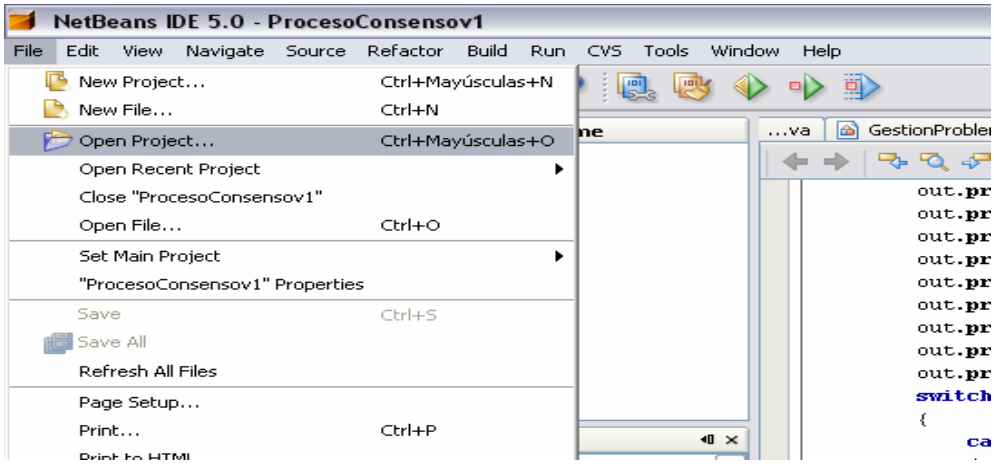


Figura: Menú File del Netbeans.

3) Busque en el directorio donde esté instalada la aplicación e el servidor y seleccione el directorio ProcesoConsensov1.

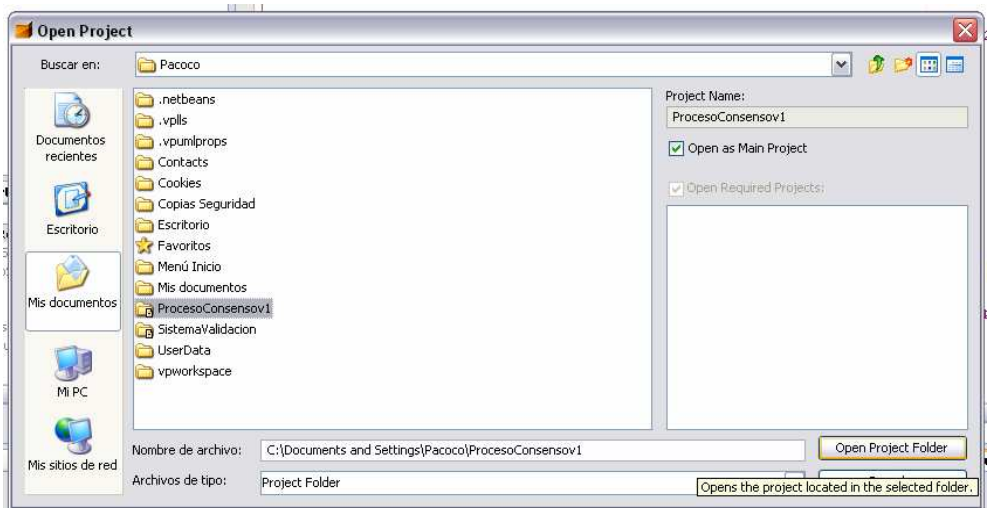


Figura: Cuadro para abrir proyecto de NetBeans.

4) Pulse en el botón Open Project Folder para abrir el directorio.

5) Una vez abierto, en la ventana principal del entorno de desarrollo NetBeans, pulse en el botón Run Main Project o pulse la tecla F6.

Figura: Botón para lanzar la aplicación web desde NetBeans.

6) Una vez ejecutado si le sale el siguiente contenido en la consola del sistema y además le ha cargado una ventana del explorador de internet, la aplicación ha sido lanzada correctamente.

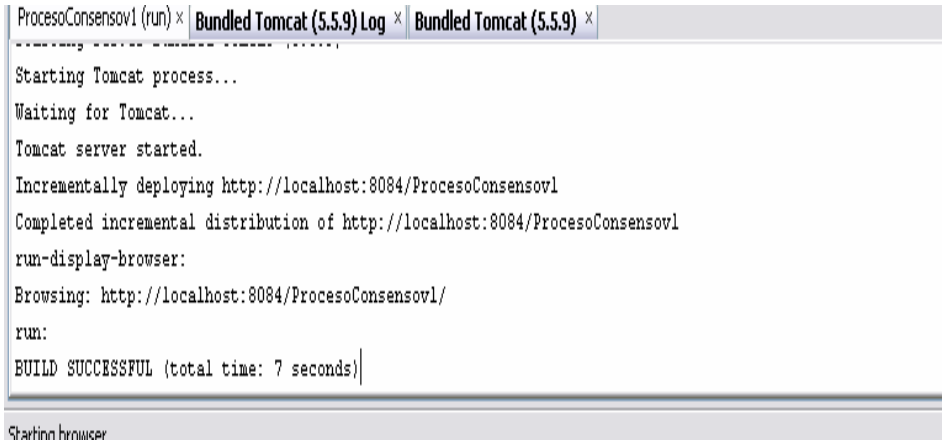


Figura: Consola del NetBeans.

La aplicación ya ha sido arrancada correctamente.

4.2 Gestión de la Base de Datos.

El Administrador de la Base de Datos es la única persona que interacciona directamente con la Base de Datos. El Administrador de problemas puede crear nuevos problemas, alternativas y expertos, pero la única persona que tiene acceso directo a las tablas de la Base de Datos es el Administrador de la Base de Datos.

Para poder entrar a la Base de Datos, el Administrador debe ejecutar el programa MySQL Query Browser.

Figura: Formulario de entrada a MySQL.

Al iniciar el programa, éste pedirá al usuario el identificador de usuario y la clave.

- Usuario → root.
- Contraseña → La que haya especificado al instalar el cliente MySQL.

Una vez dentro de la Base de Datos, tenemos que elegir el esquema en el que se encuentran almacenadas las tablas.

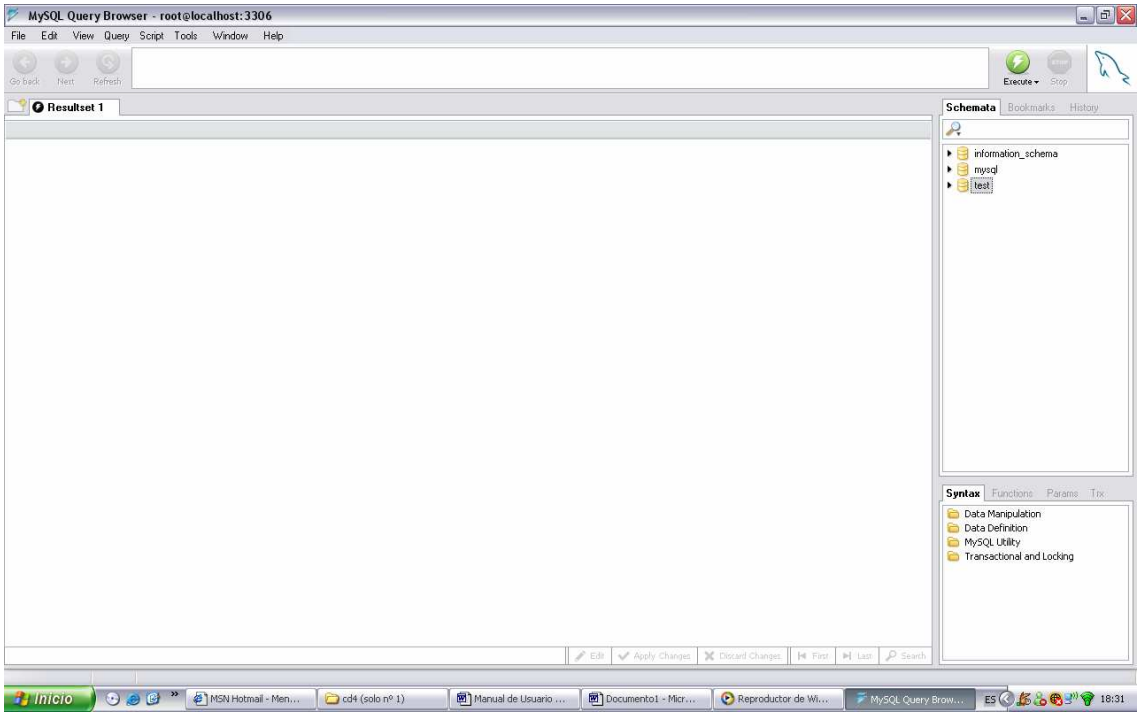


Figura: Panel principal del cliente MySQL.

Si seleccionamos el esquema que contiene las tablas tendremos un listado como el de la figura siguiente:

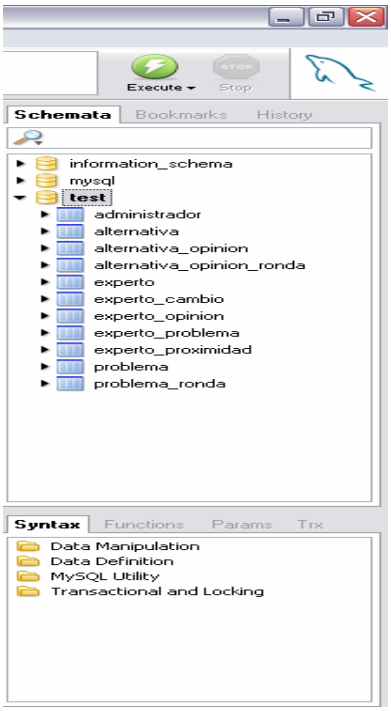


Figura: Tablas de la aplicación.

Si quiere ver el contenido de cualquier tabla haga doble clic sobre la tabla deseada y pulse el botón *Execute* de la parte superior derecha de la pantalla. Veamos el ejemplo de la tabla experto:

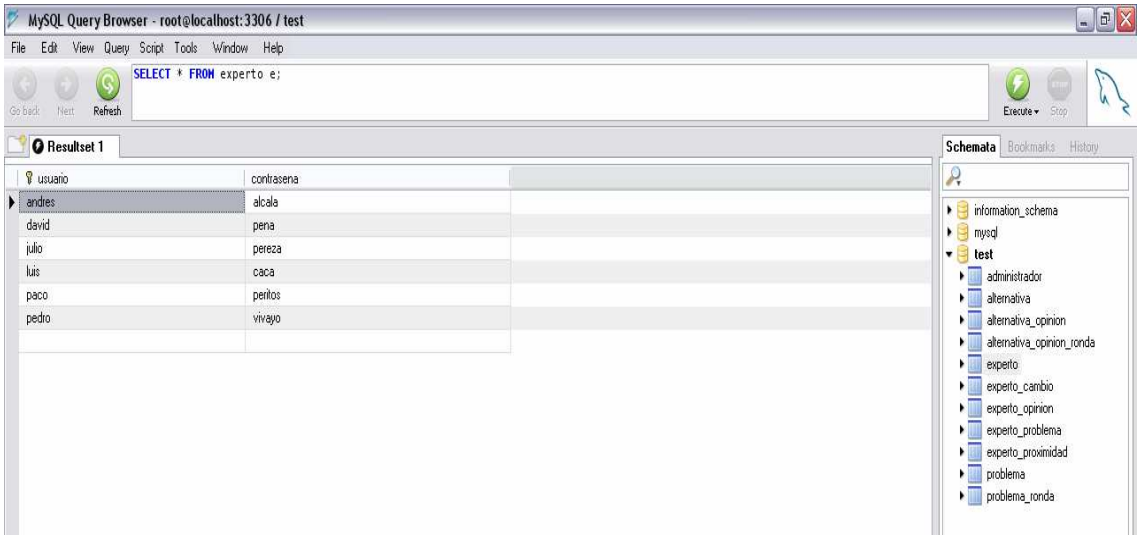


Figura: Contenido de la tabla experto.

A través de esta consola de MySQL, el Administrador de la Base de Datos puede filtar los datos de las tablas mediante consultas. Para realizar una consulta, introduzca el codigo SQL en el campo superior, y pulse el botón *Execute*.

Para eliminar instancias de una tabla siga los siguientes pasos:

Paso 1 → Muestre el contenido de la tabla como hemos visto arriba.:

Paso 2 → Pulse el botón Edit que aparece en la parte inferior de la ventana:

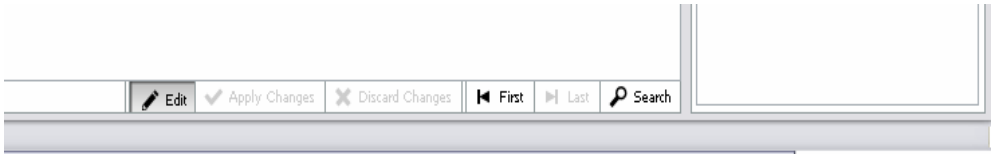


Figura: Panel de botones de la parte inferior.

Paso 3 → Seleccione el listado de instancias o filas a eliminar con el ratón.

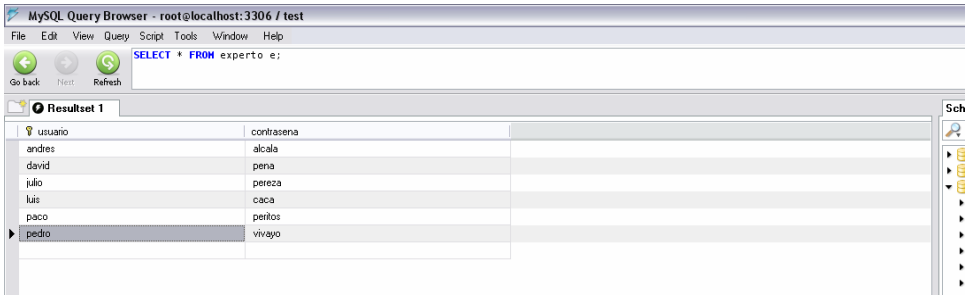


Figura: Fila a eliminar.

Paso 4 → Pinche con el botón derecho del ratón y seleccione la opción Delete Row(s).

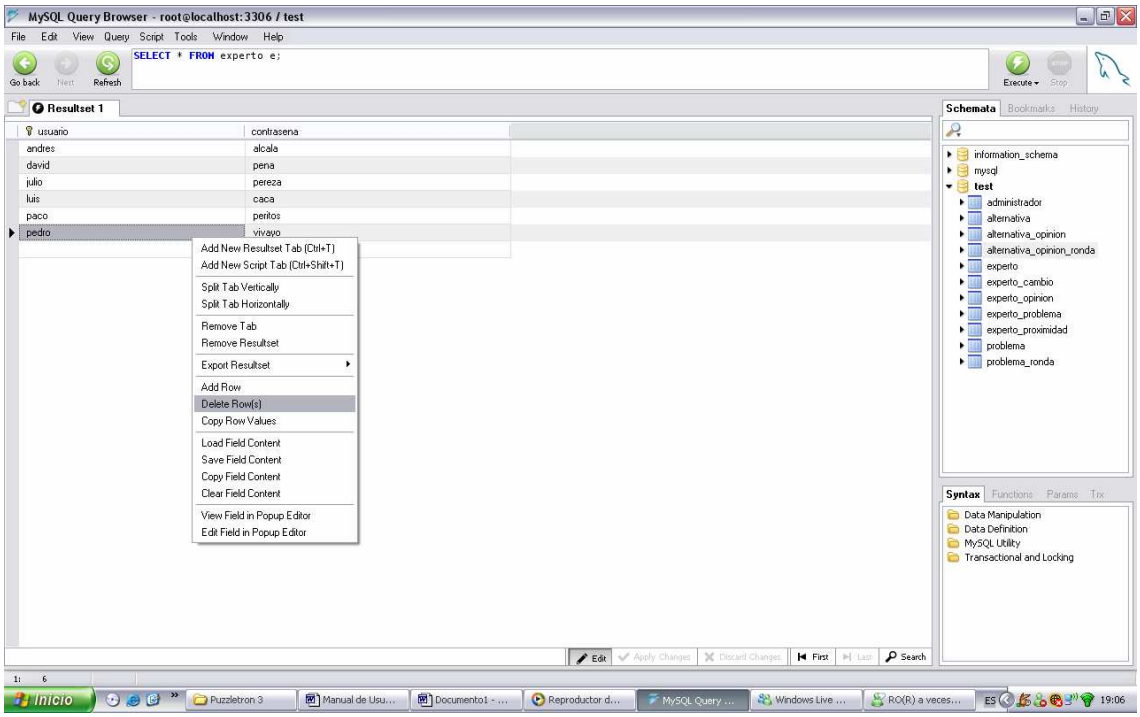


Figura: Opción de borrar una fila de la tabla experto.

Paso 5 → Las filas a eliminar saldrán con un color de fondo. Aún no han sido eliminadas. Para hacer el commit, pulse el botón Apply Changes.

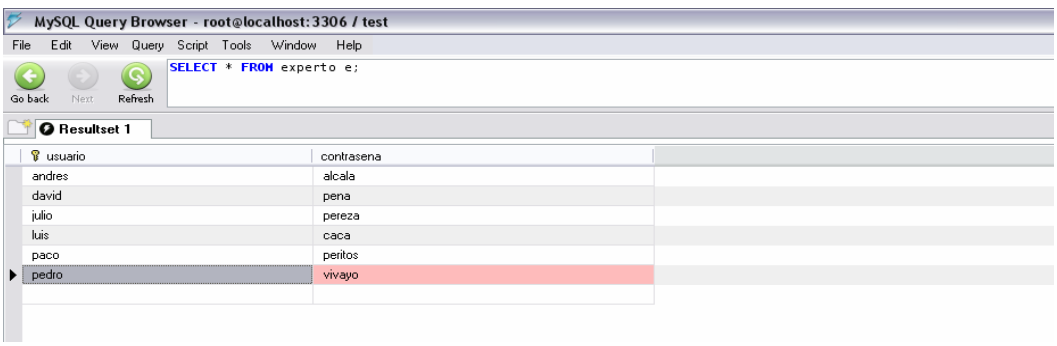


Figura: Fila a borrar a falta del commit.

Figura: Botón para aplicar los cambios (Commit).

4.2.1 Borrar un experto/administrador de la Base de Datos.

Para eliminar un Experto o un Administrador de la Base de Datos, debe tener en cuenta las referencias de ese usuario (Experto o Administrador) en el resto de tablas de la Base de Datos.

Para eliminar un Experto:

Paso 1 → Eliminar sus referencias en las tablas experto_cambio, experto_problema, experto_opinión y experto_proximidad.

Paso 2 → Eliminar la instancia del Experto en la tabla experto.

Para eliminar un Administrador, basta con eliminarlo de la tabla administradores.

4.2.2 Borrar una alternativa de la Base de Datos.

Para eliminar una alternativa de un problema de la Base de Datos, de nuevo hay que tener en cuenta las referencias de esta alternativa en la Base de Datos.

Para eliminar una alternativa de la Base de Datos:

Paso 1 → Eliminar sus referencias en las tablas alternativa_opinion y alternativa_opinion_ronda.

Paso 2 → Eliminar la instancia de la alternativa en la tabla del mismo nombre.

4.2.3 Borrar un problema de la Base de Datos.

Para eliminar un problema de la Base de Datos, de nuevo hay que tener en cuenta las referencias de este problema en la Base de Datos.

Para eliminar un problema de la Base de Datos:

Paso 1 → Eliminar sus referencias en las tablas experto_problema y problema_ronda.

Paso 2 → Eliminar la instancia del problema en la tabla del mismo nombre.

BIBLIOGRAFÍA.

En esta sección, nombramos los recursos bibliográficos empleados divididos en los siguientes bloques:

A) Temática del Proyecto.

Conjunto tanto de libros como de publicaciones electrónicas relacionadas con la temática del Proyecto Fin de Carrera.

[1] S. Saint, J.R. Lawson. “Rules for Reaching Consensus”. Jossey-Bass (1994).

[2] F. Herrera, E. Herrera-Viedma, J.L. Verdegay. “A model of consensus in group decisión making under linguistic assessments”. Fuzzy Sets an Systems 78 (1996).

[3] E. Herrera-Viedma, F. Herrera, F. Chiclana. “A Consensus Model for Multiperson Decisión Making with Different Preference Structures”. IEEE SMC Transactions on Systems, Man and Cybernetics-Part A: Systems and Humans 32 (2002).

[4] E. Herrera-Viedma, F. Mata, L. Martínez, F. Chiclana, L.G. Pérez “Measurements of Consensus in Multi-granular Linguistic Group Decision-Making”. Modeling Decisions for Artificial Intelligence, Proceedings Lecture Notes in Artificial Intelligence, Springer-Verlang 3131 (2004).

[5] E. Herrera-Viedma, F. Mata, L. Martínez, L.G. Pérez. “An Adaptative Module for the Consensus Reaching Process in Group Decision Making Problems”.

[6] L. Martínez, F. Mata, E. Herrera-Viedma. “An adaptative consensus support system model for Group Decisión Making Problems in Multi-granular Linguistic Context”

[7] E. Herrera-Viedma, F. Mata, L. Martínez, F. Chiclana. “A Consensus Support System Model for Group Decisión-Making Problems With Multigranular Linguistic Preference Relations”

B) Aplicación.

Conjunto de manuales, URLs, libros electrónicos, etc... consultados para la elaboración de la aplicación en sus distintas fases (Análisis, Diseño e Implementación).

C) Otras.

Documentos y libros de especial interés relacionados con el Proyecto Fin de Carrera en Ingeniería Informática.

"El Proyecto Fin de Carrera en Ingeniería Informática. Una Guía para el Estudiante". Christian W. Dawson, Gregorio Martín. Editorial Prentice Hall (2002).