ISSN: 1692-7257 - Volumen 1 - Número 19 - Año 2012

Revista Colombiana de Tecnologías de Avanzada

Recibido: 11 de febrero de 2011
Aceptado: 25 de marzo de 2011

STRATEGY DESIGN OF FAULT-TOLERANT CONTROL BASED ON KNOWLEDGE APPLIED TO A LOOP OF INDUSTRIAL LEVEL CONTROL

DISEÑO DE UNA ESTRATEGIA DE CONTROL TOLERANTE A FALLOS BASADO EN CONOCIMIENTO APLICADO A UN LAZO DE CONTROL DE NIVEL INDUSTRIAL

Ing. César Augusto Arias Cijanes*, MSc. Sandra Aranguren Zambrano**
PhD. Rocco Tarantino Alvarado**

* **Instituto Colombiano de Petróleos.**
Autopista Piedecuesta Km. 7, Bucaramanga, Santander, Colombia
Tel.: 57-7-6740001, 57-7-2345051, Fax: 57-7-6445444
E-mail: cijanes3@hotmail.com
** **Universidad de Pamplona**
Ciudadela Universitaria. Pamplona. Tel.: 57-7-5685303, Fax: 57-7-5685303 Ext. 156
E-mail: {saranguren, rocco}@unipamplona.edu.co

Abstract: The present paper shows the design of a fault-tolerant control strategy based on knowledge. This strategy avoids the cavitation of a centrifugal pump in a loop of level control, using a detection system and fault diagnosis and supervisory control, which are constructed from a qualitative model, where the prediction of process behavior interacts with observation itself.

Keywords: Fault-tolerant control, knowledge based system, detection and fault diagnosis system, supervisory control, cavitation.

Resumen: El presente artículo muestra el diseño de una estrategia de control tolerante a fallos basado en conocimiento. Esta estrategia evita la cavitación de una bomba centrifuga de un lazo de control de nivel, utilizando un sistema de detección y diagnóstico de fallos y un control supervisorio, los cuales se construyen a partir de un modelo cualitativo, donde la predicción del comportamiento del proceso interactúa con la observación del mismo.

Palabras clave: Control tolerante a fallos, sistema basado en conocimiento, sistema de detección y diagnóstico de fallos, control supervisorio, cavitación.

1. INTRODUCCIÓN

En las últimas décadas, el crecimiento de la demanda en la confiabilidad, seguridad y operación de las plantas en la industria petroquímica, es cada vez mayor. Se requiere de grandes inversiones en recursos físicos y humanos que ayuden a minimizar las fallas ocurridas en los sistemas de automatización y disminuir las pérdidas de oportunidad. Dentro de los sistemas de automatización se encuentran ampliamente utilizados los equipos rotativos (turbinas, bombas, compresores, entre otros.), los cuales son parte fundamental del proceso, cumpliendo la función de impulsión de los productos (líquidos y gaseosos) a través de las tuberías hacia las diferentes instancias del mismo. Debido a los grandes volúmenes de producción y a situaciones propias de la operación que se manejan en la industria petroquímica, estos equipos rotativos sufren a menudo graves daños en su integridad, debido a un fenómeno muy común de los sistemas de bombeo llamado *cavitación*, que

atiende a una condición anormal, que además de producir pérdidas de producción puede ocasionar daños a equipos y lesiones al personal.

La ocurrencia de fallas en los sistemas de automatización industrial y en especial en los lazos de control de nivel, exigen nuevas estrategias de control que sean capaces de tolerar el mal funcionamiento de los componentes del sistema y mantener los índices de desempeño de los procesos. (Hassan *et al*, 2000; Patton, 1993)

El diseño de control tolerante a fallos propuesto se enfoca en la prevención de la cavitación de una bomba centrífuga, el cual al detectar un estado anormal de operación de la planta ajusta una ley de control, que aumenta el nivel de Aceite Liviano de Ciclo (ALC) contenido en la torre despojadora y de esta manera asegura que la cabeza neta positiva de succión disponible (NPSHa) de la bomba, sea mayor que la cabeza neta positiva de succión requerida (NPSHr) y de esta manera evitar la degradación del equipo

1.1. Fallas en bombas centrífugas

En el contexto de las bombas centrifugas, el término cavitación implica un proceso dinámico de formación de burbujas dentro del líquido, su crecimiento y subsecuente colapsamiento a medida que el líquido fluye a través de la bomba.

Generalmente las burbujas que se forman dentro de un líquido son de dos tipos: Burbujas de vapor o burbujas de gas.

Las burbujas de *vapor* se forman debido a la vaporización del líquido bombeado. La cavitación inducida por la formación y colapso de estas burbujas se conoce como *cavitación vaporosa*.

Las burbujas de *gas* se forman por la presencia de gases disueltos en el líquido bombeado (generalmente aire pero puede ser cualquier gas presente en el sistema). La cavitación inducida por la formación y colapso de estas burbujas se conoce como *cavitación gaseosa*.

En ambos tipos, las burbujas se forman en un punto interior de la bomba en el que la presión estática es menor que la presión de vapor del líquido (cavitación vaporosa) o que la presión de saturación del gas (cavitación gaseosa).

La cavitación vaporosa es la forma de cavitación más común en las bombas de proceso. Generalmente ocurre debido a un insuficiente NPSH (Cabeza Neta Positiva de Succión) disponible o a fenómenos de recirculación interna. Las burbujas de vapor se forman dentro de la bomba cuando la presión estática en algún punto baja a un valor igual o menor que la presión de vapor del líquido**.**

La presión estática en algún punto dentro de la bomba puede bajar hasta un nivel inferior a la presión de vapor bajo dos condiciones:

1. Porque la caída de presión actual en el sistema externo de succión es mayor que la que se consideró durante el diseño del sistema. (Es una situación bastante corriente). Esto resulta en que la presión disponible en la succión de la bomba (NPSHa) no es suficientemente alta para suministrar la energía requerida para superar la caída de presión interna (NPSHr) propia del diseño de la bomba.
2. Porque la caída de presión actual dentro de la bomba (NPSHr) es más grande que la informada por el fabricante y que se usó para seleccionar la bomba.

Los síntomas o efectos más comunes que produce la cavitación son:

- Reducción de la capacidad de bombeo.
- Disminución de la generación de cabeza.
- Vibración y ruido anormal.
- Daños a los componentes (erosión o picaduras).

Deformaciones mecánicas, tales como:

- Torcedura y deflexión de los ejes.
- Daño a los rodamientos y roces por la vibración radial.
- Daño en el rodamiento de empuje por movimiento axial.
- Rotura de la tuerca de fijación del impulsor.
- Daño en los sello.
- Corrosión con cavitación.

Las deformaciones mecánicas pueden deteriorar completamente a la bomba y requerir reemplazo de partes. El costo de estas partes puede ser alto. (Walter, 2005). En la figura 1 se observa el daño de un impulsor de una bomba centrífuga producida por cavitación.

Fig. 1. Daño de impulsor de una bomba centrífuga producido por cavitación.
Fuente: http://www.irrigationscraft.com

1.1.1. Previniendo la cavitación

Si una bomba centrífuga esta cavitando, se deben hacer necesariamente varios cambios en el diseño del sistema o en la operación, con el objetivo de incrementar la cabeza neta positiva de succión disponible (NPSHa) sobre la cabeza neta de succión requerida (NPSHr). Por ejemplo:

Aumento de la NPSHa: Esto se logra mediante el aumento de la presión en la succión de la bomba. Por ejemplo, si una bomba está succionando el líquido contenido en un tanque cerrado, se aumenta esta presión, incrementando el nivel del líquido en el tanque. (Karassik *et al.*, 2001; Lierberman, 1991).

1.2. Sistema de detección y diagnóstico de fallas de la estrategia tolerante a fallos basado en conocimiento

El método de detección y diagnóstico de fallas planteado para la estrategia de control tolerante a fallos, se basa en el uso de diferentes técnicas de representación del conocimiento para describir la estructura y comportamiento de un proceso complejo, las cuales se comparan con la conducta observada del sistema real, y a partir de las inconsistencias se pueden elaborar y validar hipótesis acerca de los fallos presentes y sus causas. (Hamscher, 1992). En la figura 2 se muestra el esquema de detección de fallos por medio de la comparación de la conducta observada del proceso con la base de conocimiento que en este caso describe el modelo del mismo.

Los sistemas basados en conocimiento pueden incorporar varias reglas para resolver un problema y concordar estas con las posibles consecuencias. (Venkatasubramanian, 2002).

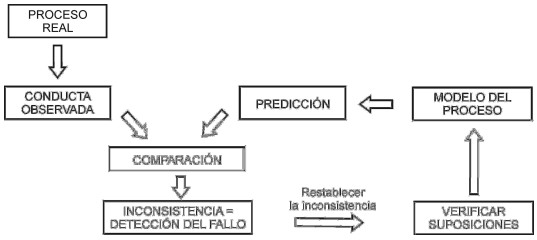

Fig. 2. Esquema del método de detección y diagnóstico de fallos para la estrategia de control tolerante a fallos basada en conocimiento experto.
Fuente: Hamscher, 1992.

En la industria han sido utilizadas muchas técnicas en la detección temprana y el diagnóstico de fallos, algunas de estas técnicas incluyen el uso de sistemas basados en conocimiento que contienen las relaciones causales entre los síntomas observados en los procesos reales y las fallas que los causaron. (Agudelo *et al.*, 2007).

Para el método de detección y diagnóstico de fallas se plantea la construcción de un algoritmo en un lenguaje de programación declarativa, el cual por medio de un motor de inferencia de un sistema Shell basado en reglas de Java, llamado JESS[1], se logra conocer el grado de verdad de cualquiera de las hipótesis que se puedan tener como causas de fallo, una vez actualizada la base de hechos. (Friedman-Hill, 2003)

2. METODOLOGIA DE DISEÑO

En la figura 3 se muestran las etapas de una metodología sistemática propuesta por (Blanke *et al.*, 2000) que es utilizada por los autores para el diseño de la estrategia de control tolerante a fallos basado en conocimiento.

2.1. Análisis del sistema

Para el presente trabajo se selecciona un lazo de control de nivel, el cual pertenece a una de las plantas de la refinería de Barrancabermeja (Colombia). Este lazo está comprendido por una torre despojadora y su respectivo control de flujo, de donde se extrae la data para la validación del sistema. La planta se muestra en la figura 4.

[1] JESS ha sido desarrollado por SNL (Sandia National Laboratories) por sus siglas en inglés.

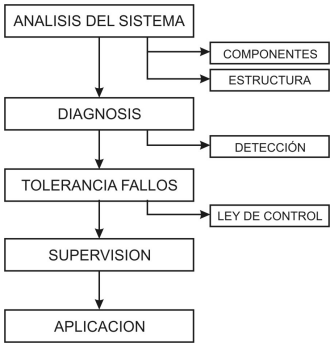

Fig. 3. Metodología para el diseño de la estrategia tolerante a fallos basado en conocimiento.
Fuente: Puig et al., *2004.*

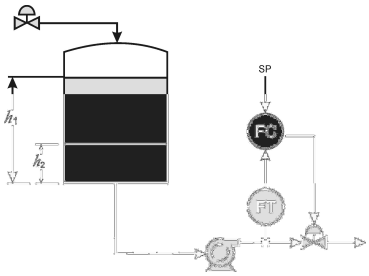

Fig. 4. Sistema de control de nivel.
Fuente: Refinería de Barrancabermeja

En este análisis se identifican los componentes físicos del sistema, los cuales son principalmente la torre despojadora, la bomba centrífuga, la válvula de control, el transmisor de flujo y el transmisor de presión de succión de la bomba. El modelo del sistema se logra a través de la reconstrucción modelo matemático de cada componente del lazo de control utilizando los datos reales de proceso y las hojas de datos de cada equipo. En cuanto al diseño de los controladores se plantea seguir el método del lugar de las raíces teniendo en cuenta los puntos de operación de cada controlador. Dicho modelamiento se realiza en mediante la *toolbox* Simulink del software MATLAB[2]. Un modelo matemático básico del sistema es mostrado en la figura 5.

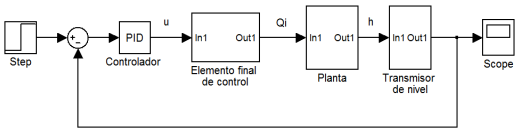

Fig. . Modelo matemático básico del sistema de control de nivel construido en Simulink.
Fuente: Autores

El objetivo de construir los modelos matemáticos del sistema y de los controladores en MATLAB, es simular el comportamiento dinámico del sistema real, tanto en operación normal como en fallo. Los datos generados de dichos modelos, el análisis causa raíz, las fallas más comunes registradas, entre otros son enviados a una base de datos de SQL[3] Server mediante la toolbox Database de MATLAB. Posteriormente estos datos son capturados por una interface de comunicación y procesamiento de datos, desarrollada en un compilador C++ y quedan finalmente dispuestos para el sistema de detección y diagnóstico de fallos.

2.2 Sistema de detección y diagnóstico de fallos

El sistema de detección y diagnóstico de fallos consiste en un algoritmo basado en conocimiento, el cual es programado mediante el lenguaje Jess e implementado en la plataforma de software JESS de Java. Para que el sistema detecte y diagnostique los fallos que pueden producir la cavitación, estos están claramente caracterizados, se conoce su patrón temporal y se tiene la base de conocimiento fenomenológico, asegurando que los síntomas que representan a cada fallo sean suficientes para la detección del evento anormal.

En la figura 6 se muestra la arquitectura del sistema de detección y diagnóstico de fallos basado en conocimiento. La memoria de trabajo o *base de hechos* almacena los hechos descritos por los síntomas que caracterizan la causa de un fallo, la base de conocimiento o *base de reglas* contiene todas las reglas para cada hecho en particular, el *motor de inferencia* es la parte en donde se ejecuta el raciocinio sobre los hechos y las reglas, el *emparejamiento de patrones* selecciona la regla aplicable para cada hecho, posteriormente en la *agenda* se activan las reglas que tiene correlación con los hechos y finalmente el *motor de ejecución* determina el orden de las activaciones entre reglas y hechos.

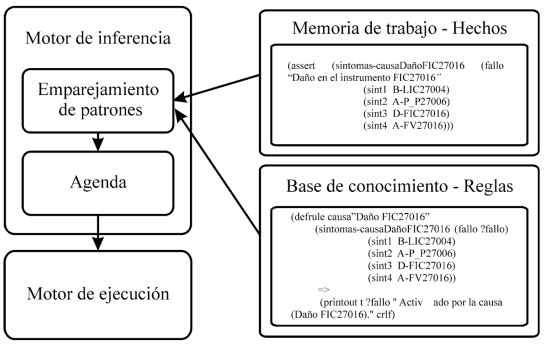

Fig. 6. Arquitectura del sistema de detección y diagnóstico de fallos basado en conocimiento.
Fuente: Strauss, 2007

[2] MATLAB es una marca registrada de MathWorks

[3] SQL Server es una marca de Microsoft Corporations

Para realizar el diseño del sistema de detección y diagnóstico de fallos se plantean los siguientes pasos, los cuales son la base de su arquitectura.

2.2.1 Base de conocimiento

Como punto de partida se realizan todas las actividades relacionadas con la adquisición del conocimiento, las cuales consisten en la revisión bibliográfica de proceso, entrevistas con los operadores, supervisores y coordinadores del sistema de control distribuido de la planta en cuestión, en donde basados en la experiencia se abordan las causas y síntomas de los fallos que pueden producir la cavitación en la bomba centrifuga y finalmente se revisan los históricos de fallas del equipo, consignados en las bitácoras de operación. Cada una de las causa de fallos con sus respectivos síntomas son almacenados en una base de datos de SQL Server, las cuales conforman la base de conocimiento experto. Después de obtener una cantidad suficiente de información se procede a construir la base del conocimiento mediante la validación, organización, depuración y tabulación de la información adquirida, en donde se definen las variables a utilizar, por ejemplo, *Nivel de la torre despojadora LIC27004, presión de succión de P27006, Flujo a viscorreductora FIC27016, Porcentaje de apertura de la válvula de flujo a viscorreductora FV27016, entre otras,* para la posterior construcción de los algoritmos de la base de hechos y la base de reglas.

2.2.2. Base de hechos (memoria de trabajo)

La base de hechos está compuesta por todos los síntomas que caracterizan una causa generadora de algún fallo. Por ejemplo, si se presenta la cavitación en la bomba centrifuga y su causa raíz es el "*Daño en el instrumento FIC27016 (controlador de flujo hacia viscorreductora)*", se pueden generar los siguientes síntomas que determinan su posible presencia:

- Nivel de LIC27004 por debajo del 15%.
- Incremento en la presión de succión de la bomba P27006.
- Decremento en el FIC27016 (Flujo a viscorreductora).
- Incremento en el FV27016.

A continuación se muestra la programación declarativa del hecho llamado "*sintomas-causaDañoFIC27016*", el cual es la causa raíz generadora de la cavitación y se determina su presencia mediante 4 síntomas: **sint1 B-LIC27004** (*Bajo nivel de en la torre despojadora*), **sint2 A-P_P27006** (*Aumento de la presión de succión de P27006*), **sint3 D-FIC27016** (*Disminución del flujo a viscorreductora*) y **sint4 A-FV27016** (*Aumento de porcentaje de apertura de la válvula de flujo a viscorreductora FV27016*). El código en lenguaje Jess para este hecho, se presenta de la siguiente manera:

```
(assert (sintomas-causaDañoFIC27016 (fallo "Daño en el instrumento FIC27016"
                (sint1 B-LIC27004)
                (sint2 A-P_P27006)
                (sint3 D-FIC27016)
                (sint4 A-FV27016)))
```

2.2.3. Base de reglas

La base de reglas contiene las instrucciones condicionadas para cada hecho en particular. Por ejemplo, a continuación se muestra la programación declarativa de la regla llamada *causa"Daño FIC27016"*, la cual se activa por medio del emparejamiento con el hecho *"sintomas-causaDañoFIC27016"*, compuesto por 4 síntomas: sint1 B-LIC27004, sint2 A-P_P27006, sint3 D-FIC27016 y sint4 A-FV27016. El código en lenguaje Jess para dicha regla, se presenta de la siguiente manera:

```
(defrule causa"Daño FIC27016"
    (sintomas-causaDañoFIC27016 (fallo ?fallo)
                (sint1 B-LIC27004)
                (sint2 A-P_P27006)
                (sint3 D-FIC27016)
                (sint4 A-FV27016))
    =>
     (printout t ?fallo " Activado por la causa (Daño FIC27016)." crlf)
```

2.2.4. Motor de inferencia

El motor de inferencia es el mecanismo usado para extraer el conocimiento de la base de conocimientos, el cual por medio de un emparejamiento de patrones se logra una solución o conclusión determinada.

El motor de inferencia se ejecuta por medio de un sistema Shell basado en reglas de Java, el cual utiliza como interprete el lenguaje Jess, que es ampliamente utilizado para construir sistemas inteligentes, el cual por medio de interrogaciones permite conocer el grado de verdad de cualquiera de las hipótesis que se puedan tener como causas de fallo una vez actualizada la base de hechos.

2.2.5. Interface de usuario

La interface de usuario se desarrolla mediante el compilador C++ Builder[4], el cual maneja un lenguaje de programación imperativa, con la gran ventaja que ofrece de crear algoritmos orientados a objetos con un ambiente de desarrollo visual que facilita la construcción del programa y solución de problema, además permite concentrarse solo en resolver el problema planteado, debido a que la interface gráfica (pantallas) es generada por el propio compilador.

En la interface de usuario planteada se puede visualizar el comportamiento dinámico del proceso y monitorizar de manera continua sus variables de: nivel de la torre despojadora y presión de succión de la bomba centrífuga. Además, permite conocer la condición operacional (normal o en falla) en que se encuentra el sistema en determinado momento.

En la figura 4 se muestra una imagen de la interface gráfica del sistema de control tolerante a fallos creada en C++ Builder.

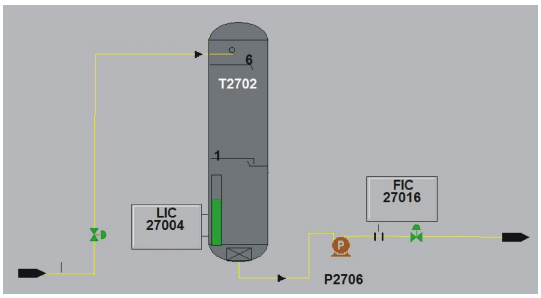

Fig. 4. Interface de usuario de la planta de control de nivel creada en Borland C++ Builder.
Fuente: Autores.

3. CONTROL TOLERANTE A FALLOS

El problema de control tolerante mediante el enfoque *activo* puede resolverse de dos formas: ya sea mediante la *acomodación al fallo*, o bien, mediante la *reconfiguración*. La *acomodación al fallo* consiste en resolver el problema manteniendo la estructura del controlador y modificando solamente los parámetros. Por otro lado, la *reconfiguración* consiste en cambiar las entradas y salidas del controlador así como reajustar la ley de control. (Puig *et al.*, 2004).

El presente trabajo plantea el diseño de una estrategia de control tolerante a fallos que se basa en el tipo *activo* y el enfoque utilizado es mediante la *reconfiguración*, en donde se cambia la ley de control cuando se detecta la cavitación. En la figura 5 se plantea un esquema para realizar la reconfiguración de la ley de control del sistema tolerante a fallos.

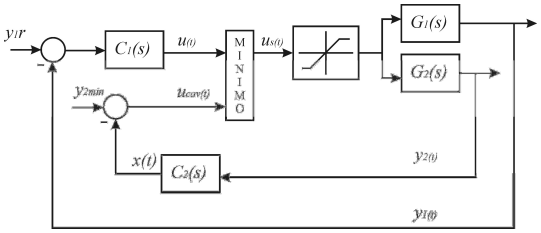

Fig. 5. Diagrama del sistema de control para la reconfiguración de la ley de control.
Fuente: López y Aguilar, 1999

El proceso está caracterizado por dos funciones de transferencia $G_1(s)$ y $G_2(s)$ los cuales corresponden a la salida principal $y_1(t)$ y la salida secundaria $y_2(t)$ respectivamente. La acción de controlador principal $u(t)$ puede saturarse con la señal de controlador secundario $u(cav)$. $C_1(t)$ es el controlador principal llamado "controlador de nivel normal" y $C_2(t)$ es el controlador secundario "controlador de nivel mínimo".

4. SUPERVISORIO

El control supervisorio, vigila que los modos y puntos de operación de los distintos controladores estén en coordinación y optimización para mantener la operación del proceso en las zonas de operación normal, y para esto debe reconocer zonas de riesgo operacional, para activar estructuras, algoritmos, etc. de forma que el proceso se mantenga dentro de los límites normales de operación, según las características inherentes de cada equipo. En otras palabras un control supervisorio mantiene al proceso en las zonas de operación de bajo riesgo operacional por que en esas zonas se protege a los equipos de fallas potenciales y funcionales a fin de asegurar la disponibilidad del proceso altamente crítico.
Volviendo a la figura 5, en condiciones de operación normal (nivel por encima del valor crítico), el controlador de flujo es quien gobierna al elemento final de control garantizando un flujo constante de salida (ya que la salida del controlador de nivel mínimo se satura al 100% por encontrarse la señal de nivel por encima de su set point), si la acción de control $u(t)$ es mayor que $u_{cav}(t)$ el

[4] C++ Builder es una marca de Borland Inprise.

sistema trabaja con $C_1(t)$ como si el selector no existiera.

Por otro lado, si $y_2(t)$ alcanza el límite inferior $y_{2min}(t)$, el control $u_{cav}(t)$ se vuelve más grande que $u(t)$ y por consiguiente se selecciona la acción de control es $u_{cav}(t)$. La estructura de éste controlador de nivel mínimo es similar al del control normal o sea PID, la diferencia es que el *set point* de nivel se configura ligeramente por encima del valor mínimo de presión de succión permitida, para garantizar el incremento de esta presión en la bomba, cerrando la válvula posterior a la bomba y permitiendo el llenado de la torre despojadora y de esta manera se aumenta su nivel y por consiguiente la presión hidrostática en la entrada de la bomba, evitando su cavitación

El control supervisorio está vigilando contantemente las conclusiones emitidas por el sistema de detección y diagnóstico de fallos, cuando se detecta la cavitación, éste asigna un valor de set point alto para $y_{2min}(t)$ con el fin de mantener a $y_2(t)$ por encima del límite inferior permisible. En la figura 6 se muestra la estrategia de control tolerante a fallos basado en conocimiento.

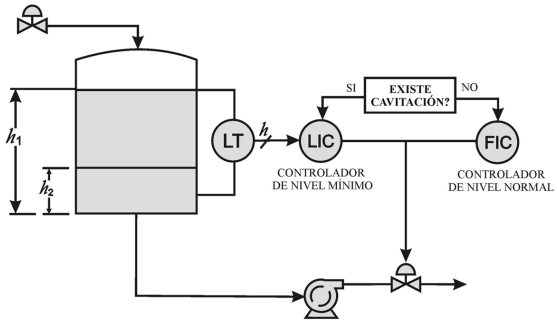

Fig. 6. Estrategia de control tolerante a fallos basado en conocimiento para el lazo de control de nivel.

5. INTERFASE DE PROCESAMIENTO

Esta interfase proporciona un análisis matemático a los datos disponibles en la base de datos, con el fin de determinar las tendencias de las señales. También aquí se realiza las rutinas de búsqueda y actualización de datos sobre los registros de la base de datos y se ejecuta la validación de la presencia de los síntomas del sistema de detección y diagnóstico de fallos. La codificación de esta interface se construye por medio del compilador C++ builder.

6. ARQUITECTURA DEL SISTEMA

En la figura 7 se muestra la arquitectura de software del diseño propuesto.

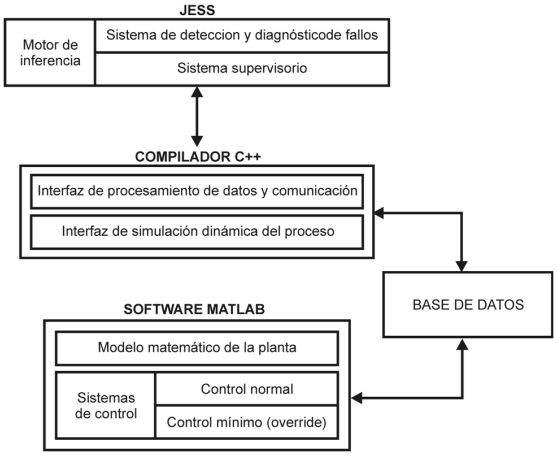

Fig. 7. Arquitectura del diseno de la estategia de control tolerante a fallos basado en conocimiento experto.

En la figura 7, se distingue cada uno de los componentes del sistema junto con la herramienta de software utilizada para su construcción. El modelo del sistema y los modelos de los controladores se diseñan e implementan en MATLAB, el motor de inferencia lógica para el sistema de detección y diagnóstico de fallos y el controlador supervisorio, se realiza por medio de JESS que es un sistema Shell basado en reglas de Java, el cual utiliza como interprete el lenguaje Jess, la interface de usuario y procesamiento de datos se construyen en el compilador Borland C++ Builder y finalmente la conexión entre aplicaciones es realizada a través de la base de datos SQL Server.

8. CONCLUSIONES

El diseño de la estrategia de control tolerante a fallos basado en conocimiento de un lazo de control de nivel, se plantea mediante la integración herramientas de software especializadas, obteniendo como resultado un sistema dinámico animado que simula el comportamiento real del sistema de control de nivel de la torre despojadora y la detección temprana de la cavitación en su sistema de bombeo.

Con el presente planteamiento se evidencia la potencialidad que tiene los sistemas basados en conocimiento para resolver problemas complejos.

El aporte innovador del presente trabajo consiste en la creación de una estrategia de control tolerante a fallos utilizando el conocimiento de los expertos sobre las causas y sus síntomas que caracterizan un fallo de manera cualitativa.

RECONOCIMIENTO

Los autores agradecen el soporte ofrecido por ECOPETROL S.A. durante el desarrollo del presente trabajo.

REFERENCIAS

Agudelo C., Quiles E., Morant F. (2007). Uso de Sistemas Expertos en el Diagnóstico de fallos en Procesos Complejos. *XII Convención de Ingeniería eléctrica*. U. Central Maria Abreu de las Villas. Villa Clara (Cuba).

Blanke M., Frei C. W., Kraus F., Patton R. J., Staroswieck. (2000). What is Fault-Tolerant Control?. 4th IFAC Symposium on Fault Detection Supervision and Safety for Technical Process, Safeprocess 2000, Vol 1 Budapest, 14-16 June. Pág. 40-51.

Friedman-Hill E. (2003). *Jess in Action*, p.19, chapter 2. Manning Publications Co., Greenwich, USA.

Hamscher, W.; Console, L.; De Kleer, J. Readings in Model-based Diagnosis. Morgan Kaufmann Publishers Inc., California, 1992.

Karassik, I., Messina, J., Cooper, P., Heald, C., (2001). *Pump Handbook*, pp. 2.224, 2.248, 2.347, 2.356-2.357, 2.397-2.398, 2,448, 2.71, 2.88. McGraw-Hill, New York USA.

Lierberman, N. (1991). *Troubleshooting Process Operations*, pp. 431, 245-248, 252, 255, 431. Pennwell Publishing Company, Tulsa, Oklahoma USA.

López A., Aguilar M., (1999). *Using Multivariable Nonlinear Stability Theory for Override Control System*. European Control Conference. Germany.

López A., Aguilar M., (1999). *On The Statbility of Override Control System*. LAAS-CNRS, Toulouse France, Rapport No. 98304.

Puig V., Quevedo J., Escobet T., Morcego B., Ocampo C. (2004). Control Tolerante a Fallos (Parte I): Fundamentos y Diagnóstico de Fallos. *Revista Iberoamericana de automática e informática industrial*, volume (1), pp. 15-31.

Venkatasubramanian, V. (2003). A review of process fault detection and diagnosis. Part III: Process history based methods. *Computer and Chemical Engineering*, Volume (27.) pp. 15-31.

Walter, D. (1995). Controlling Centrifugal Pumps. *First published in Hydrocarbon Processing.*

SITIOS WEB

Neff R. *Cavitation in depth.*
http://www.irrigationcraft.com/diagnosing_cavitation.htm