Management Intelligent Systems Group
http://sicodinet.unileon.es

Propuesta de Módulo del Proyecto OSSIM: Desarrollo de un SEB para la Detección de Anomalias (*)

Enrique López González, Carlos Caño Alegre, David Sáenz Tagarro, Iago Trancón Barros

Technical Report #MISYG-2004-03
July, 2004

(*) *Este trabajo está soportado por el proyecto de investigación DPI 2001–0105 del MCT.*

Dept. of Management and Economics Business
Business and Economics Sciences Faculty. University of León
E-24071 León, SPAIN
Tel. & Fax: +34 987 291742

INDICE:

INTRODUCCION:

El punto de partida para la creación del sistema inteligente a desarrollar ha sido el proyecto de sourceforge denominado OSSIM (http://www.ossim.net). Este proyecto consiste en una distribución de productos open source integrados para construir una infraestructura de monitorización de seguridad. Su objetivo es ofrecer un marco para centralizar, organizar y mejorar las capacidades de detección y visibilidad en la monitorización de eventos de seguridad de la organización.

Un modo sencillo de ver la funcionalidad del OSSIM se refleja en el siguiente gráfico:

Durante las semanas previas a la realización del proyecto desarrollado se analizó la documentación del OSSIM, así como su código fuente, el diseño de su base de datos... todo ello para tratar de, en este enorme sistema, definir un tema de trabajo acorde con las posibilidades de los alumnos y en concordancia con el objetivo de la asignatura.

El tema elegido fue el desarrollo de un sistema inteligente orientado hacia los **DETECTORES DE ANOMALÍAS**. El tema, aparte de ser "viable" en su desarrollo (que se comentará a continuación), nos ha llamado fuertemente la atención. Explicaremos brevemente la utilidad de dichos detectores de anomalías:

La detección de anomalías puede ser especialmente útil para prevenir ataques perimetrales, estos son en sí una anomalía continua, en la dirección, el sentido de las comunicaciones, y el camino que definen, en el flujo de datos, el tamaño, el tiempo, el horario, el contenido, etc.
Veamos otros ejemplos en los que estos detectores serían útiles:

- Un nuevo ataque del cual no existen todavía firmas podría traspasar los sistemas de detección de patrones pero producir una anomalía clara.
- Los ataques internos, de empleados desleales desde dentro de nuestra red, no implican la violación de ninguna política ni la ejecución de ningún exploit. Implican sin embargo una anomalía en el uso y la forma de uso de un servicio.

- Un gusano que se ha introducido en la organización, un ataque de spamming, o el mismo uso de programas P2P, generarían un número de conexiones anómalas fácilmente detectable.
- Podremos detectar así mismo:
    - Usos de servicios anormales por origen y destino
    - Usos en horario anormal
    - Exceso en el uso de tráfico o conexiones
    - Copia anormal de ficheros en la red interna
    - Cambios en el sistema operativo de una máquina
    - Etc

Podemos pensar que como efecto negativo estos detectores generarán un número de nuevas alertas, amplificando nuestra señal y empeorando nuestro problema (nuestro objetivo es limitar el número de alertas), sin embargo si las tomamos como información adicional que acompaña a las clásicas alertas de patrones permitirá cualificar y por lo tanto diferenciar aquellas que puedan implicar una situación de mayor de riesgo.

ASPECTOS IMPORTANTES PARA LA IMPLEMENTACION DEL SISTEMA INTELIGENTE:

¿Qué es un detector?

Definiremos un detector como cualquier programa capaz de procesar información en tiempo real, información normalmente a bajo nivel como tráfico o eventos de sistema y lanzar alertas ante la localización de situaciones previamente definidas.

La definición de estas situaciones se puede hacer de dos formas:

1. A través de patrones, o reglas definidas por el usuario
2. A través de grados de anomalía

Variables que influyen en la capacidad del detector

Para discutir sobre la capacidad de un detector la definiremos mediante 2 variables:

- *Sensibilidad*, o la capacidad de análisis, en profundidad y complejidad, que posee nuestro detector a la hora de localizar un posible ataque.
- *Fiabilidad*, como su nombre indica es el grado de certeza que nos ofrece nuestro detector ante el aviso de un posible evento.

Variables relacionadas con las dificultades de los detectores en relación con su sensibilidad y fiabilidad

- *Falsos Positivos*. La falta de fiabilidad en nuestros detectores es el causante de los falsos positivos, es decir alertas que realmente no corresponden con ataques reales.
- *Falsos Negativos*. La incapacidad de detección implicaría que un ataque es pasado por alto.

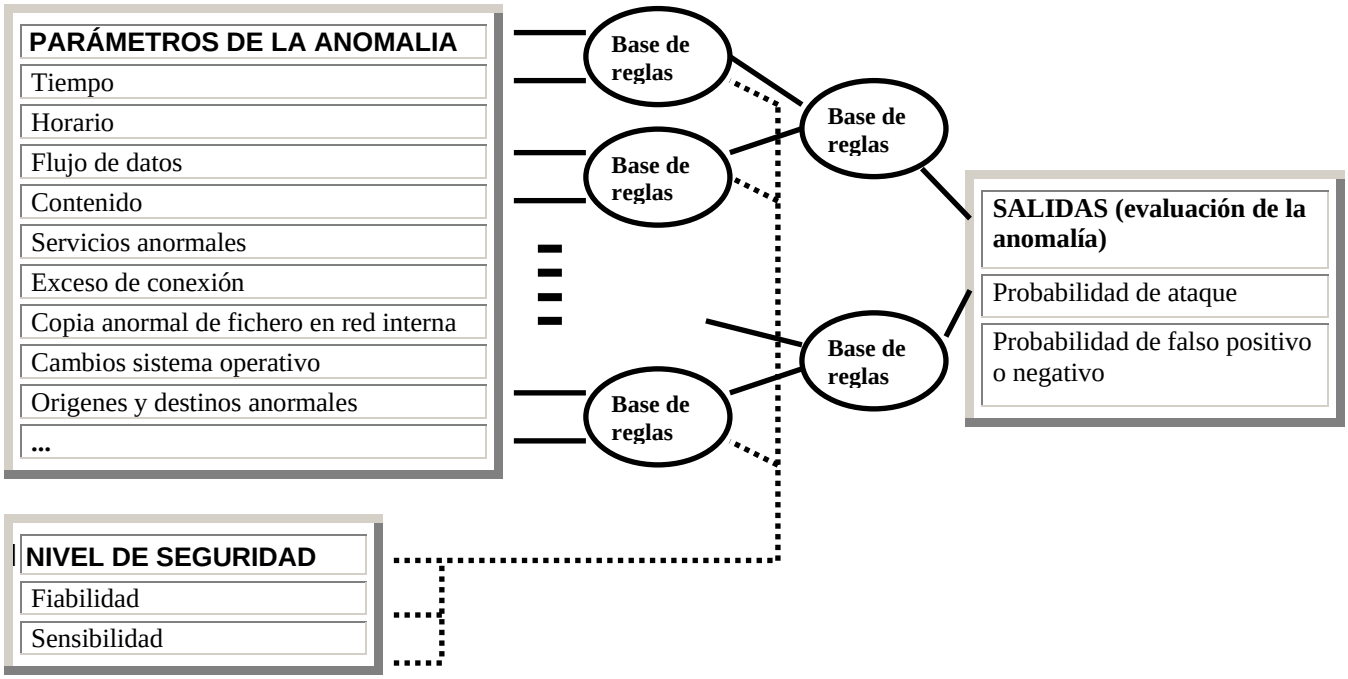
La relación existente entre las variables anteriores puede mostrarse a través del siguiente cuadro:

	Propiedad	**Efecto ante su ausencia**
Fiabilidad	El grado de certeza que nos ofrece nuestro detector ante el aviso de un posible evento.	Falsos Positivos
Sensibilidad	La capacidad de análisis, en profundidad y complejidad, que posee nuestro detector a la hora de localizar un posible ataque	Falsos Negativos

IMPLEMENTACION (1) - ESPECIFICACION DE LOS OBJETIVOS DEL SISTEMA INTELIGENTE DESARROLLADO

1. El sistema tomará como variables de entradas parámetros característicos de la "supuesta anomalía" detectada por otros módulos del OSSIM (que no serán implementados en nuestro sistema).
2. El sistema permitirá definir también como entrada el nivel de seguridad que se desea para el sistema, de forma que una misma anomalía pueda recibir diferente tratamiento según el nivel de seguridad que se establezca, es decir, una anomalía puede ser considerada un ataque en un nivel de seguridad "Alto", y puede considerarse una operación normal en un nivel de seguridad "Bajo".
3. El nivel de seguridad del sistema quedará definido en función del grado de fiabilidad y sensibilidad establecida.
4. El sistema devolverá 2 variables de salida:
    a. La probabilidad de que la anomalía seleccionada (que en nuestro caso será descrita a través de las opciones de interfaz que se detallen) sea realmente un ataque a la seguridad del sistema.
    b. Si la probabilidad de que la anomalía sea un ataque al sistema es mayor del 50%, entonces se indicará la probabilidad de que el ataque sea un "falso positivo". Si la probabilidad de que la anomalía sea un ataque al sistema es menor del 50%, entonces se indicará la probabilidad de que el ataque sea un "falso negativo".

A continuación mostraremos las principales variables de entrada que se han contemplado para la elaboración del sistema. Queremos señalar que los parámetros que determinan si las entradas definen un ataque al sistema los determinaremos durante el desarrollo.



IMPLEMENTACION (2) - DESARROLLO DE LAS BURBUJAS DEL SISTEMA INTELIGENTE

Teniendo en cuenta el objetivo de nuestro sistema inteligente (ser un detector de anomalías), con vista al desarrollo del sistema inteligente, se han tomado algunas decisiones al modo en que se definir{an las burbujas del sistema, es decir, se han tomado decisiones acerca del modo en que se definirán las reglas que determinarán cómo se modificarán las variables de entrada para obtener las variables de salida.

Emplearemos básicamente cuatro modelos estadísticos en la elaboración del sistema. Estos modelos son:

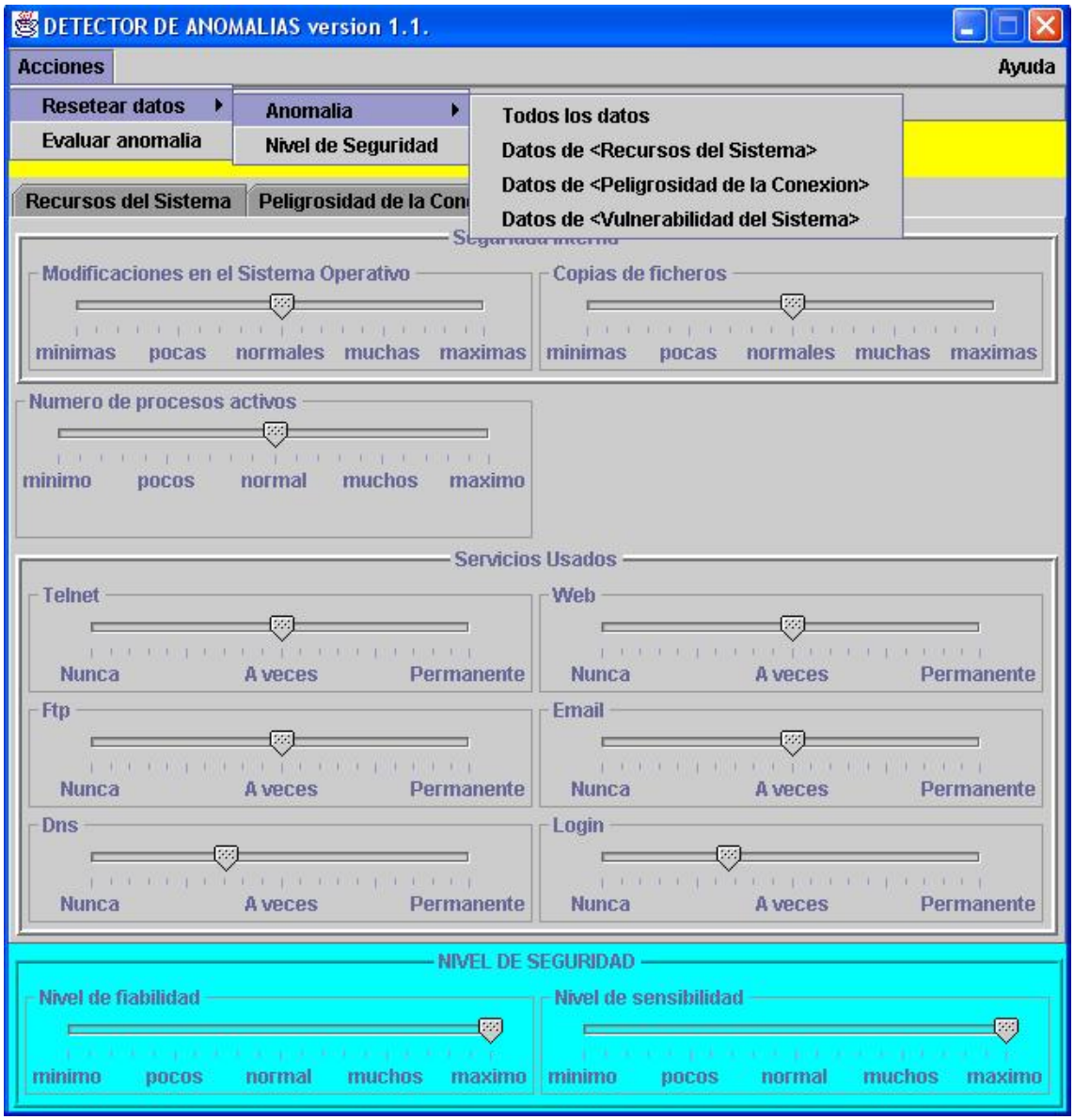
1. **Modelo operacional:** Se aplica sobre eventos para determinar por ejemplo, el número de intentos de acceso fallidos al sistema. Compara su métrica con un valor umbral, que normalmente le indica si se ha cometido una intrusión. Este modelo también es aplicable a la detección de usos indebidos, y es el desarrollado en la detección de umbral, explicada más adelante.
2. **Modelo de desviación media y estándar:** Este modelo aplica el concepto de desviación media y estándar típico a la hora de elaborar los perfiles de comportamiento. Supone que a partir de las dos primeras medidas, se puede establecer un intervalo de confianza. Este intervalo sirve para determinar la desviación estándar. Si las siguientes medidas caen fuera de este intervalo revelarían un comportamiento anormal.
3. **Modelo *multivariable:*** Es una ampliación del modelo de desviación media y estándar. Utiliza correlaciones entre dos o más métricas para definir un comportamiento. Por ejemplo, en vez de determinar la conducta de un usuario exclusivamente por la duración de su sesión, también se tiene en cuenta la actividad de tráfico de red que genera durante la misma.

Modelo del Proceso Markov: **Este es el modelo más complejo de los cuatro. Considera cada evento de auditoría como una variable de estados, y utiliza una matriz de transiciones de estados para describir la frecuencia de transiciones de estados (no la de cada evento, sino la de todos juntos). Un evento determinado indica una anomalía si su probabilidad es demasiado baja.**

FUNCIONAMIENTO DEL PROGRAMA

La interfaz del programa esta desarrollada en Java, y permite establecer los valores de todas las variables con facilidad gracias al uso de barras de scroll.

El programa consta de dos submenus: el menú ayuda donde se puede observar a los dos creadores de esta aplicación y el menú Acciones desde donde se puede o bien resetear los datos introducidos, estableciendose los parámetros por defecto , o bien evaluar el sistema con los datos introducidos.

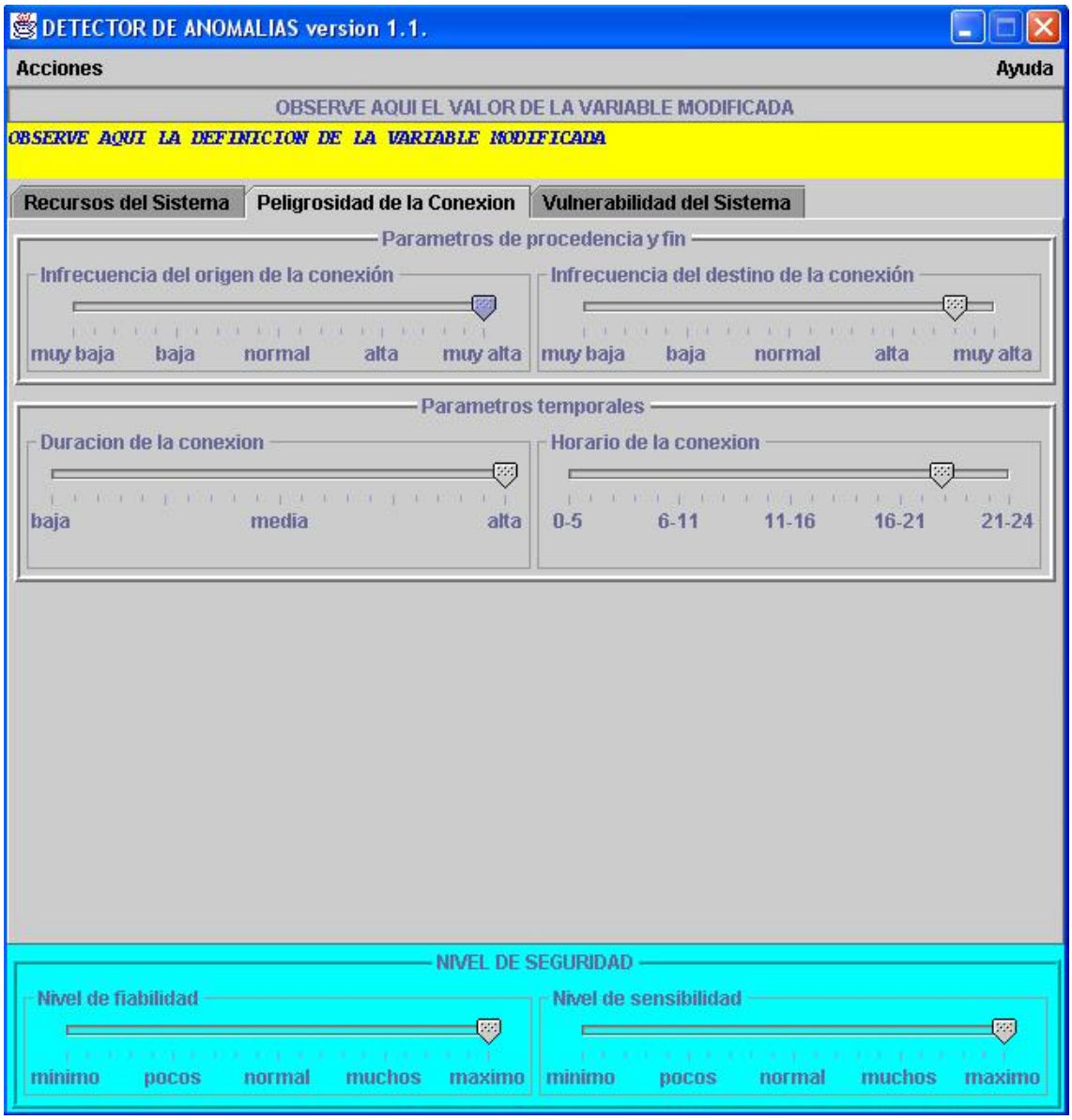


La pantalla principal posee tres fichas independientes que nos permiten modificar las diferentes variables, englobadas en Recursos del sistema, Peligrosidad de la conexión y Vulnerabilidad del Sistema.

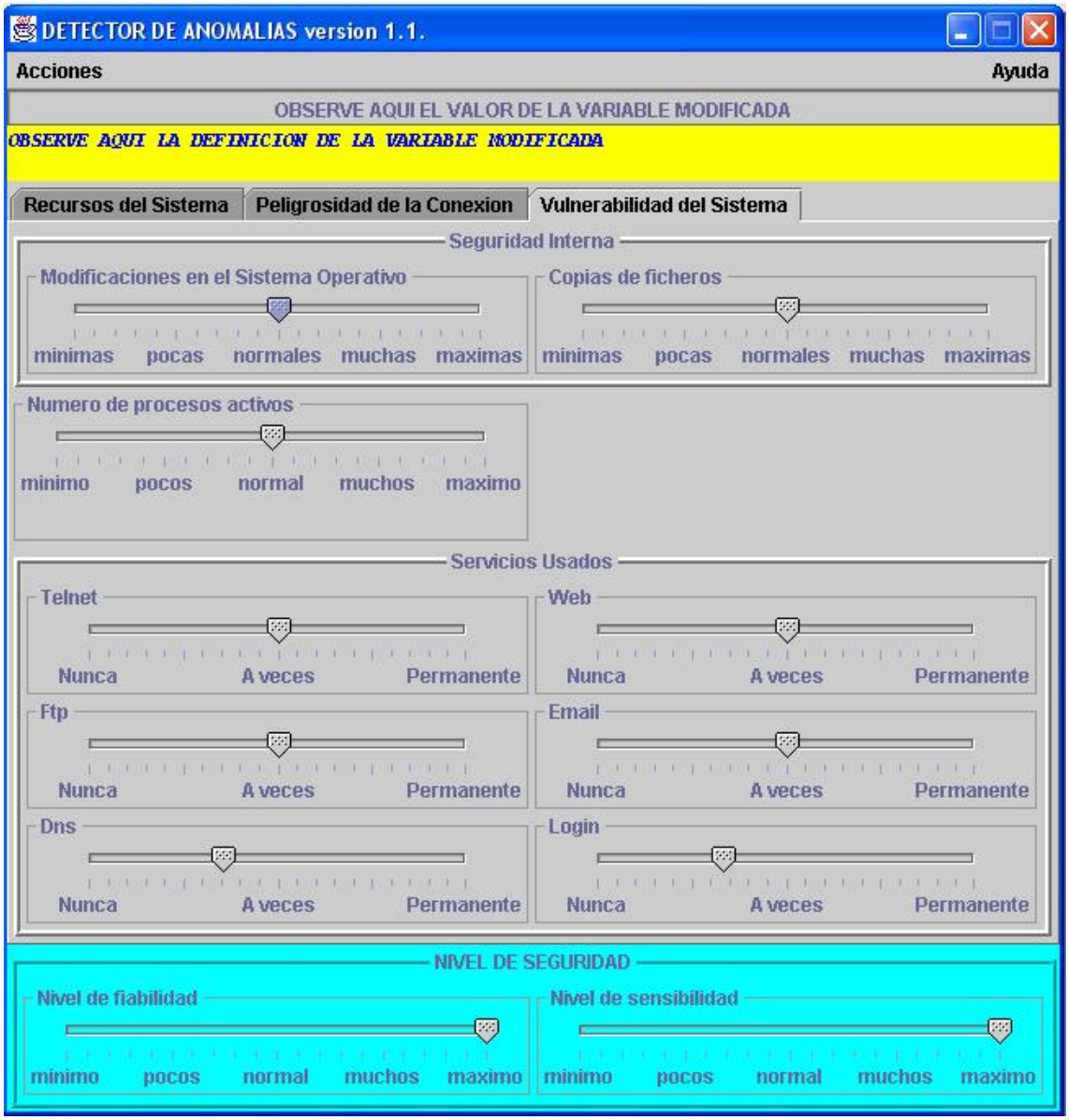
Ademas de estas tres fichas, en la aplicación se pueden observar tres paneles adicionales. Uno de ellos es donde se visualiza el valor de la variable que se esta modificando, otro donde se puede observar el significado de la variable que se modifica y por ultimo el panel inferior que se puede observar constantemente, donde se visualiza el nivel de seguridad del sistema.

En la primera ficha Recursos del Sistema, podemos modificar los parámetros de Hardware del ordenador y los parámetros de la red.

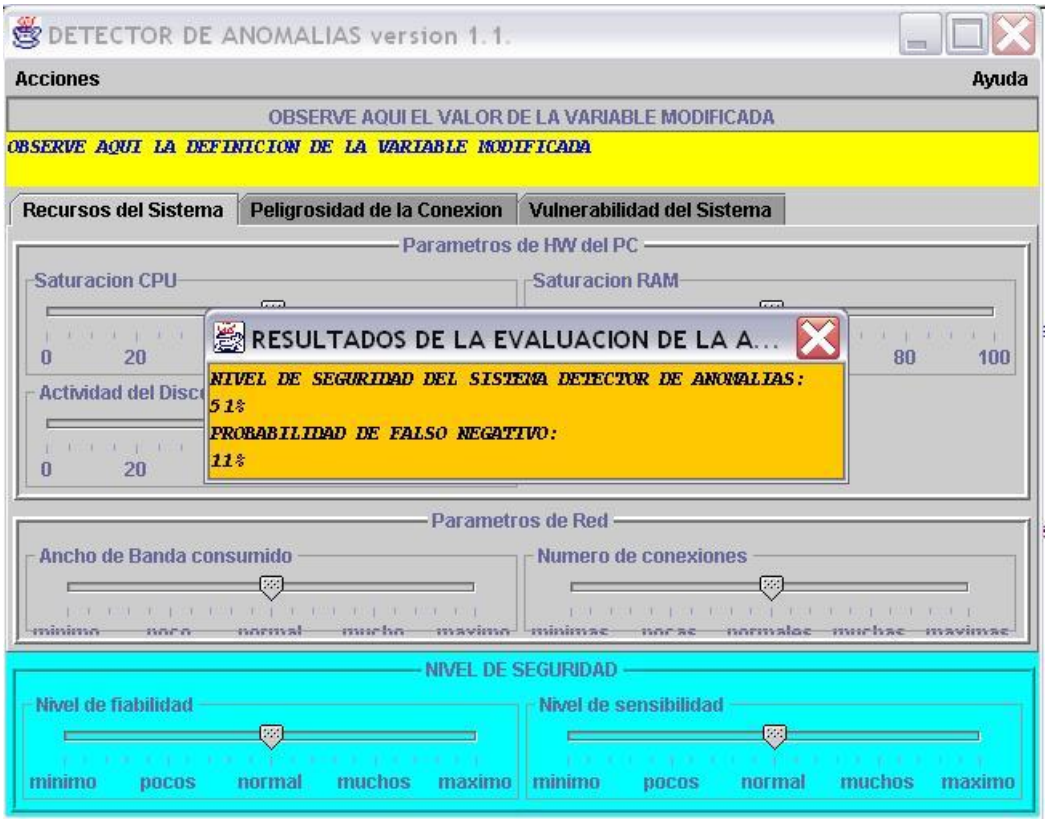
En la segunda ficha Peligrosidad de la Conexión se modifican los valores de procedencia y fin de la conexión y los parámetros temporales.



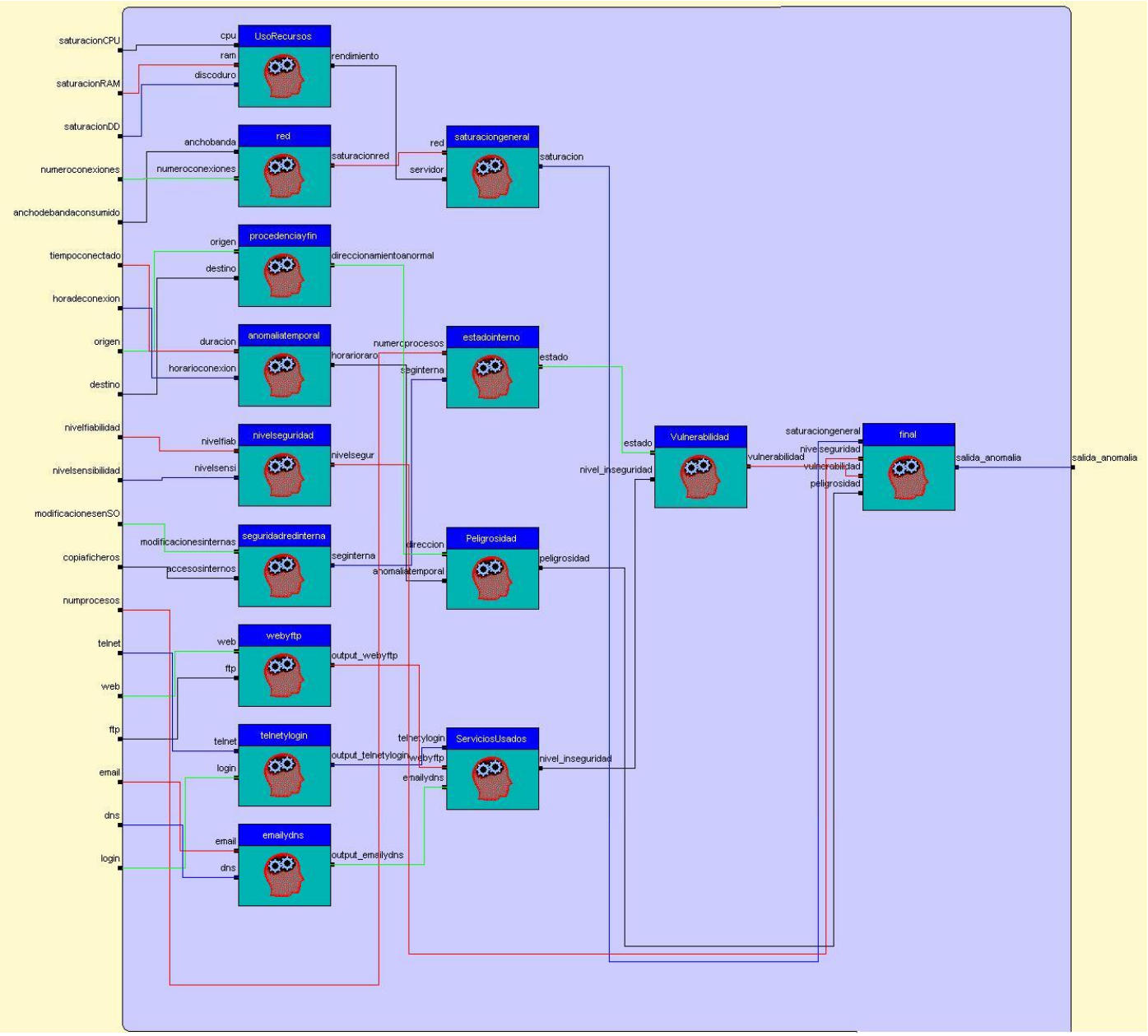
Por ultimo en la tercera ficha modificamos y seleccionamos el nivel de seguridad interna y los servicios que estan disponibles en el ordenador a controlar.



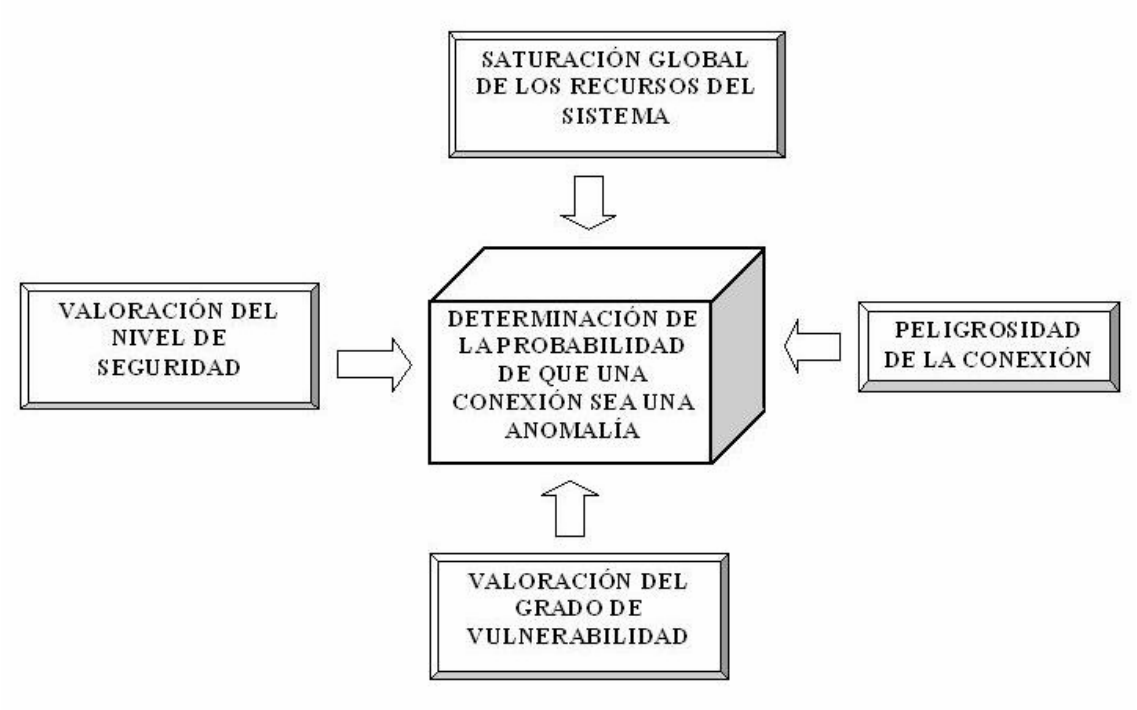
Una vez seleccionados todos estos parámetros se ejecuta la opcion evaluar anomalia que nos permite saber la probabilidad de que ocurra una anomalia en el sistema y tambien nos muestra la probabilidad de que ocurra un falso positivo o un falso negativo.

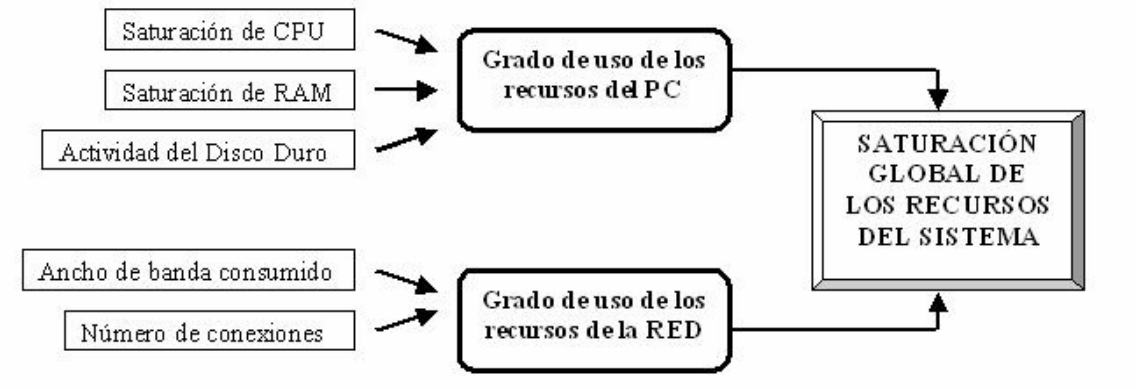

DETECTOR DE ANOMALIAS version 1.1.
Acciones
Ayuda
OBSERVE AQUI EL VALOR DE LA VARIABLE MODIFICADA
OBSERVE AQUI LA DEFINICION DE LA VARIABLE MODIFICADA
Recursos del Sistema
Peligrosidad de la Conexion
Vulnerabilidad del Sistema
Parametros de HW del PC
Saturacion CPU
Saturacion RAM
0
20
80
100
Actividad del Disc
0
20
RESULTADOS DE LA EVALUACION DE LA A...
NIVEL DE SEGURIDAD DEL SISTEMA DETECTOR DE ANOMALIAS:
51%
PROBABILIDAD DE FALSO NEGATIVO:
11%
Parametros de Red
Ancho de Banda consumido
Numero de conexiones
NIVEL DE SEGURIDAD
Nivel de fiabilidad
minimo
pocos
normal
muchos
maximo
Nivel de sensibilidad
minimo
pocos
normal
muchos
maximo

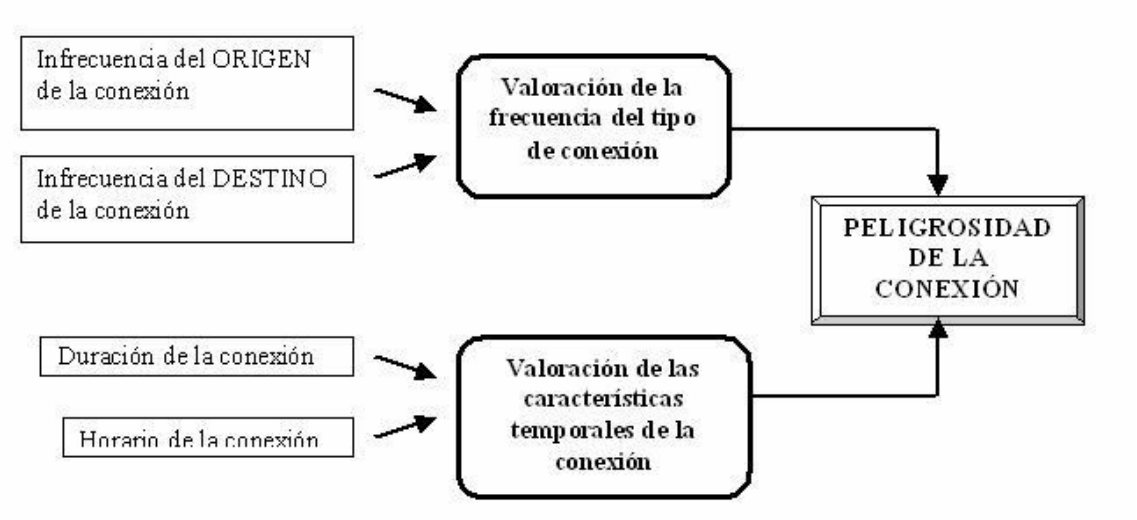
ESQUEMA DEL SISTEMA BORROSO

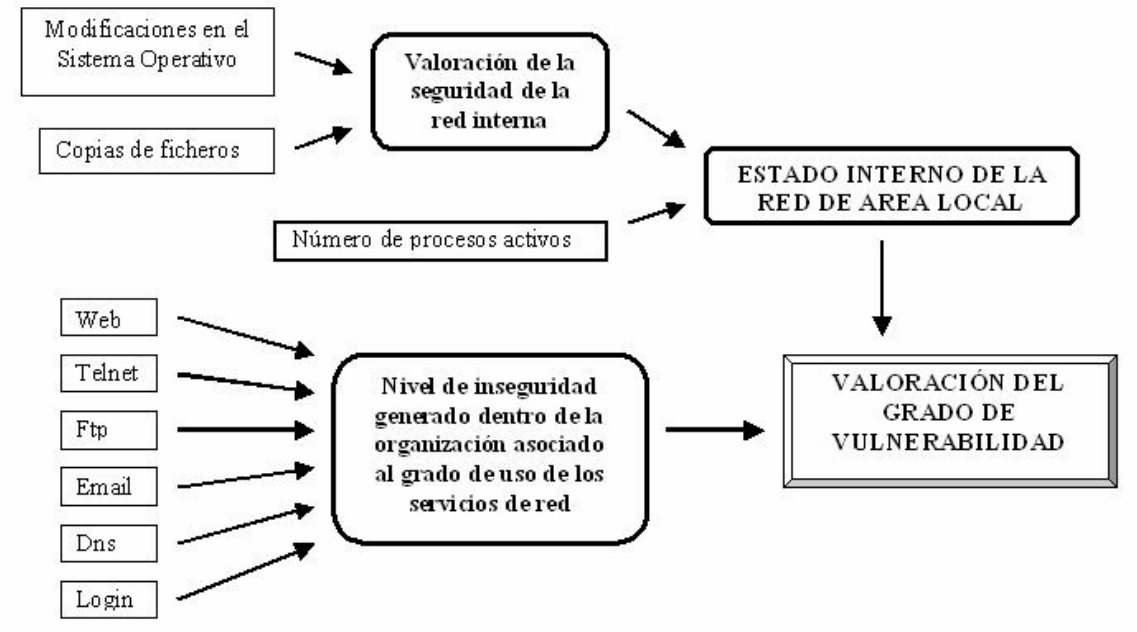


DISEÑO DEL SISTEMA BORROSO






Infrecuencia del ORIGEN de la conexión
Infrecuencia del DESTINO de la conexión
Valoración de la frecuencia del tipo de conexión
PELIGROSIDAD DE LA CONEXIÓN
Duración de la conexión
Horario de la conexión
Valoración de las características temporales de la conexión


Modificaciones en el Sistema Operativo
Copias de ficheros
Valoración de la seguridad de la red interna
ESTADO INTERNO DE LA RED DE AREA LOCAL
Número de procesos activos
Web
Telnet
Ftp
Email
Dns
Login
Nivel de inseguridad generado dentro de la organización asociado al grado de uso de los servicios de red
VALORACIÓN DEL GRADO DE VULNERABILIDAD

VARIABLES

En este sistema de logica difusa hemos utilizado una serie de variables que consideramos prioritarias para la evaluacion de anomalias, estas son:

VARIABLES DE ENTRADA:

saturacion_CPU
indica el Nivel de utilizacion en operaciones del CPU.

saturacion_RAM
indica la Memoria que se esta utilizando en el pc.

actividad_DD
indica el Nivel de transacciones de lectura y escritura al Disco Duro.

ancho_banda_consumido
indica la Franja consumida de la capacidad de transmision de la linea de conexión a internet

numero_conexiones
indica el Numero de peticiones y envios desde el pc y hacia el pc respectivamente

origen_anormal
es la Clasificacion del origen de una conexion establecida segun si es un usuario conocido y con permiso o no.

destino_anormal
Destino de la conexion depedendiendo de si es un destino comun o extraño

duracion_conexion
Tiempo que permanece abierta la conexión

horario_conexion
Clasificacion dependiendo del horario. La peligrosidad de un ataque nocturno siempre será mayor que el de uno diurno

numero_procesos_activos
Numero de procesos que se ejecutan concurrentemente en el ordenador

modificaciones_SO
Alteraciones en archivos fundamentales del sistema, asi como en archivos de recopilacion de actividades de usuarios

copia_ficheros
Copia o acceso a ficheros de seguridad del sistema , como pueden ser los archivos de claves o bases de datos

nivel_fiabilidad
El grado de certeza que nos ofrece nuestro detector ante el aviso de un posible evento

nivel_sensibilidad
La capacidad de análisis, en profundidad y complejidad, que posee nuestro detector a la hora de localizar un posible ataque.

telnet
Servicio de consola remota

web
Este servicio esta activo si el ordenador analizado es un servidor web

ftp
Servicio activo si existe un servidor de ftp en el ordenador

Email
Servicio activo si en el ordenador existe un servidor de correo

Dns
Servicio activo si el ordenador actua como servidor de direcciones en internet o servidor de DNS

Login
Posibilidad de acceso a los recursos del ordenador mediante cuenta con contraseñas.

VARIABLES DE SALIDA

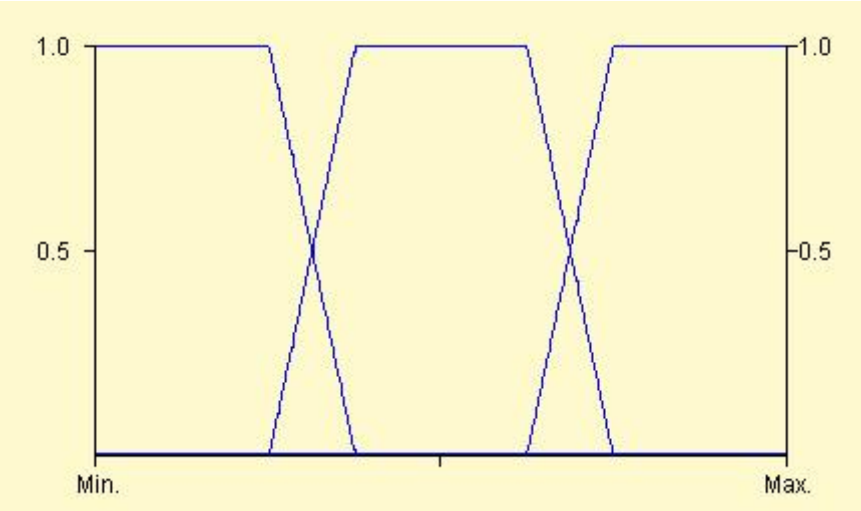
Probabilidad_anomalia
Probabilidad de que ocurra una anomalia en el sistema, se calcula a partir de todas las variables de entrada.

TIPOS DE DATOS

Para la creación del sistema hemos construido los siguientes tipos de datos:

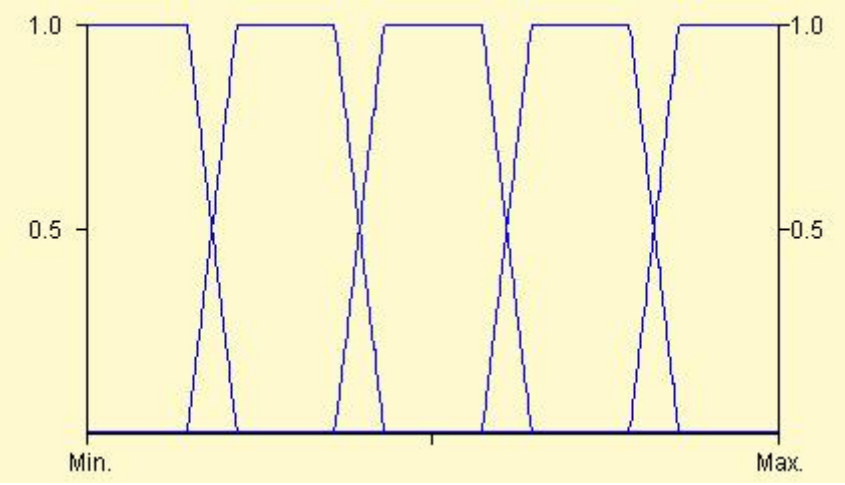
Tipo Tiempo

Puede tomar los siguientes valores: bajo, medio, alto

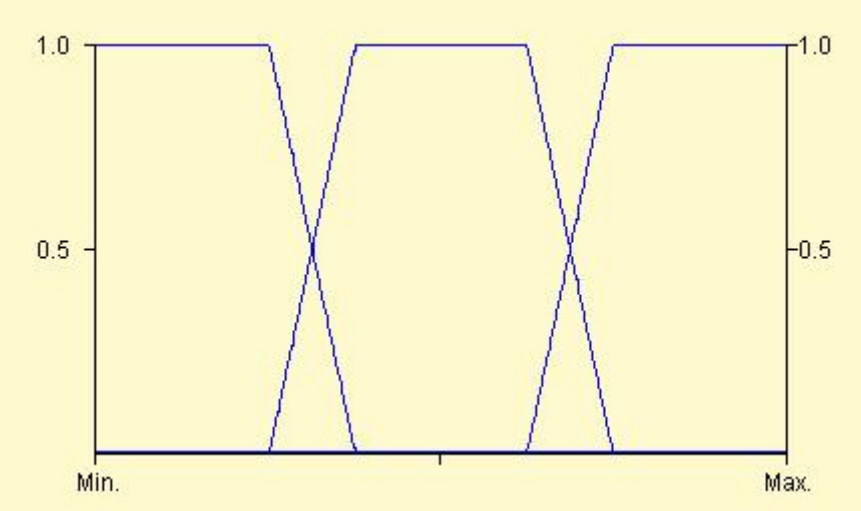


Tipo NumConex

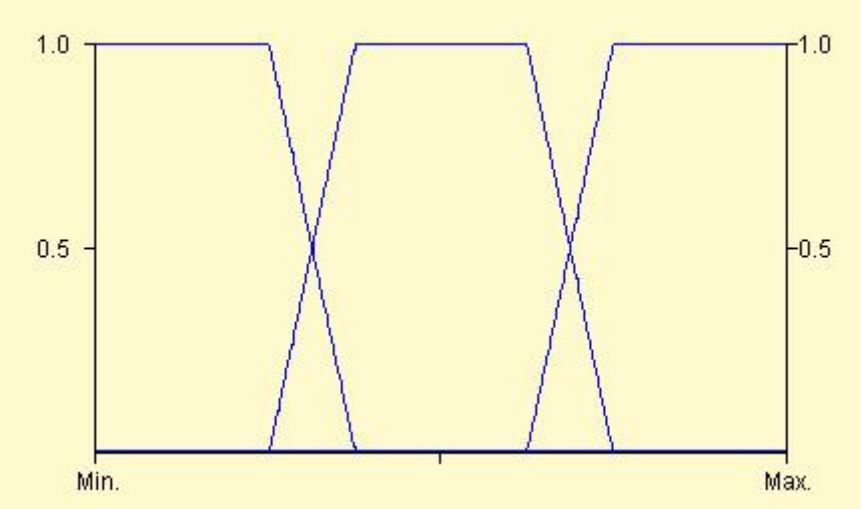
Puede tomar los valores: minimas, pocas, normales, muchas, maximas.



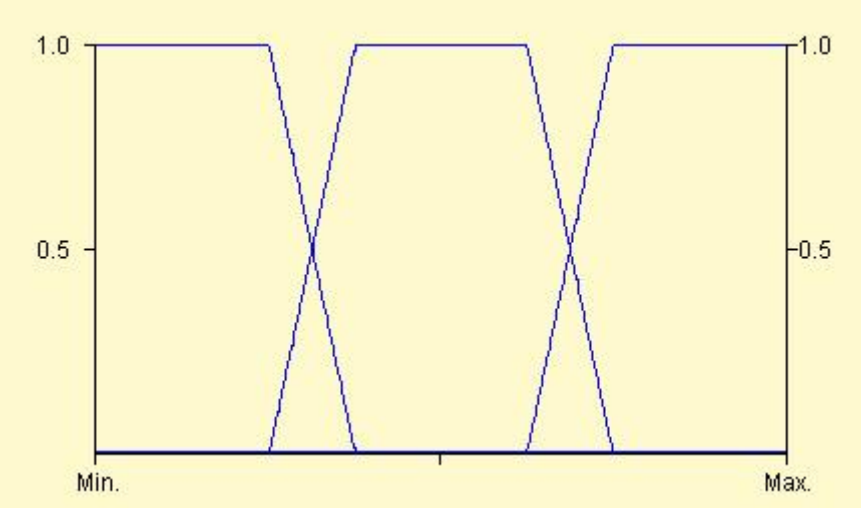
Tipo ProbabilidadAnomalia
Puede tomar los valores: poca, normal , alta



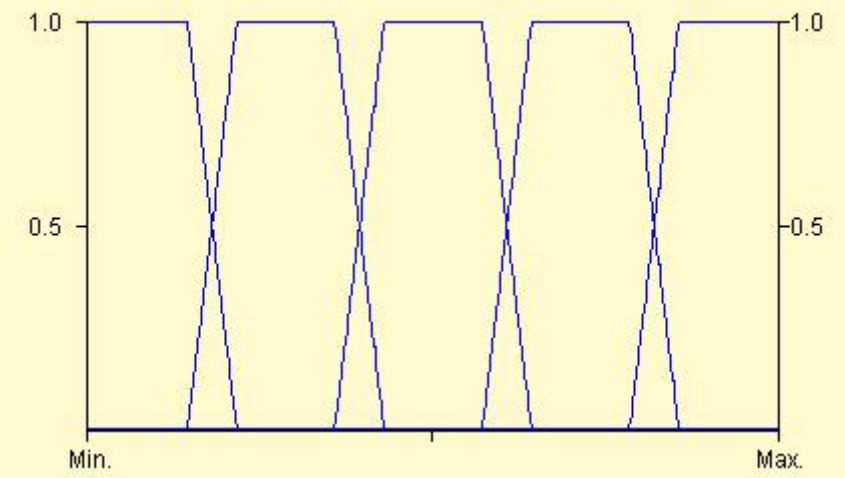
Tipo ProbabilidadFalsos
Puede tomar los valores: poca, normal , alta



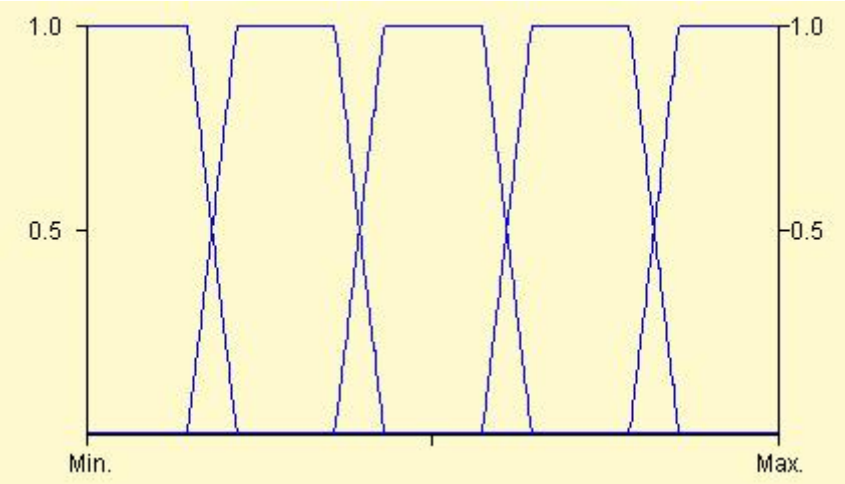
Tipo NivelSeguridad
Puede tomar los valores : bajo, normal, alto



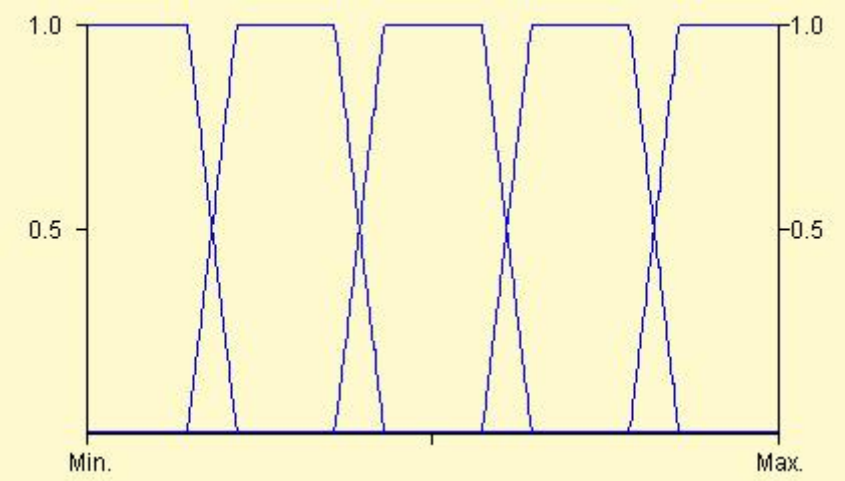
Tipo AnchodebandaCons
Puede tomar los valores: minimas, pocas, normales, muchas, maximas.



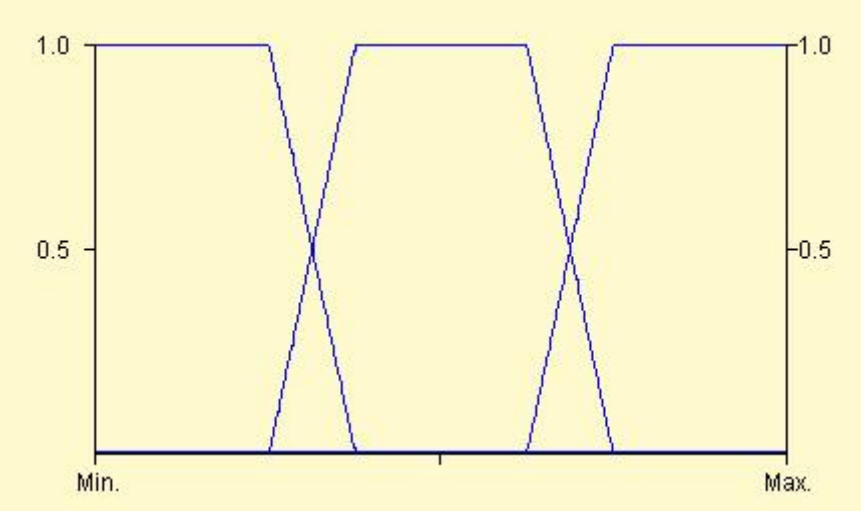
Tipo NivelSaturacionRed
Puede tomar los valores: minimasaturacion, pocasaturacion, normalsaturacion, muchasaturacion, maximasaturacion.



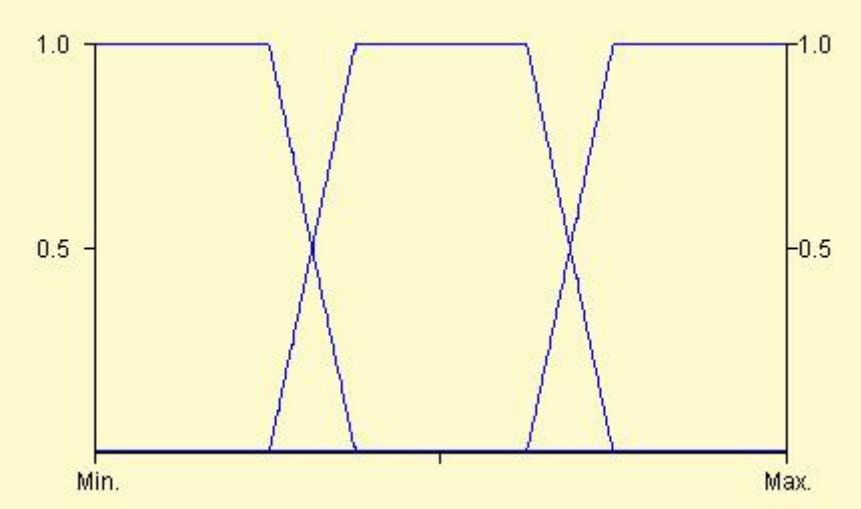
Tipo horario
Puede tomar los valores: 0-5, 6-11, 11-16, 16-21, 21-24.



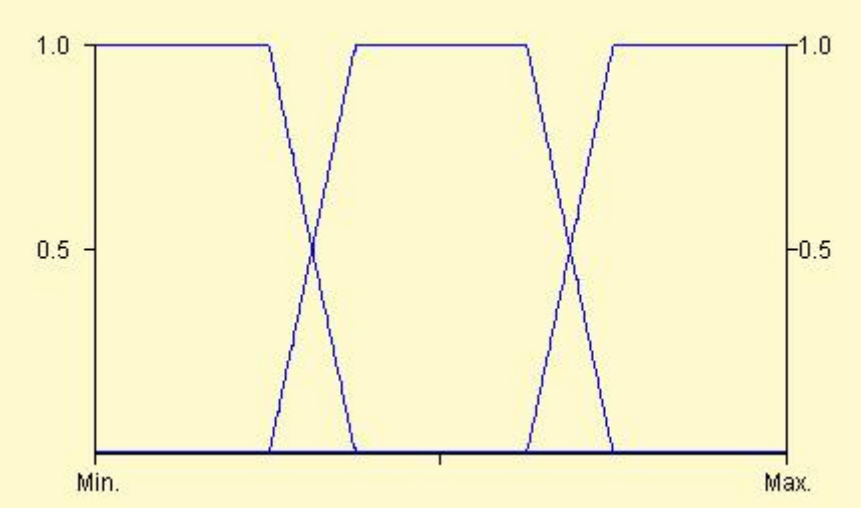
Tipo ConsumoCPU
Puede tomar los valores : bajo, normal, alto



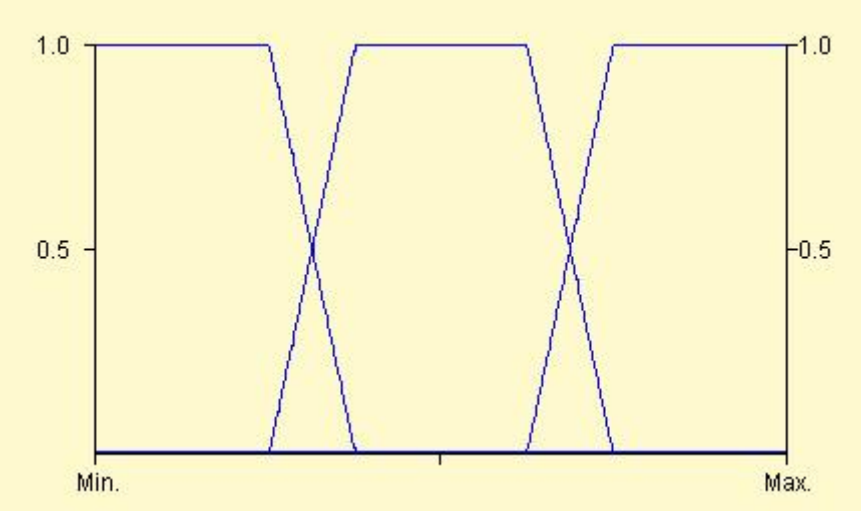
Tipo ConsumoRAM
Puede tomar los valores : bajo, normal, alto



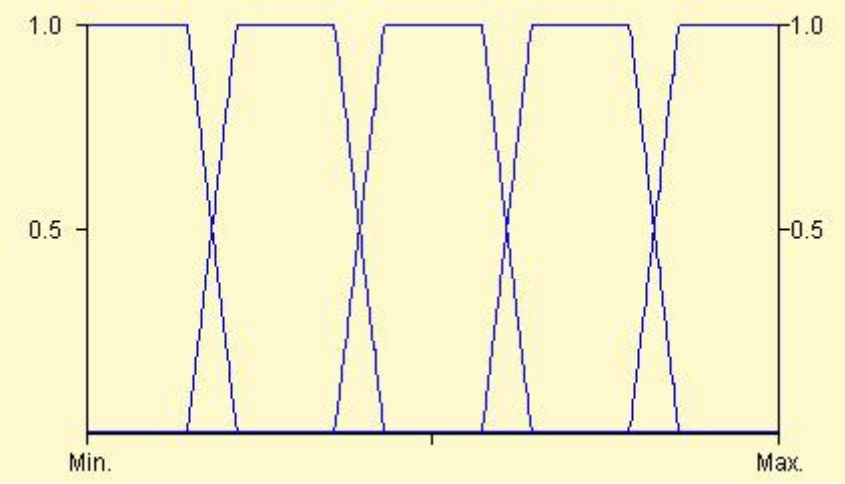
Tipo ActividadDiscoDuro
Puede tomar los valores : bajo, normal, alto



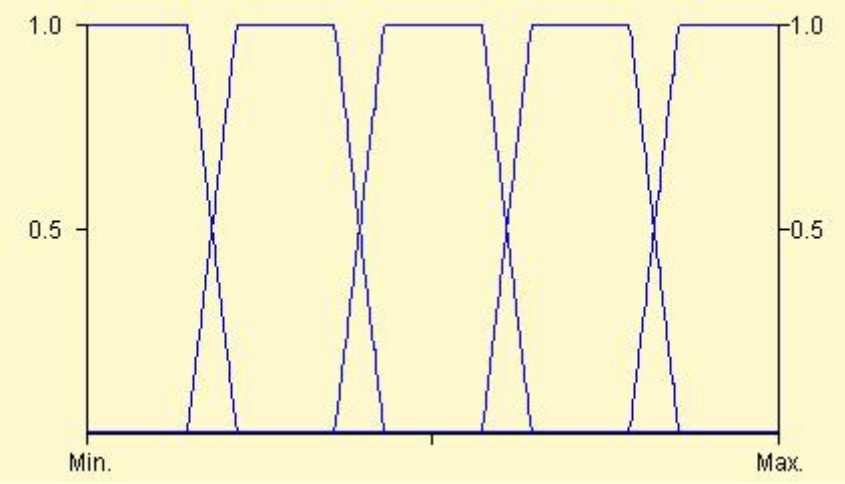
Tipo NumerodePuertos
Puede tomar los valores : bajo, normal, alto



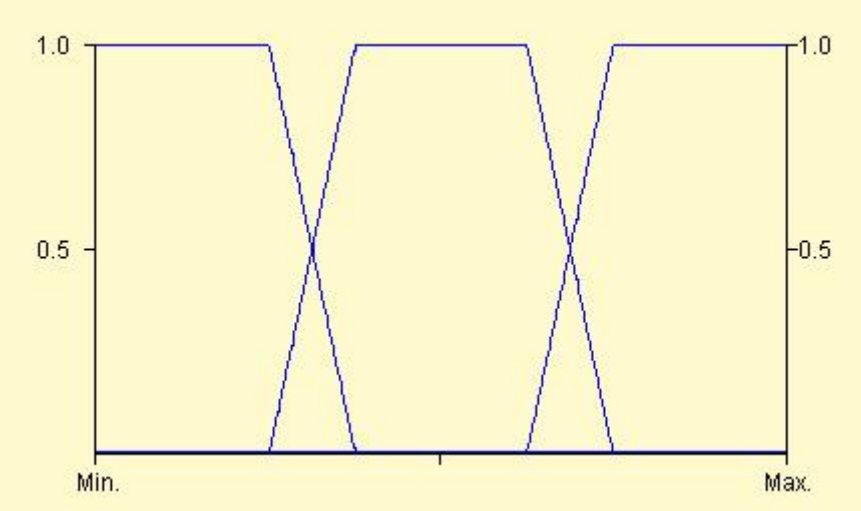
Tipo OrigenAnormal
Puede tomar los valores: minimas, pocas, normales, muchas, maximas.



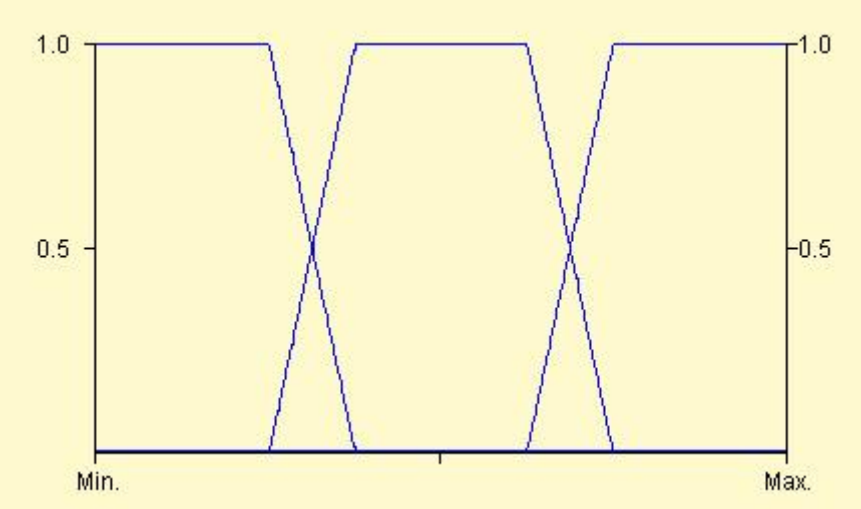
Tipo DestinoAnormal
Puede tomar los valores: minimas, pocas, normales, muchas, maximas.



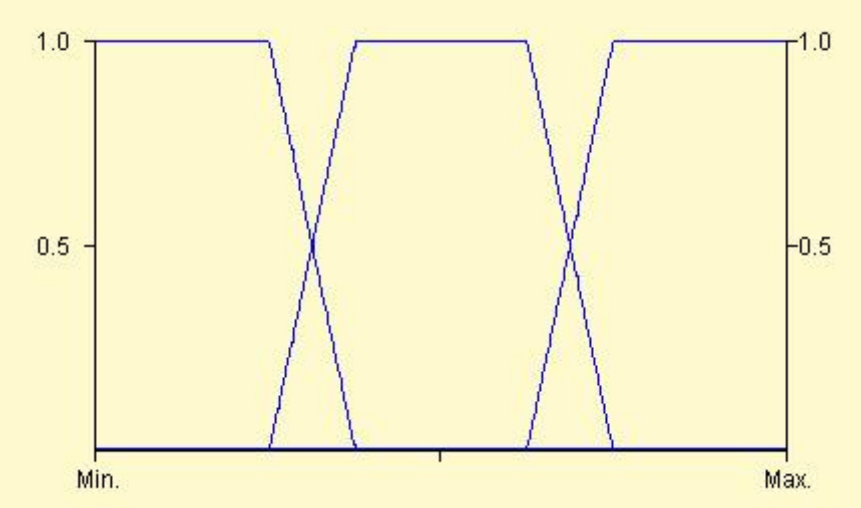
Tipo Fiabilidad
Puede tomar los valores : bajo, normal, alto



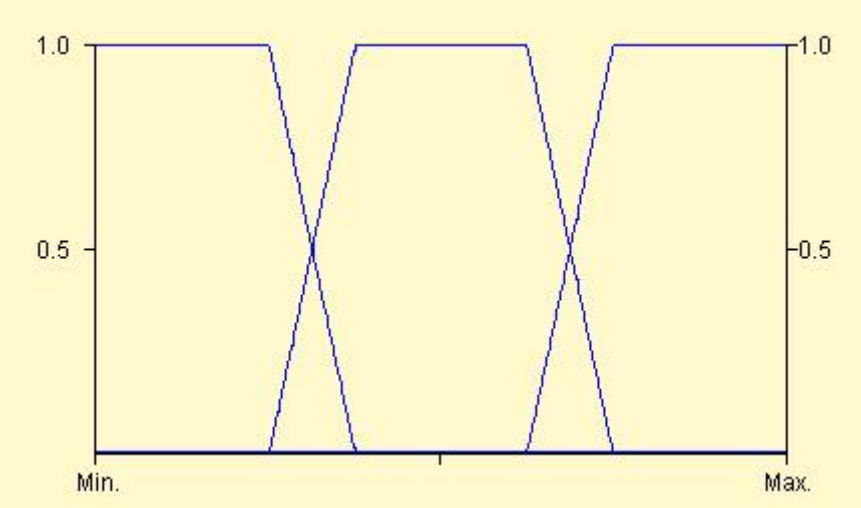
Tipo Sensibilidad
Puede tomar los valores : bajo, normal, alto



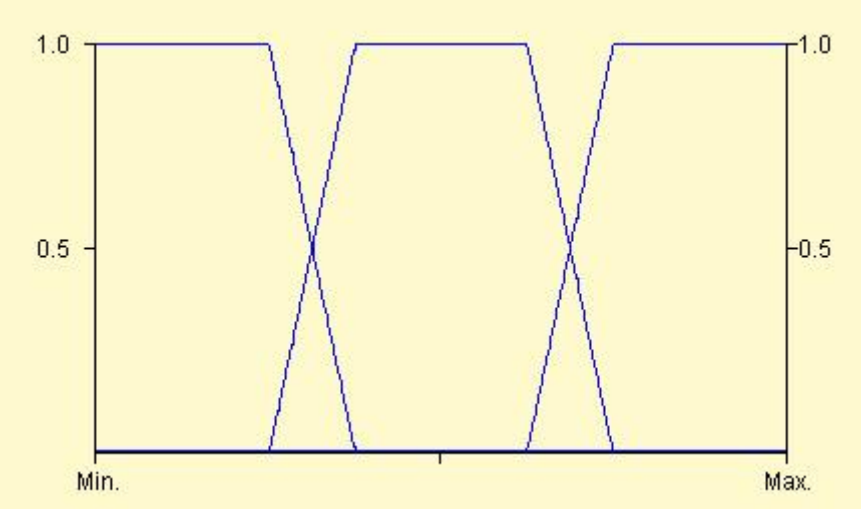
Tipo CambiosSO
Puede tomar los valores : bajo, normal, alto



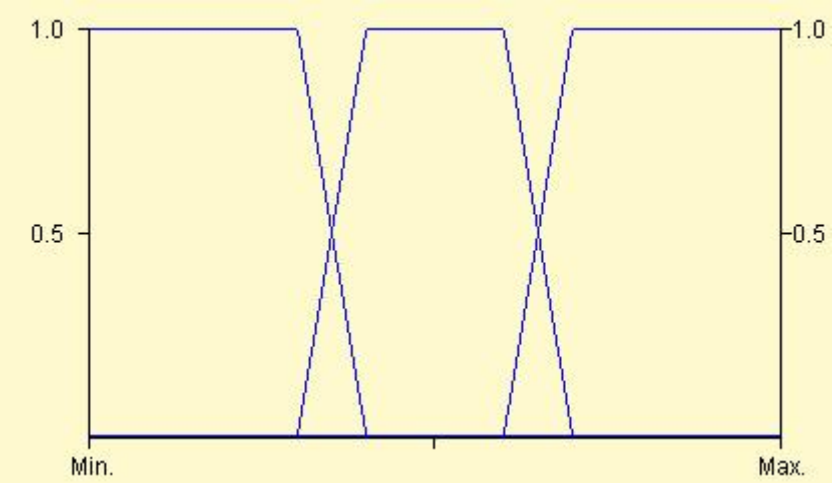
Tipo CopiaRedInterna
Puede tomar los valores : bajo, normal, alto



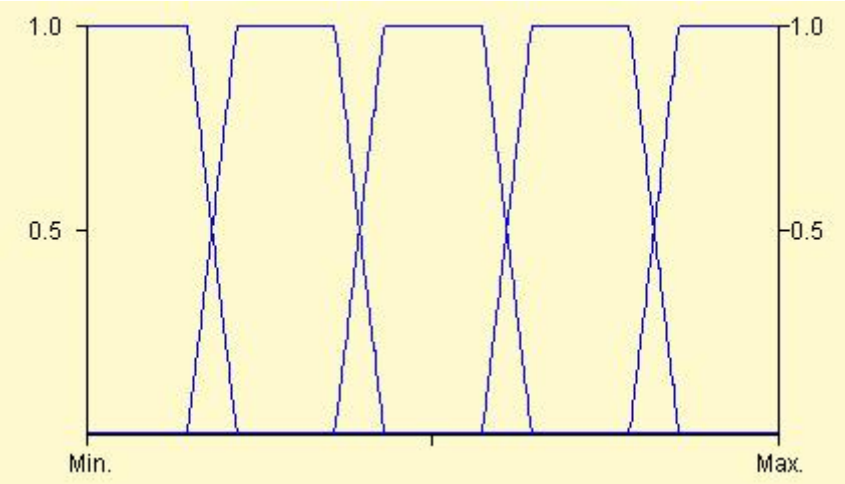
Tipo EstadoInterno
Puede tomar los valores : bajo, normal, alto



Tipo Ternario
Puede tomar los valores : nunca_conectado, a_intervalos, siempre_conectado



Tipo ProcEnEjec
Puede tomar los valores: minimas, pocas, normales, muchas, maximas.



REGLAS

Ahora mostraremos el conjunto de reglas utilizadas en el sistemas, las reglas con dos entradas y una salida se muestran en forma de matriz, el resto en forma de condiciones o conjuntos de reglas.

Red

Entrada: numeroconexiones y anchobanda.
Salida: saturacionred

	muypocas	pocas	normales	muchas	demasiadas
muypoco	muypocasaturacion	muypocasaturacion	pocasaturacion	normalsaturacion	demasiadasaturacion
poco	pocasaturacion	pocasaturacion	normalsaturacion	muchasaturacion	demasiadasaturacion
normal	pocasaturacion	normalsaturacion	muchasaturacion	muchasaturacion	demasiadasaturacion
mucho	muchasaturacion	muchasaturacion	muchasaturacion	muchasaturacion	demasiadasaturacion
demasiado	demasiadasaturacion	demasiadasaturacion	demasiadasaturacion	demasiadasaturacion	demasiadasaturacion

UsoRecursos

Entrada: cpu, ram, discoduro.
Salida: rendimiento

Rule			cpu		ram		discoduro		rendimiento
0	1.0	if	cpu == bajo	&	ram == bajo	&	discoduro == bajo	->	rendimiento = bajo
1	1.0	if	cpu == bajo	&	ram == medio	&	discoduro == bajo	->	rendimiento = bajo
2	1.0	if	cpu == bajo	&	ram == alto	&	discoduro == bajo	->	rendimiento = bajo
3	1.0	if	cpu == medio	&	ram == bajo	&	discoduro == bajo	->	rendimiento = bajo
4	1.0	if	cpu == medio	&	ram == medio	&	discoduro == bajo	->	rendimiento = medio
5	1.0	if	cpu == medio	&	ram == alto	&	discoduro == bajo	->	rendimiento = medio
6	1.0	if	cpu == alto	&	ram == bajo	&	discoduro == bajo	->	rendimiento = bajo
7	1.0	if	cpu == alto	&	ram == medio	&	discoduro == bajo	->	rendimiento = medio
8	1.0	if	cpu == alto	&	ram == alto	&	discoduro == bajo	->	rendimiento = alto
9	1.0	if	cpu == bajo	&	ram == bajo	&	discoduro == medio	->	rendimiento = bajo
10	1.0	if	cpu == bajo	&	ram == medio	&	discoduro == medio	->	rendimiento = medio
11	1.0	if	cpu == bajo	&	ram == alto	&	discoduro == medio	->	rendimiento = medio
12	1.0	if	cpu == medio	&	ram == bajo	&	discoduro == medio	->	rendimiento = medio
13	1.0	if	cpu == medio	&	ram == medio	&	discoduro == medio	->	rendimiento = medio
14	1.0	if	cpu == medio	&	ram == alto	&	discoduro == medio	->	rendimiento = medio
15	1.0	if	cpu == alto	&	ram == bajo	&	discoduro == medio	->	rendimiento = medio
16	1.0	if	cpu == alto	&	ram == medio	&	discoduro == medio	->	rendimiento = medio
17	1.0	if	cpu == alto	&	ram == alto	&	discoduro == medio	->	rendimiento = alto
18	1.0	if	cpu == bajo	&	ram == bajo	&	discoduro == alto	->	rendimiento = bajo
19	1.0	if	cpu == bajo	&	ram == medio	&	discoduro == alto	->	rendimiento = medio
20	1.0	if	cpu == bajo	&	ram == alto	&	discoduro == alto	->	rendimiento = alto
21	1.0	if	cpu == medio	&	ram == bajo	&	discoduro == alto	->	rendimiento = medio
22	1.0	if	cpu == medio	&	ram == medio	&	discoduro == alto	->	rendimiento = medio
23	1.0	if	cpu == medio	&	ram == alto	&	discoduro == alto	->	rendimiento = alto
24	1.0	if	cpu == alto	&	ram == bajo	&	discoduro == alto	->	rendimiento = alto
25	1.0	if	cpu == alto	&	ram == medio	&	discoduro == alto	->	rendimiento = alto
26	1.0	if	cpu == alto	&	ram == alto	&	discoduro == alto	->	rendimiento = alto
*									

SaturacionGeneral

Entrada: red y servidor.
Salida: saturación

	bajo	medio	alto
bajo	bajo	medio	alto
medio	medio	medio	alto
alto	alto	alto	alto

Procedencia y fin

Entrada: origen y destino.
Salida: direccionamiento anormal

	muypoco	poco	normal	mucho	demasiado
muypoco	muypoco	poco	normal	normal	demasiado
poco	poco	poco	normal	mucho	demasiado
normal	normal	normal	normal	normal	demasiado
mucho	normal	mucho	mucho	mucho	demasiado
demasiado	demasiado	demasiado	demasiado	demasiado	demasiado

AnomaliaTemporal

Entrada: duracion y horarioconexion.
Salida: horarioraro

	hora0a5	hora6a11	hora11a16	hora16a21	hora21a24
bajo	medio	bajo	bajo	bajo	medio
medio	alto	medio	bajo	medio	alto
alto	alto	medio	medio	medio	alto

NivelSeguridad

Entrada: nivelfaibilidad y nivelsensibilidad.
Salida: nivelsegur

	bajo	medio	alto
bajo	bajo	bajo	bajo
medio	bajo	medio	medio
alto	bajo	medio	alto

SeguridadRedInterna

Entrada: modificacionesinternas y accesosinternos
Salida: seginterna

	bajo	medio	alto
bajo	alto	medio	bajo
medio	medio	medio	bajo
alto	bajo	bajo	bajo

Final

Entrada: saturaciongeneral, nivelseguridad, vulnerabilidad y peligrosidad.

Salida: salida_anomalia

Rule			Premise		Conclusion
0	1.0	if	(saturaciongeneral == bajo & nivelseguridad == bajo & vulnerabilidad == bajo & peligrosidad == bajo)	->	salida_anomalia = poca
1	1.0	if	(saturaciongeneral == bajo & nivelseguridad == bajo & vulnerabilidad == medio & peligrosidad == bajo)	->	salida_anomalia = poca
2	1.0	if	(saturaciongeneral == bajo & nivelseguridad == bajo & vulnerabilidad == alto & peligrosidad == bajo)	->	salida_anomalia = normal
3	1.0	if	(saturaciongeneral == bajo & nivelseguridad == medio & vulnerabilidad == bajo & peligrosidad == bajo)	->	salida_anomalia = poca
4	1.0	if	(saturaciongeneral == bajo & nivelseguridad == medio & vulnerabilidad == medio & peligrosidad == bajo)	->	salida_anomalia = poca
5	1.0	if	(saturaciongeneral == bajo & nivelseguridad == medio & vulnerabilidad == alto & peligrosidad == bajo)	->	salida_anomalia = poca
6	1.0	if	(saturaciongeneral == bajo & nivelseguridad == alto & vulnerabilidad == bajo & peligrosidad == bajo)	->	salida_anomalia = poca
7	1.0	if	(saturaciongeneral == bajo & nivelseguridad == alto & vulnerabilidad == medio & peligrosidad == bajo)	->	salida_anomalia = poca
8	1.0	if	(saturaciongeneral == bajo & nivelseguridad == alto & vulnerabilidad == alto & peligrosidad == bajo)	->	salida_anomalia = poca
9	1.0	if	(saturaciongeneral == medio & nivelseguridad == bajo & vulnerabilidad == bajo & peligrosidad == bajo)	->	salida_anomalia = poca
10	1.0	if	(saturaciongeneral == medio & nivelseguridad == bajo & vulnerabilidad == medio & peligrosidad == bajo)	->	salida_anomalia = poca
11	1.0	if	(saturaciongeneral == medio & nivelseguridad == bajo & vulnerabilidad == alto & peligrosidad == bajo)	->	salida_anomalia = normal
12	1.0	if	(saturaciongeneral == medio & nivelseguridad == medio & vulnerabilidad == bajo & peligrosidad == bajo)	->	salida_anomalia = poca
13	1.0	if	(saturaciongeneral == medio & nivelseguridad == medio & vulnerabilidad == medio & peligrosidad == bajo)	->	salida_anomalia = poca
14	1.0	if	(saturaciongeneral == medio & nivelseguridad == medio & vulnerabilidad == alto & peligrosidad == bajo)	->	salida_anomalia = normal
15	1.0	if	(saturaciongeneral == medio & nivelseguridad == alto & vulnerabilidad == bajo & peligrosidad == bajo)	->	salida_anomalia = poca
16	1.0	if	(saturaciongeneral == medio & nivelseguridad == alto & vulnerabilidad == medio & peligrosidad == bajo)	->	salida_anomalia = poca
17	1.0	if	(saturaciongeneral == medio & nivelseguridad == alto & vulnerabilidad == alto & peligrosidad == bajo)	->	salida_anomalia = poca
18	1.0	if	(saturaciongeneral == alto & nivelseguridad == bajo & vulnerabilidad == bajo & peligrosidad == bajo)	->	salida_anomalia = poca
19	1.0	if	(saturaciongeneral == alto & nivelseguridad == bajo & vulnerabilidad == medio & peligrosidad == bajo)	->	salida_anomalia = normal
20	1.0	if	(saturaciongeneral == alto & nivelseguridad == bajo & vulnerabilidad == alto & peligrosidad == bajo)	->	salida_anomalia = normal
21	1.0	if	(saturaciongeneral == alto & nivelseguridad == medio & vulnerabilidad == bajo & peligrosidad == bajo)	->	salida_anomalia = poca
22	1.0	if	(saturaciongeneral == alto & nivelseguridad == medio & vulnerabilidad == medio & peligrosidad == bajo)	->	salida_anomalia = poca
23	1.0	if	(saturaciongeneral == alto & nivelseguridad == medio & vulnerabilidad == alto & peligrosidad == bajo)	->	salida_anomalia = normal
24	1.0	if	(saturaciongeneral == alto & nivelseguridad == alto & vulnerabilidad == bajo & peligrosidad == bajo)	->	salida_anomalia = poca
25	1.0	if	(saturaciongeneral == alto & nivelseguridad == alto & vulnerabilidad == medio & peligrosidad == bajo)	->	salida_anomalia = poca
26	1.0	if	(saturaciongeneral == alto & nivelseguridad == alto & vulnerabilidad == alto & peligrosidad == bajo)	->	salida_anomalia = poca
27	1.0	if	(saturaciongeneral == bajo & nivelseguridad == bajo & vulnerabilidad == bajo & peligrosidad == medio)	->	salida_anomalia = poca
28	1.0	if	(saturaciongeneral == bajo & nivelseguridad == bajo & vulnerabilidad == medio & peligrosidad == medio)	->	salida_anomalia = poca
29	1.0	if	(saturaciongeneral == bajo & nivelseguridad == bajo & vulnerabilidad == alto & peligrosidad == medio)	->	salida_anomalia = normal
30	1.0	if	(saturaciongeneral == bajo & nivelseguridad == medio & vulnerabilidad == bajo & peligrosidad == medio)	->	salida_anomalia = poca
31	1.0	if	(saturaciongeneral == bajo & nivelseguridad == medio & vulnerabilidad == medio & peligrosidad == medio)	->	salida_anomalia = poca
32	1.0	if	(saturaciongeneral == bajo & nivelseguridad == medio & vulnerabilidad == alto & peligrosidad == medio)	->	salida_anomalia = normal
33	1.0	if	(saturaciongeneral == bajo & nivelseguridad == alto & vulnerabilidad == bajo & peligrosidad == medio)	->	salida_anomalia = poca
34	1.0	if	(saturaciongeneral == bajo & nivelseguridad == alto & vulnerabilidad == medio & peligrosidad == medio)	->	salida_anomalia = poca
35	1.0	if	(saturaciongeneral == bajo & nivelseguridad == alto & vulnerabilidad == alto & peligrosidad == medio)	->	salida_anomalia = normal
36	1.0	if	(saturaciongeneral == medio & nivelseguridad == bajo & vulnerabilidad == bajo & peligrosidad == medio)	->	salida_anomalia = poca
37	1.0	if	(saturaciongeneral == medio & nivelseguridad == bajo & vulnerabilidad == medio & peligrosidad == medio)	->	salida_anomalia = poca
38	1.0	if	(saturaciongeneral == medio & nivelseguridad == bajo & vulnerabilidad == alto & peligrosidad == medio)	->	salida_anomalia = alta
39	1.0	if	(saturaciongeneral == medio & nivelseguridad == medio & vulnerabilidad == bajo & peligrosidad == medio)	->	salida_anomalia = poca
40	1.0	if	(saturaciongeneral == medio & nivelseguridad == medio & vulnerabilidad == medio & peligrosidad == medio)	->	salida_anomalia = normal
41	1.0	if	(saturaciongeneral == medio & nivelseguridad == medio & vulnerabilidad == alto & peligrosidad == medio)	->	salida_anomalia = normal
42	1.0	if	(saturaciongeneral == medio & nivelseguridad == alto & vulnerabilidad == bajo & peligrosidad == medio)	->	salida_anomalia = poca
43	1.0	if	(saturaciongeneral == medio & nivelseguridad == alto & vulnerabilidad == medio & peligrosidad == medio)	->	salida_anomalia = poca
44	1.0	if	(saturaciongeneral == medio & nivelseguridad == alto & vulnerabilidad == alto & peligrosidad == medio)	->	salida_anomalia = normal
45	1.0	if	(saturaciongeneral == alto & nivelseguridad == bajo & vulnerabilidad == bajo & peligrosidad == medio)	->	salida_anomalia = normal
46	1.0	if	(saturaciongeneral == alto & nivelseguridad == bajo & vulnerabilidad == medio & peligrosidad == medio)	->	salida_anomalia = normal
47	1.0	if	(saturaciongeneral == alto & nivelseguridad == bajo & vulnerabilidad == alto & peligrosidad == medio)	->	salida_anomalia = alta
48	1.0	if	(saturaciongeneral == alto & nivelseguridad == medio & vulnerabilidad == bajo & peligrosidad == medio)	->	salida_anomalia = poca
49	1.0	if	(saturaciongeneral == alto & nivelseguridad == medio & vulnerabilidad == medio & peligrosidad == medio)	->	salida_anomalia = normal
50	1.0	if	(saturaciongeneral == alto & nivelseguridad == medio & vulnerabilidad == alto & peligrosidad == medio)	->	salida_anomalia = normal
51	1.0	if	(saturaciongeneral == alto & nivelseguridad == alto & vulnerabilidad == bajo & peligrosidad == medio)	->	salida_anomalia = poca
52	1.0	if	(saturaciongeneral == alto & nivelseguridad == alto & vulnerabilidad == medio & peligrosidad == medio)	->	salida_anomalia = poca
53	1.0	if	(saturaciongeneral == alto & nivelseguridad == alto & vulnerabilidad == alto & peligrosidad == medio)	->	salida_anomalia = normal
54	1.0	if	(saturaciongeneral == bajo & nivelseguridad == bajo & vulnerabilidad == bajo & peligrosidad == alto)	->	salida_anomalia = poca
55	1.0	if	(saturaciongeneral == bajo & nivelseguridad == bajo & vulnerabilidad == medio & peligrosidad == alto)	->	salida_anomalia = normal
56	1.0	if	(saturaciongeneral == bajo & nivelseguridad == bajo & vulnerabilidad == alto & peligrosidad == alto)	->	salida_anomalia = normal
57	1.0	if	(saturaciongeneral == bajo & nivelseguridad == medio & vulnerabilidad == bajo & peligrosidad == alto)	->	salida_anomalia = poca
58	1.0	if	(saturaciongeneral == bajo & nivelseguridad == medio & vulnerabilidad == medio & peligrosidad == alto)	->	salida_anomalia = normal
59	1.0	if	(saturaciongeneral == bajo & nivelseguridad == medio & vulnerabilidad == alto & peligrosidad == alto)	->	salida_anomalia = normal
60	1.0	if	(saturaciongeneral == bajo & nivelseguridad == alto & vulnerabilidad == bajo & peligrosidad == alto)	->	salida_anomalia = poca
61	1.0	if	(saturaciongeneral == bajo & nivelseguridad == alto & vulnerabilidad == medio & peligrosidad == alto)	->	salida_anomalia = poca
62	1.0	if	(saturaciongeneral == bajo & nivelseguridad == alto & vulnerabilidad == alto & peligrosidad == alto)	->	salida_anomalia = normal
63	1.0	if	(saturaciongeneral == medio & nivelseguridad == bajo & vulnerabilidad == bajo & peligrosidad == alto)	->	salida_anomalia = normal
64	1.0	if	(saturaciongeneral == medio & nivelseguridad == bajo & vulnerabilidad == medio & peligrosidad == alto)	->	salida_anomalia = normal
65	1.0	if	(saturaciongeneral == medio & nivelseguridad == bajo & vulnerabilidad == alto & peligrosidad == alto)	->	salida_anomalia = alta
66	1.0	if	(saturaciongeneral == medio & nivelseguridad == medio & vulnerabilidad == bajo & peligrosidad == alto)	->	salida_anomalia = poca
67	1.0	if	(saturaciongeneral == medio & nivelseguridad == medio & vulnerabilidad == medio & peligrosidad == alto)	->	salida_anomalia = normal
68	1.0	if	(saturaciongeneral == medio & nivelseguridad == medio & vulnerabilidad == alto & peligrosidad == alto)	->	salida_anomalia = normal
69	1.0	if	(saturaciongeneral == medio & nivelseguridad == alto & vulnerabilidad == bajo & peligrosidad == alto)	->	salida_anomalia = poca
70	1.0	if	(saturaciongeneral == medio & nivelseguridad == alto & vulnerabilidad == medio & peligrosidad == alto)	->	salida_anomalia = poca
71	1.0	if	(saturaciongeneral == medio & nivelseguridad == alto & vulnerabilidad == alto & peligrosidad == alto)	->	salida_anomalia = normal
72	1.0	if	(saturaciongeneral == alto & nivelseguridad == bajo & vulnerabilidad == bajo & peligrosidad == alto)	->	salida_anomalia = normal
73	1.0	if	(saturaciongeneral == alto & nivelseguridad == bajo & vulnerabilidad == medio & peligrosidad == alto)	->	salida_anomalia = alta
74	1.0	if	(saturaciongeneral == alto & nivelseguridad == bajo & vulnerabilidad == alto & peligrosidad == alto)	->	salida_anomalia = alta
75	1.0	if	(saturaciongeneral == alto & nivelseguridad == medio & vulnerabilidad == bajo & peligrosidad == alto)	->	salida_anomalia = poca
76	1.0	if	(saturaciongeneral == alto & nivelseguridad == medio & vulnerabilidad == medio & peligrosidad == alto)	->	salida_anomalia = normal
77	1.0	if	(saturaciongeneral == alto & nivelseguridad == medio & vulnerabilidad == alto & peligrosidad == alto)	->	salida_anomalia = alta
78	1.0	if	(saturaciongeneral == alto & nivelseguridad == alto & vulnerabilidad == bajo & peligrosidad == alto)	->	salida_anomalia = poca
79	1.0	if	(saturaciongeneral == alto & nivelseguridad == alto & vulnerabilidad == medio & peligrosidad == alto)	->	salida_anomalia = normal
80	1.0	if	(saturaciongeneral == alto & nivelseguridad == alto & vulnerabilidad == alto & peligrosidad == alto)	->	salida_anomalia = normal
*					

ServiciosUsados

Entrada: telnetylogin , webyftp y emailydns.
Salida: nivel_inseguridad

Rule			Premise		Conclusion
0	1.0	if	(telnetylogin == nunca_conectado & webyftp == nunca_conectado & emailydns == nunca_conectado)	->	nivel_inseguridad = bajo
1	1.0	if	(telnetylogin == nunca_conectado & webyftp == nunca_conectado & emailydns == a_intervalos)	->	nivel_inseguridad = bajo
2	1.0	if	(telnetylogin == nunca_conectado & webyftp == nunca_conectado & emailydns == siempre_conectado)	->	nivel_inseguridad = bajo
3	1.0	if	(telnetylogin == nunca_conectado & webyftp == a_intervalos & emailydns == nunca_conectado)	->	nivel_inseguridad = bajo
4	1.0	if	(telnetylogin == nunca_conectado & webyftp == a_intervalos & emailydns == a_intervalos)	->	nivel_inseguridad = bajo
5	1.0	if	(telnetylogin == nunca_conectado & webyftp == a_intervalos & emailydns == siempre_conectado)	->	nivel_inseguridad = bajo
6	1.0	if	(telnetylogin == nunca_conectado & webyftp == siempre_conectado & emailydns == nunca_conectado)	->	nivel_inseguridad = medio
7	1.0	if	(telnetylogin == nunca_conectado & webyftp == siempre_conectado & emailydns == a_intervalos)	->	nivel_inseguridad = medio
8	1.0	if	(telnetylogin == nunca_conectado & webyftp == siempre_conectado & emailydns == siempre_conectado)	->	nivel_inseguridad = medio
9	1.0	if	(telnetylogin == a_intervalos & webyftp == nunca_conectado & emailydns == nunca_conectado)	->	nivel_inseguridad = bajo
10	1.0	if	(telnetylogin == a_intervalos & webyftp == nunca_conectado & emailydns == a_intervalos)	->	nivel_inseguridad = bajo
11	1.0	if	(telnetylogin == a_intervalos & webyftp == nunca_conectado & emailydns == siempre_conectado)	->	nivel_inseguridad = medio
12	1.0	if	(telnetylogin == a_intervalos & webyftp == a_intervalos & emailydns == nunca_conectado)	->	nivel_inseguridad = medio
13	1.0	if	(telnetylogin == a_intervalos & webyftp == a_intervalos & emailydns == a_intervalos)	->	nivel_inseguridad = bajo
14	1.0	if	(telnetylogin == a_intervalos & webyftp == a_intervalos & emailydns == siempre_conectado)	->	nivel_inseguridad = bajo
15	1.0	if	(telnetylogin == a_intervalos & webyftp == siempre_conectado & emailydns == nunca_conectado)	->	nivel_inseguridad = bajo
16	1.0	if	(telnetylogin == a_intervalos & webyftp == siempre_conectado & emailydns == a_intervalos)	->	nivel_inseguridad = bajo
17	1.0	if	(telnetylogin == a_intervalos & webyftp == siempre_conectado & emailydns == siempre_conectado)	->	nivel_inseguridad = bajo
18	1.0	if	(telnetylogin == siempre_conectado & webyftp == nunca_conectado & emailydns == nunca_conectado)	->	nivel_inseguridad = bajo
19	1.0	if	(telnetylogin == siempre_conectado & webyftp == nunca_conectado & emailydns == a_intervalos)	->	nivel_inseguridad = bajo
20	1.0	if	(telnetylogin == siempre_conectado & webyftp == nunca_conectado & emailydns == siempre_conectado)	->	nivel_inseguridad = bajo
21	1.0	if	(telnetylogin == siempre_conectado & webyftp == a_intervalos & emailydns == nunca_conectado)	->	nivel_inseguridad = medio
22	1.0	if	(telnetylogin == siempre_conectado & webyftp == a_intervalos & emailydns == a_intervalos)	->	nivel_inseguridad = medio
23	1.0	if	(telnetylogin == siempre_conectado & webyftp == a_intervalos & emailydns == siempre_conectado)	->	nivel_inseguridad = medio
24	1.0	if	(telnetylogin == siempre_conectado & webyftp == siempre_conectado & emailydns == nunca_conectado)	->	nivel_inseguridad = medio
25	1.0	if	(telnetylogin == siempre_conectado & webyftp == siempre_conectado & emailydns == a_intervalos)	->	nivel_inseguridad = alto
26	1.0	if	(telnetylogin == siempre_conectado & webyftp == siempre_conectado & emailydns == siempre_conectado)	->	nivel_inseguridad = bajo
*					

Peligrosidad

Entrada:direccion y anomalia temporal.
Salida: peligrosidad

	bajo	medio	alto
muypoco	bajo	bajo	bajo
poco	bajo	bajo	medio
normal	bajo	medio	alto
mucho	medio	medio	alto
demasiado	alto	alto	alto

EstadoInterno

Entrada: numprocesos, seginterna.
Salida: estado.

	bajo	medio	alto
muypoco	normal	normal	bueno
poco	normal	normal	bueno
normal	malo	normal	bueno
mucho	malo	normal	normal
demasiado	malo	malo	malo

Vulnerabilidad

Entrada: estado y nivel_inseguridad.
Salida: vulnerabilidad.

	bajo	medio	alto
malo	medio	medio	alto
normal	bajo	medio	alto
bueno	bajo	bajo	medio

Webyftp

Entrada: web y ftp.
Salida: output_webyftp.

	nunca_conectado	a_intervalos	siempre_conectado
nunca_cone...	nunca_conectado	nunca_conectado	a_intervalos
a_intervalos	nunca_conectado	a_intervalos	a_intervalos
siempre_co...	nunca_conectado	a_intervalos	siempre_conectado

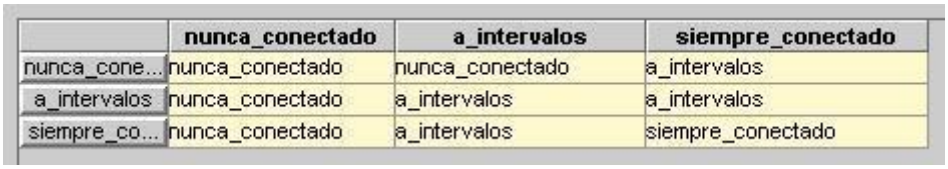
Telnetylogin

Entrada: telnet y login.
Salida: output_telnetylogin.

	nunca_conectado	a_intervalos	siempre_conectado
nunca_cone...	nunca_conectado	nunca_conectado	a_intervalos
a_intervalos	nunca_conectado	a_intervalos	a_intervalos
siempre_co...	nunca_conectado	a_intervalos	siempre_conectado

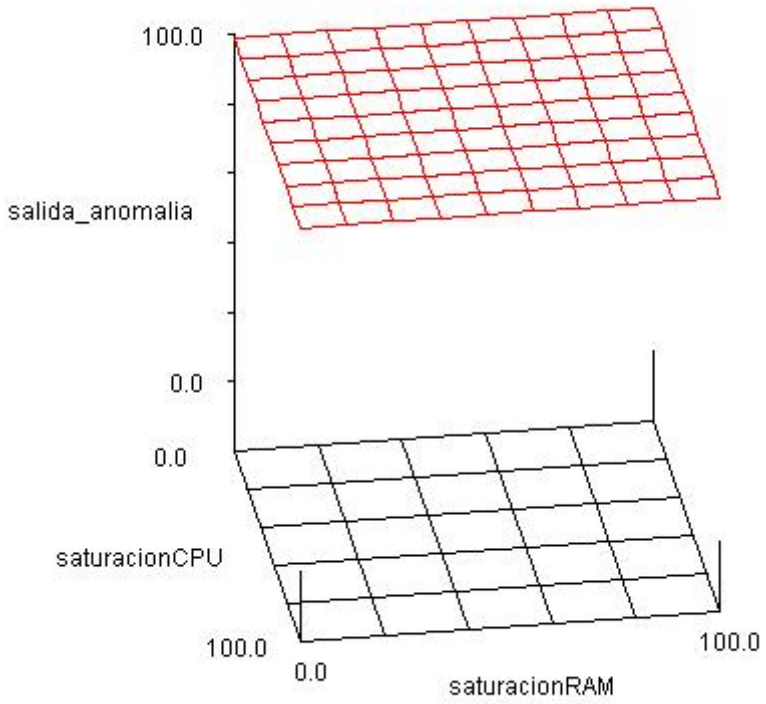
Emailydns

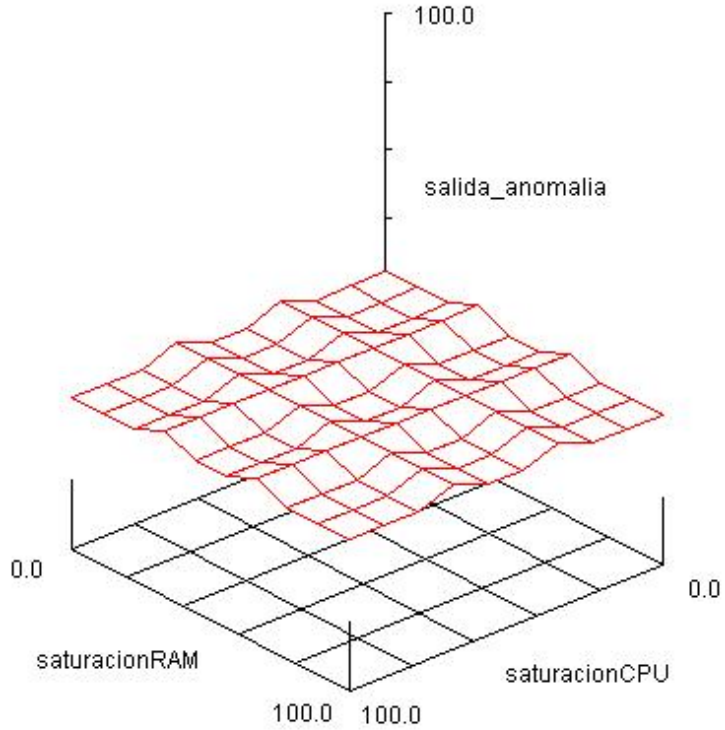
Entrada: email y dns.
Salida: output_emailydns.



	nunca_conectado	a_intervalos	siempre_conectado
nunca_cone...	nunca_conectado	nunca_conectado	a_intervalos
a_intervalos	nunca_conectado	a_intervalos	a_intervalos
siempre_co...	nunca_conectado	a_intervalos	siempre_conectado

Plot


100.0
salida_anomalia
0.0
0.0
saturacionCPU
100.0
0.0
saturacionRAM
100.0


100.0
salida_anomalia
0.0
saturacionRAM
100.0
100.0
saturacionCPU
0.0

TECNOLOGÍAS UTILIZADAS

Para la construcción del sistema experto de evaluación de la Inteligencia Emocional se ha utilizado la herramienta software Xfuzzy 3.0 desarrollada en el IMSE (Instituto de Microelectrónica de Sevilla) integrado en el CNM (Centro Nacional de Microelectrónica).

Xfuzzy es un entorno de diseño que facilita la especificación y verificación de sistemas difusos, así como su implementación final. Los módulos integrados en Xfuzzy están basados en el lenguaje XFL. La potencia y flexibilidad de este lenguaje permite el uso de Xfuzzy en un amplio rango de aplicaciones: desde la evaluación de diferentes operadores difusos, a la síntesis hardware de sistemas de inferencia basados en lógica difusa.

El proceso de desarrollo seguido es el que se muestra a continuación:

En primer lugar se han utilizado las facilidades gráficas ofrecidas por el entorno de Xfuzzy para la edición del sistema experto. Se han definido las variables de entrada pertinentes con sus tipos de conjuntos borrosos adecuados.

Posteriormente se han establecido las reglas que gobernarán el funcionamiento del sistema.

En último término, se procedió a la síntesis software a código Java.
Una vez obtenida la lógica de funcionamiento del sistema experto encapsulada en código Java, hemos incrustado este en una aplicación Java. Java es un lenguaje orientado a objetos de última generación que proporciona una amplia jerarquía de clases. En particular, para proporcionar una interfaz amigable de usuario se ha utilizado el paquete Swing.

CASOS EXTREMOS DE FUNCIONAMIENTO DEL SISTEMA

A continuación se muestran algunos casos particulares de funcionamiento del sistema detector de anomalías, ya que, no siempre la respuesta del sistema resulta tan fácil de predecir. Hay que señalar que el nivel de seguridad definido influirá notablemente (como es de esperar) en la respuesta del sistema, que ignorará o bien penalizará los valores de los parámetros de la anomalía según su relevancia para el nivel de seguridad dado. Por ello, el primer caso va acompañado de uno similar en el que no se modifican los parámetros de la anomalía, sino sólo el nivel de seguridad del sistema.

Caso 1.A.: Probabilidad alta de anomalía peligrosa
Nivel de seguridad del sistema detector de anomalías: 83%
Probabilidad de Falso Positivo: 9.13%

Nivel de seguridad:
Nivel de Fiabilidad: 99%
Nivel de Sensibilidad: 11%

Recursos del Sistema:
Saturación CPU: 80%
Saturación RAM: 98%
Actividad de Disco Duro: 60%

Peligrosidad de la conexión:
Infrecuencia del origen de la conexión: 90%
Infrecuencia del destino de la conexión 60%
Duración de la conexión: 37%
Horario de la conexión: 36%

Vulnerabilidad del sistema:
Modificaciones en el Sistema Operativo: 90%
Copias de ficheros: 39%
Numero de procesos activos: 74%
Telnet: 81%
Web: 97%
Ftp: 100%

Caso 1.B: Probabilidad relativamente alta de falso positivo debido a una gran sensibilidad

Nivel de seguridad del sistema detector de anomalías:
58%
Probabilidad de Falso Positivo:
77.6%

Nivel de seguridad:
Nivel de Fiabilidad: 70%
Nivel de Sensibilidad: 95%

Caso 2: Probabilidad baja de anomalía peligrosa debido a una sensibilidad baja

Nivel de seguridad del sistema detector de anomalías: 30%
Probabilidad de Falso Negativo: 7.02%

Nivel de seguridad:
Nivel de Fiabilidad: 93%
Nivel de Sensibilidad: 22%

Recursos del Sistema:
Saturación CPU: 14%
Saturación RAM: 22%
Actividad de Disco Duro: 30%

Peligrosidad de la conexión:
Infrecuencia del origen de la conexión: 9%
Infrecuencia del destino de la conexión 26%
Duración de la conexión: 21%
Horario de la conexión: 54%

Vulnerabilidad del sistema:
Telnet: 13%
Web: 0%
Ftp: 0%
Login: 0%

Caso 3: Probabilidad muy baja de anomalía peligrosa

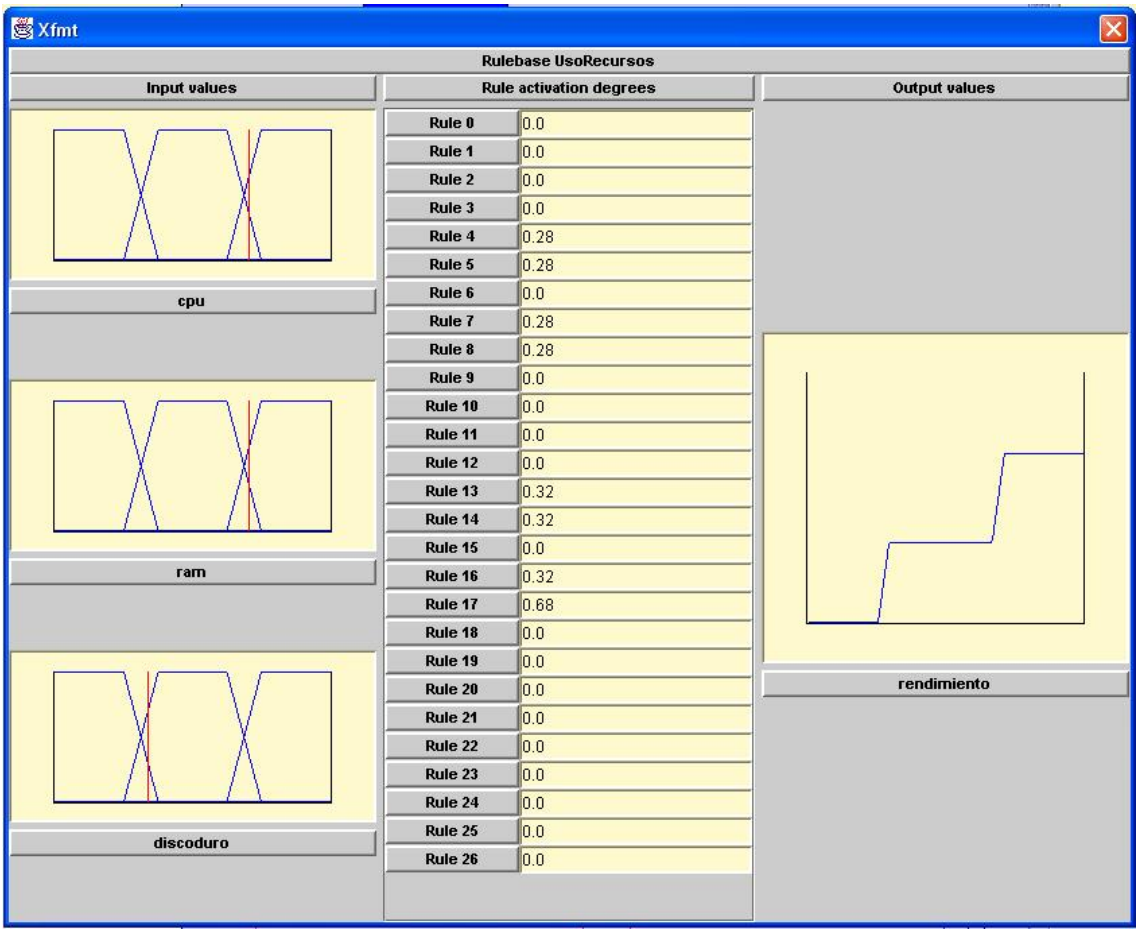
Nivel de seguridad del sistema detector de anomalías: 15%
Probabilidad de Falso Negativo: 14.7%

Nivel de seguridad:
Nivel de Fiabilidad: 50%
Nivel de Sensibilidad: 50%

Parámetros "normales" (todos al 50%)

FUNCIONAMIENTO DEL XFUZZY

El Xfuzzy tambien permite la monitorización en tiempo real del valor de cada salida de cada regla cuando se van modificando las entradas, con esto se puede estudiar el comportamiento del sistema ante cada caso. Permite afinar el sistema para una mejor adaptación al problema estudiado.



CONCLUSIONES

Una vez desarrollado el sistema experto, nos hemos dado cuenta de la complejidad que entraña la “valoración” de las probabilidades que tiene una anomalía de ser realmente peligrosa. Somos conscientes de que para un más eficiente funcionamiento de la aplicación sería necesario la aplicación de técnicas heurísticas, basadas en la experiencia, que constituyesen un factor a tener en cuenta ante futuros ataques... ¡¡¡con tanta importancia como el nivel de seguridad establecido!!!. Tal vez, en futuras versiones podría tenerse en cuenta no sólo la peligrosidad de una anomalía (en base a sus parámetros), sino también el “historial” del sistema que nuestro sistema detector trata de proteger del ataque de intrusos.

Por otro lado, no ha sido fácil “pulir” los números borrosos, sobre todo cuando hemos de determinar el nivel de seguridad del sistema (fiabilidad y sensibilidad): el futuro en este tema debe abordarse teniendo en cuenta que:

1. Ante unos mismos parámetros de anomalía, el sistema debe poder responder claramente en función de la seguridad dada de modo que:
    - al aumentar la sensibilidad, aumentará también la probabilidad de que se origine un falso positivo
    - al disminuir la sensibilidad, disminuye el riesgo de que la valoración de la peligrosidad de la anomalía sea errónea
2. Nos ha resultado muy complicado aplicar borrosidad en referencia al uso de servicios como ftp, web, telnet... dado que “sin ser un experto” en ese tema AUN MAS ESPECÍFICO, la única posibilidad de obtener información es determinando el grado de uso de dichos servicios... pero no obtenemos información acerca de la “inteligencia” en el uso de los mismos... algo que deberían tener en cuenta las personas que aborden este tema.

BIBLIOGRAFÍA

www.ossim.net -

Open Source Security Information Management (página oficial del proyecto OSSIM)

10th SIGEF Congress – First International Workshop – Intelligent systems for intrusion detection and facing malware

CODIGO FUENTE DE LA APLICACIÓN

<interfaz.java>

```
import java.awt.*;
import java.awt.Color.*;
import javax.swing.text.*;
import java.lang.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.border.*;
import javax.swing.event.*;
import java.util.*;

class Tparametros_anomalia
{
	//1.PANEL "RECURSOS DEL SISTEMA"
		//1.1.SUBPANEL "PARAMETROS DEL PC"
			int saturacion_CPU;
			int saturacion_RAM;
			int actividad_DD;
		//1.2.SUBPANEL "PARAMETROS DE RED"
			int ancho_banda_consumido;
			int numero_conexiones;
	//2.PANEL "PELIGROSIDAD DE LA CONEXION"
		//2.1.SUBPANEL "PROCEDENCIA Y FIN"
			int origen_anormal;
			int destino_anormal;
		//2.2.SUBPANEL "PARAMETROS TEMPORALES"
			int duracion_conexion;
			int horario_conexion;
	//3.PANEL "VULNERABILIDAD DEL SISTEMA"
		int numero_procesos_activos;
		//3.1.SUBPANEL "SEGURIDAD INTERNA"
			int modificaciones_SO;
			int copia_de_ficheros;
		//3.2.SUBPANEL "SERVICIOS USADOS"
			int telnet; //0-no 1-si
			int web;
			int ftp;
			int email;
			int dns;
			int login;
	//*.PANEL "NIVEL DE SEGURIDAD DEL SISTEMA DETECTOR DE ANOMALIAS"
		int nivel_fiabilidad;
		int nivel_sensibilidad;
};


class Tinterfaz_xfuzzy extends JFrame
{
	Tparametros_anomalia parametros_anomalia = new Tparametros_anomalia();

	//************************************************************
	// DEFINICION DE LOS ELEMENTOS DE MENU DEL MENU PRINCIPAL *
	//************************************************************

	final String MENU_RESETEAR_DATOS_OPCION1 = new String("Todos los datos");
	final String MENU_RESETEAR_DATOS_OPCION2 = new String("Datos de <Recursos del Sistema>");
	final String MENU_RESETEAR_DATOS_OPCION3 = new String("Datos de <Peligrosidad de la Conexion>");
	final String MENU_RESETEAR_DATOS_OPCION4 = new String("Datos de <Vulnerabilidad del Sistema>");
	final String MENU_RESETEAR_DATOS_OPCION5 = new String("Nivel de Seguridad");

	JMenuBar menu_principal = new JMenuBar();
	JMenu acciones = new JMenu("Acciones");
		JMenu acc_resetear_datos = new JMenu("Resetear datos");
			JMenu acc_resetear_datos_anomalia = new JMenu("Anomalia");
				JMenuItem acc_resetear_datos_anomalia_TODOS = new
JMenuItem(MENU_RESETEAR_DATOS_OPCION1);
				JMenuItem acc_resetear_datos_anomalia_RECURSOS_SISTEMA = new
JMenuItem(MENU_RESETEAR_DATOS_OPCION2);
```

```
                                        JMenuItem acc_resetear_datos_anomalia_PELIGROSIDAD_CONEXION = new
JMenuItem(MENU_RESETEAR_DATOS_OPCION3);
                                        JMenuItem acc_resetear_datos_anomalia_VULNERABILIDAD_SISTEMA = new
JMenuItem(MENU_RESETEAR_DATOS_OPCION4);
                        JMenuItem acc_resetear_datos_nivel_seguridad = new JMenuItem("Nivel de Seguridad");
                JMenuItem acc_evaluar_anomalia = new JMenuItem("Evaluar anomalia");
        JMenu ayuda = new JMenu("Ayuda");
                JMenuItem ay_acerca_autores = new JMenuItem("Acerca de los autores");

        //*********************************************************************
        // OBJETOS QUE DETERMINAN EL VALOR DE LOS PARAMETROS DE LA ANOMALIA *
        //*********************************************************************

        JSlider slider_INVISIBLE1 = crear_JSlider_INVISIBLE();
        JSlider slider_INVISIBLE2 = crear_JSlider_INVISIBLE();
        JSlider slider_INVISIBLE3 = crear_JSlider_INVISIBLE();
        JSlider slider_INVISIBLE4 = crear_JSlider_INVISIBLE();
        JSlider slider_INVISIBLE5 = crear_JSlider_INVISIBLE();
        JSlider slider_INVISIBLE6 = crear_JSlider_INVISIBLE();
        JSlider slider_INVISIBLE7 = crear_JSlider_INVISIBLE();
        JSlider slider_INVISIBLE8 = crear_JSlider_INVISIBLE();
        //1.PANEL "RECURSOS DEL SISTEMA"
                //1.1.SUBPANEL "PARAMETROS DEL PC"
                        JSlider slider_saturacion_CPU = crear_JSlider_0_a_100("Saturacion CPU");
                        JSlider slider_saturacion_RAM = crear_JSlider_0_a_100("Saturacion RAM");
                        JSlider slider_actividad_DD = crear_JSlider_0_a_100(" Actividad del Disco Duro ");
                //1.2.SUBPANEL "PARAMETROS DE RED"
                        JSlider slider_ancho_banda_consumido = crear_JSlider_5_estados_A(" Ancho de Banda consumido ");
                        JSlider slider_numero_conexiones = crear_JSlider_5_estados_B(" Numero de conexiones ");
        //2.PANEL "PELIGROSIDAD DE LA CONEXION"
                //2.1.SUBPANEL "PROCEDENCIA Y FIN"
                        JSlider slider_origen_anormal = crear_JSlider_5_estados_E(" Infrecuencia del origen de la conexión ");
                        JSlider slider_destino_anormal = crear_JSlider_5_estados_E(" Infrecuencia del destino de la conexión ");
                //2.2.SUBPANEL "PARAMETROS TEMPORALES"
                        JSlider slider_duracion_conexion = crear_JSlider_3_estados(" Duracion de la conexion ");
                        JSlider slider_horario_conexion = crear_JSlider_5_estados_C(" Horario de la conexion ");
        //3.PANEL "VULNERABILIDAD DEL SISTEMA"
                JSlider slider_numero_procesos_activos = crear_JSlider_5_estados_D(" Numero de procesos activos ");
                //3.1.SUBPANEL "SEGURIDAD INTERNA"
                        JSlider slider_modificaciones_SO = crear_JSlider_5_estados_B(" Modificaciones en el Sistema Operativo ");
                        JSlider slider_copia_ficheros = crear_JSlider_5_estados_B(" Copias de ficheros ");
                //3.2.SUBPANEL "SERVICIOS USADOS"
                        JSlider slider_telnet  = crear_JSlider_3_estados_B(" Telnet ");
                        JSlider slider_web = crear_JSlider_3_estados_B(" Web ");
                        JSlider slider_ftp = crear_JSlider_3_estados_B(" Ftp ");
                        JSlider slider_email = crear_JSlider_3_estados_B(" Email ");
                        JSlider slider_dns = crear_JSlider_3_estados_B(" Dns ");
                        JSlider slider_login = crear_JSlider_3_estados_B(" Login ");
        //*.PANEL "NIVEL DE SEGURIDAD DEL SISTEMA DETECTOR DE ANOMALIAS"
                JSlider slider_nivel_fiabilidad = crear_JSlider_5_estados_D(" Nivel de fiabilidad ");
                JSlider slider_nivel_sensibilidad = crear_JSlider_5_estados_D(" Nivel de sensibilidad ");

        //*********************************************************************
        // GESTION DE EVENTOS ASOCIADA AL "MENU PRINCIPAL" DE LA APLICACION *
        //*********************************************************************

        class EventoMenu_resetear_datos implements ActionListener
        {
                private void resetear_datos_RECURSOS_SISTEMA() {
                        slider_saturacion_CPU.setValue(50);
                        slider_saturacion_RAM.setValue(50);
                        slider_actividad_DD.setValue(50);
                        slider_ancho_banda_consumido.setValue(50);
                        slider_numero_conexiones.setValue(50);
                }
                private void resetear_datos_PELIGROSIDAD_CONEXION() {
                        slider_origen_anormal.setValue(50);
                        slider_destino_anormal.setValue(50);
                        slider_duracion_conexion.setValue(50);
                        slider_horario_conexion.setValue(50);
                }
                private void resetear_datos_VULNERABILIDAD_SISTEMA() {
                        slider_modificaciones_SO.setValue(50);
                        slider_copia_ficheros.setValue(50);
                        slider_numero_procesos_activos.setValue(50);
                        slider_telnet.setValue(50);
```

```
                slider_web.setValue(50);
                slider_ftp.setValue(50);
                slider_email.setValue(50);
                slider_dns.setValue(50);
                slider_login.setValue(50);
        }
        private void resetear_datos_TODOS() {
                resetear_datos_RECURSOS_SISTEMA();
                resetear_datos_PELIGROSIDAD_CONEXION();
                resetear_datos_VULNERABILIDAD_SISTEMA();
        }
        private void resetear_datos_NIVEL_SEGURIDAD() {
                slider_nivel_fiabilidad.setValue(50);
                slider_nivel_sensibilidad.setValue(50);
        }
        public void actionPerformed(ActionEvent e)
        {
                /*      MENU_RESETEAR_DATOS_OPCION1 = "Todos los datos"
                        MENU_RESETEAR_DATOS_OPCION2 = "Datos de <Recursos del Sistema>"
                        MENU_RESETEAR_DATOS_OPCION3 = "Datos de <Peligrosidad de la Conexion>"
                        MENU_RESETEAR_DATOS_OPCION4 = "Datos de <Vulnerabilidad del Sistema>"
                        MENU_RESETEAR_DATOS_OPCION5 = "Nivel de Seguridad" */
                String menu_item = new String(e.getActionCommand());
                if (menu_item.compareTo(MENU_RESETEAR_DATOS_OPCION1)==0)
                        resetear_datos_TODOS();
                else
                        if (menu_item.compareTo(MENU_RESETEAR_DATOS_OPCION2)==0)
                                resetear_datos_RECURSOS_SISTEMA();
                        else
                                if (menu_item.compareTo(MENU_RESETEAR_DATOS_OPCION3)==0)
                                        resetear_datos_PELIGROSIDAD_CONEXION();
                                else
                                        if (menu_item.compareTo(MENU_RESETEAR_DATOS_OPCION4)==0)
                                                resetear_datos_VULNERABILIDAD_SISTEMA();
                                        else
                                                if
(menu_item.compareTo(MENU_RESETEAR_DATOS_OPCION5)==0)
                                                        resetear_datos_NIVEL_SEGURIDAD();
        }
};

public class EventoMenu_evaluar_anomalia extends JDialog implements ActionListener {
        int i = 0;
        public class TDetectorAnomalias
        {
                double [] entradas = new double [20];
           double [] salida = new double[1];
                fozzimA sistema_experto =new fozzimA();
                public TDetectorAnomalias()
                {
                        entradas[0] = slider_saturacion_CPU.getValue();
                        entradas[1] = slider_saturacion_RAM.getValue();
                        entradas[2] = slider_actividad_DD.getValue();
                        entradas[3] = slider_numero_conexiones.getValue();
                        entradas[4] = slider_ancho_banda_consumido.getValue();
                        entradas[5] = slider_duracion_conexion.getValue();
                        entradas[6] = slider_horario_conexion.getValue();
                        entradas[7] = slider_origen_anormal.getValue();
                        entradas[8] = slider_destino_anormal.getValue();
                        entradas[9] = slider_nivel_fiabilidad.getValue();
                        entradas[10] = slider_nivel_sensibilidad.getValue();
                        entradas[11] = slider_modificaciones_SO.getValue();
                        entradas[12] = slider_copia_ficheros.getValue();
                        entradas[13] = slider_numero_procesos_activos.getValue();
                        entradas[14] = slider_telnet.getValue()/100;
                        entradas[15] = slider_web.getValue()/100;
                        entradas[16] = slider_ftp.getValue()/100;
                        entradas[17] = slider_email.getValue()/100;
                        entradas[18] = slider_dns.getValue()/100;
                        entradas[19] = slider_login.getValue()/100;
                        salida = sistema_experto.crispInference(entradas);
                }

                public int get_probabilidad_anomalia() {
                        return (int) salida[0];
```

```
		}

	}

	String OPCION1 = new String("PROBABILIDAD DE FALSO POSITIVO: \n");
	String OPCION2 = new String("PROBABILIDAD DE FALSO NEGATIVO: \n");
	String OPCION;
	//constructor de la clase "EventoMenu_evaluar_anomalia"
	public EventoMenu_evaluar_anomalia(JFrame owner) {
		super (owner,"RESULTADOS DE LA EVALUACION DE LA ANOMALIA...",true);

	}

	public void centerParent () {
		int x;
		int y;
		// Find out our parent
		Container myParent = getParent();
		Point topLeft = myParent.getLocationOnScreen();
		Dimension parentSize = myParent.getSize();
		Dimension mySize = getSize();
		if (parentSize.width > mySize.width)
			x = ((parentSize.width - mySize.width)/2) + topLeft.x;
		else
			x = topLeft.x;
		if (parentSize.height > mySize.height)
		   y = ((parentSize.height - mySize.height)/2) + topLeft.y;
		else
		   y = topLeft.y;
		setLocation (x, y);
		requestFocus();
	}

	public void actionPerformed(ActionEvent e) {
		Container c = this.getContentPane();
		c.setLayout(new BorderLayout());
		JTextArea texto = new JTextArea();
		texto.setEditable(false);
		c.add("Center",texto);
		setSize(400,100);
		setModal(true);
		setResizable(false);
		texto.setFont(new Font("Courier",Font.ITALIC+Font.BOLD,12));
		texto.setBackground(Color.orange);
		texto.setForeground(Color.black);
		//**************************************
		// CODIGO DE EVALUACION DE LA ANOMALIA *
		//**************************************
		TDetectorAnomalias SE = new TDetectorAnomalias();
		int probabilidad_anomalia = SE.get_probabilidad_anomalia();
		if (probabilidad_anomalia >= 50) OPCION = OPCION1;
		else OPCION = OPCION2;
		double fiabilidad = SE.entradas[9];
		double sensibilidad = SE.entradas[10];
		double probabilidad_falso = (((sensibilidad * probabilidad_anomalia) / 100)/ (fiabilidad+1))*100;
		if (probabilidad_falso >= 100)
			probabilidad_falso = 100;
		String str_prob_falso = new String((new Double(probabilidad_falso)).toString());
		if (str_prob_falso.length()>3)
			str_prob_falso = str_prob_falso.substring(0,4);
		texto.append("NIVEL DE SEGURIDAD DEL SISTEMA DETECTOR DE ANOMALIAS: \n" +
					probabilidad_anomalia + "% \n" +
					OPCION +
					str_prob_falso + "%" +
					"\n\n (Hacer doble-click en la esquina superior derecha \n para finalizar)");

		centerParent();
		show();
		c.remove(texto);

	}

};

public class EventoMenu_Acerca_de_autores extends JDialog implements ActionListener {
	JTextArea texto = new JTextArea();
```

```
		public EventoMenu_Acerca_de_autores(JFrame owner) {
			super (owner,"Acerca de los autores...",true);
			String nuevalinea = new String("\n");
			texto.append(" - DAVID SÁENZ TAGARRO" + nuevalinea +
								"   correo: davidst@usuarios.retecal.es" + nuevalinea +
								" - IAGO TRANCÓN BARROS" + nuevalinea +
								"   correo: el_sokette@hotmail.com");
			texto.setEditable(false);
		}
		public void actionPerformed(ActionEvent e) {
			Container c = this.getContentPane();
			c.setLayout(new BorderLayout());
			c.add("Center",texto);
			setSize(285,100);
			setModal(true);
			setResizable(false);
			texto.setFont(new Font("Courier",Font.ITALIC+Font.BOLD,12));
			texto.setBackground(Color.orange);
			texto.setForeground(Color.black);
			show();
		}
	};

	//*********************************************************************
	//		VARIABLE QUE SIRVE DE "ADMINISTRADOR DE DISEÑO DE LA APLICACION" *
	//*********************************************************************

	JPanel admon_disenio;

	//*****************************************************************************************************************************
	// GESTION DE EVENTOS PARA MOSTRAR EN UN AREA DE TEXTO LA DEFINICION DEL PARAMETRO SOBRE EL CUAL SE ENCUENTRA EL RATON *
	//*****************************************************************************************************************************

	class Evento_informacion_datos_slider implements MouseListener
	{
		public void mouseEntered(MouseEvent me) {
			JSlider slider = (JSlider) me.getSource();
			if (slider == slider_saturacion_CPU) {
				definiciones_datos_sistema_experto.setText("Nivel de utilizacion en operaciones del CPU");

			}
			else
			if (slider == slider_saturacion_RAM)
				definiciones_datos_sistema_experto.setText("Memoria que se esta utilizando en el pc");

			else
			if (slider == slider_actividad_DD)
				definiciones_datos_sistema_experto.setText("Nivel de transacciones de lectura y escritura al Disco Duro");
			else
			if (slider == slider_ancho_banda_consumido)
				definiciones_datos_sistema_experto.setText("Franja consumida de la capacidad de transmision de la linea de conexiona internet");
			else
			if (slider == slider_numero_conexiones)
				definiciones_datos_sistema_experto.setText("Numero de peticiones y envios desde el pc y hacia el pc respectivamente");
			else
			if (slider == slider_origen_anormal)
				definiciones_datos_sistema_experto.setText("Clasificacion del origen de una conexion establecida segun si es un usuario conocido y con permiso o no");
			else
			if (slider == slider_destino_anormal)
				definiciones_datos_sistema_experto.setText("Destino de la conexion depedendiendo de si es un destino comun o extraño");
			else
			if (slider == slider_duracion_conexion)
				definiciones_datos_sistema_experto.setText("Tiempo que permanece abierta la conexion");
			else
			if (slider == slider_horario_conexion)
				definiciones_datos_sistema_experto.setText("Clasificacion dependiendo del horario. La peligrosidad de un ataque nocturno siempre será mayor que el de uno diurno");
			else
			if (slider == slider_numero_procesos_activos)
```

```
                                        definiciones_datos_sistema_experto.setText("Numero de procesos que se ejecutan
concurrentemente en el ordenador");
                        else
                        if (slider == slider_modificaciones_SO)
                                        definiciones_datos_sistema_experto.setText("Alteraciones en archivos fundamentales del sistema,
asi como en archivos de recopilacion de actividades de usuarios");
                        else
                        if (slider == slider_copia_ficheros)
                                        definiciones_datos_sistema_experto.setText("Copia o acceso a ficheros de seguridad del sistema ,
como pueden ser los archivos de claves o bases de datos");
                        else
                        if (slider == slider_nivel_fiabilidad)
                                        definiciones_datos_sistema_experto.setText("El grado de certeza que nos ofrece nuestro detector
ante el aviso de un posible evento");
                        else
                        if (slider == slider_nivel_sensibilidad)
                                        definiciones_datos_sistema_experto.setText("La capacidad de análisis, en profundidad y
complejidad, que posee nuestro detector a la hora de localizar un posible ataque");
                        else
                        if (slider == slider_telnet)
                                        definiciones_datos_sistema_experto.setText("Servicio de consola remota");
                        else
                        if (slider == slider_web)
                                        definiciones_datos_sistema_experto.setText("Este servicio esta activo si el ordenador analizado es
un servidor web");
                        else
                        if (slider == slider_ftp)
                                        definiciones_datos_sistema_experto.setText("Servicio activo si existe un servidor de ftp en el
ordenador");
                        else
                        if (slider == slider_email)
                                        definiciones_datos_sistema_experto.setText("Servicio activo si en el ordenador existe un servidor
de correo");
                        else
                        if (slider == slider_dns)
                                        definiciones_datos_sistema_experto.setText("Servicio activo si el ordenador actua como servidor
de direcciones en internet o servidor de DNS");
                        else
                        if (slider == slider_login)
                                        definiciones_datos_sistema_experto.setText("Posibilidad de acceso a los recursos del ordenador
mediante cuenta con contraseñas");

                }
                //Necesitamos re-declarar los siguientes metodos ya que en "MouseListener" estos metodos son abstractos
                public void mouseClicked(MouseEvent event){}
                public void mousePressed(MouseEvent event){}
                public void mouseExited(MouseEvent event){
                        definiciones_datos_sistema_experto.setText("OBSERVE AQUI LA DEFINICION DE LA VARIABLE
MODIFICADA");
                }
                public void mouseReleased(MouseEvent event){}
        }

        //*************************************************************************************************
        // PANELES - SOBRES LOS QUE SE DISPONEN LOS JSLIDER QUE PERMITEN DAR VALOR A LAS VARIABLES *
        // BORDES - QUE DELIMITAN LOS SLIDERS INDICANDO LA VARIABLE A LA QUE HACEN REFERENCIA              *
        //*************************************************************************************************

        JTabbedPane panel_datos = new JTabbedPane();
                JPanel panel_RECURSOS_SISTEMA = new JPanel(new BorderLayout());
                        JPanel subpanel_PARAMETROS_PC = new JPanel(new GridLayout(2,2));
                        JPanel subpanel_PARAMETROS_RED = new JPanel(new GridLayout(1,2));
                        JPanel subpanel_recursos_sistema_INVISIBLE = new JPanel(new GridLayout(3,1));
                JPanel panel_PELIGROSIDAD_CONEXION = new JPanel(new BorderLayout());
                        JPanel subpanel_PARAMETROS_PROCEDENCIA_Y_FIN = new JPanel(new GridLayout(1,2));
                        JPanel subpanel_PARAMETROS_TEMPORALES = new JPanel(new GridLayout(1,2));
                        JPanel subpanel_peligrosidad_conexion_INVISIBLE = new JPanel(new GridLayout(4,1));
                JPanel panel_VULNERABILIDAD_SISTEMA = new JPanel(new BorderLayout());
                        JPanel subpanel_SEGURIDAD_INTERNA = new JPanel(new GridLayout(1,2));
                        JPanel subpanel_NUMERO_PROCESOS_ACTIVOS = new JPanel(new GridLayout(1,2));
                        JPanel subpanel_SERVICIOS_USADOS = new JPanel(new GridLayout(3,2));

        Border raisedbevel = BorderFactory.createRaisedBevelBorder();
        Border loweredbevel = BorderFactory.createLoweredBevelBorder();
        Border borde_subpaneles = BorderFactory.createCompoundBorder(raisedbevel, loweredbevel);
```

```
JPanel panel_herramientas = new JPanel(new BorderLayout());
		JProgressBar barra_progreso = new JProgressBar(0,100);
		JTextArea definiciones_datos_sistema_experto = new JTextArea();
JPanel panel_NIVEL_SEGURIDAD_SISTEMA_DETECTOR_ANOMALIAS = new JPanel(new GridLayout(1,2));

//*************************************************************
// CONSTRUCTORES - DE LOS DISTINTOS TIPOS DE OBJETOS JSLIDER *
//*************************************************************

private JSlider crear_JSlider(String nombre_variable) {
		JSlider slider = new JSlider();
		slider.setBorder(BorderFactory.createTitledBorder(nombre_variable));
   slider.setPaintTicks(true); //muestra las divisiones sobre la barrita
		slider.setPaintLabels(true); //muestra el valor de las divisiones grandes
		slider.setMinorTickSpacing(5); //cada 5 unidades una division pequeña
		slider.addMouseListener(new Evento_informacion_datos_slider());
		return slider;
}

private JSlider crear_JSlider_0_a_100(String nombre_variable) {
		JSlider slider = crear_JSlider(nombre_variable);
   slider.setMajorTickSpacing(20); //cada 20 unidades una division grande
		return slider;
}

private JSlider crear_JSlider_INVISIBLE() {
		JSlider slider = crear_JSlider_0_a_100("");
		slider.setVisible(false);
		return slider;
}

private JSlider crear_JSlider_5_estados_A(String nombre_variable) {
		JSlider slider = crear_JSlider(nombre_variable);
   slider.setMajorTickSpacing(25); //cada 25 unidades una division grande
		Hashtable tabla = slider.createStandardLabels(25);
		tabla.put(new Integer(0),new JLabel("minimo"));
		tabla.put(new Integer(25),new JLabel("poco"));
		tabla.put(new Integer(50),new JLabel("normal"));
		tabla.put(new Integer(75),new JLabel("mucho"));
		tabla.put(new Integer(100),new JLabel("maximo"));
		slider.setLabelTable(tabla);
		return slider;
}

private JSlider crear_JSlider_5_estados_B(String nombre_variable) {
		JSlider slider = crear_JSlider(nombre_variable);
   slider.setMajorTickSpacing(25); //cada 25 unidades una division grande
		Hashtable tabla = slider.createStandardLabels(25);
		tabla.put(new Integer(0),new JLabel("minimas"));
		tabla.put(new Integer(25),new JLabel("pocas"));
		tabla.put(new Integer(50),new JLabel("normales"));
		tabla.put(new Integer(75),new JLabel("muchas"));
		tabla.put(new Integer(100),new JLabel("maximas"));
		slider.setLabelTable(tabla);
		return slider;
}

private JSlider crear_JSlider_5_estados_C(String nombre_variable) {
		JSlider slider = crear_JSlider(nombre_variable);
   slider.setMajorTickSpacing(25); //cada 25 unidades una division grande
		Hashtable tabla = slider.createStandardLabels(25);
		tabla.put(new Integer(0),new JLabel("0-5"));
		tabla.put(new Integer(25),new JLabel("6-11"));
		tabla.put(new Integer(50),new JLabel("11-16"));
		tabla.put(new Integer(75),new JLabel("16-21"));
		tabla.put(new Integer(100),new JLabel("21-24"));
		slider.setLabelTable(tabla);
		return slider;
}

private JSlider crear_JSlider_5_estados_D(String nombre_variable) {
		JSlider slider = crear_JSlider(nombre_variable);
   slider.setMajorTickSpacing(25); //cada 25 unidades una division grande
		Hashtable tabla = slider.createStandardLabels(25);
		tabla.put(new Integer(0),new JLabel("minimo"));
		tabla.put(new Integer(25),new JLabel("pocos"));
```

```
		tabla.put(new Integer(50),new JLabel("normal"));
		tabla.put(new Integer(75),new JLabel("muchos"));
		tabla.put(new Integer(100),new JLabel("maximo"));
		slider.setLabelTable(tabla);
		return slider;
	}

	private JSlider crear_JSlider_5_estados_E(String nombre_variable) {
		JSlider slider = crear_JSlider(nombre_variable);
	   slider.setMajorTickSpacing(25); //cada 25 unidades una division grande
		Hashtable tabla = slider.createStandardLabels(25);
		tabla.put(new Integer(0),new JLabel("muy baja"));
		tabla.put(new Integer(25),new JLabel("baja"));
		tabla.put(new Integer(50),new JLabel("normal"));
		tabla.put(new Integer(75),new JLabel("alta"));
		tabla.put(new Integer(100),new JLabel("muy alta"));
		slider.setLabelTable(tabla);
		return slider;
	}

	private JSlider crear_JSlider_3_estados(String nombre_variable) {
		JSlider slider = crear_JSlider(nombre_variable);
	   slider.setMajorTickSpacing(25); //cada 25 unidades una division grande
		Hashtable tabla = slider.createStandardLabels(50);
		tabla.put(new Integer(0),new JLabel("baja"));
		tabla.put(new Integer(50),new JLabel("media"));
		tabla.put(new Integer(100),new JLabel("alta"));
		slider.setLabelTable(tabla);
		return slider;
	}

	private JSlider crear_JSlider_3_estados_B (String nombre_variable) {
		JSlider slider = crear_JSlider(nombre_variable);
	   slider.setMajorTickSpacing(25); //cada 25 unidades una division grande
		Hashtable tabla = slider.createStandardLabels(50);
		tabla.put(new Integer(0),new JLabel("Nunca"));
		tabla.put(new Integer(50),new JLabel("A veces"));
		tabla.put(new Integer(100),new JLabel("Permanente"));
		slider.setLabelTable(tabla);
		return slider;
	}

	class Slider_Listener implements ChangeListener {
	   public void stateChanged(ChangeEvent e) {
			JSlider slider_modificado = (JSlider)e.getSource(); //identificamos el slider para el cual se ha modificado su valor
				//Si el boton del "slider" esta siendo arrastrado, entonces obtenemos su valor
		   if (!slider_modificado.getValueIsAdjusting()) {
					barra_progreso.setValue(0);
					barra_progreso.setString("OBSERVE AQUI EL VALOR DE LA VARIABLE MODIFICADA");
				}
				else {
					int valor = (int)slider_modificado.getValue();
					barra_progreso.setValue(valor);
					barra_progreso.setString((new Integer(valor)).toString()+"%");
					}
		}
	}

	//*********************************************
	// PROCEDIMIENTO - PARA LA CREACION DEL MENU *
	//*********************************************

	private void creacion_menu() {
		setJMenuBar(menu_principal);
		menu_principal.setVisible(true);
		menu_principal.add(acciones);
		menu_principal.add(Box.createHorizontalGlue()); //el ultimo elemento de la barra de menus lo desplazamos a la
derecha del todo
		menu_principal.add(ayuda);
		acciones.add(acc_resetear_datos);
			acc_resetear_datos.add(acc_resetear_datos_anomalia);
				acc_resetear_datos_anomalia.add(acc_resetear_datos_anomalia_TODOS);
				acc_resetear_datos_anomalia.add(acc_resetear_datos_anomalia_RECURSOS_SISTEMA);
				acc_resetear_datos_anomalia.add(acc_resetear_datos_anomalia_PELIGROSIDAD_CONEXION);
				acc_resetear_datos_anomalia.add(acc_resetear_datos_anomalia_VULNERABILIDAD_SISTEMA);
			acc_resetear_datos.add(acc_resetear_datos_nivel_seguridad);
```

```
		acciones.add(acc_evaluar_anomalia);
		ayuda.add(ay_acerca_autores);
		//Asociamos las opciones de menu con su gestor de eventos
		acc_resetear_datos_anomalia_TODOS.addActionListener(new EventoMenu_resetear_datos());
		acc_resetear_datos_anomalia_RECURSOS_SISTEMA.addActionListener(new EventoMenu_resetear_datos());
		acc_resetear_datos_anomalia_PELIGROSIDAD_CONEXION.addActionListener(new EventoMenu_resetear_datos());
		acc_resetear_datos_anomalia_VULNERABILIDAD_SISTEMA.addActionListener(new EventoMenu_resetear_datos());
		acc_resetear_datos_nivel_seguridad.addActionListener(new EventoMenu_resetear_datos());
		acc_evaluar_anomalia.addActionListener(new EventoMenu_evaluar_anomalia(this));
		ay_acerca_autores.addActionListener(new EventoMenu_Acerca_de_autores(this));
	}

	//*************************************************************************************************
	// CREACION DE LOS PANELES QUE PERMITEN INTRODUCIR LOS VALORES DE LOS PARAMETROS DE LA ANOMALIA *
	//*************************************************************************************************

	private void creacion_panel_datos_RECURSOS_SISTEMA() {
		subpanel_PARAMETROS_PC.setBorder(BorderFactory.createTitledBorder(borde_subpaneles," Parametros de HW del
PC ",TitledBorder.CENTER,TitledBorder.TOP));
		subpanel_PARAMETROS_RED.setBorder(BorderFactory.createTitledBorder(borde_subpaneles," Parametros de Red
",TitledBorder.CENTER,TitledBorder.TOP));
		panel_RECURSOS_SISTEMA.add("North",subpanel_PARAMETROS_PC);
			subpanel_PARAMETROS_PC.add(slider_saturacion_CPU);
			subpanel_PARAMETROS_PC.add(slider_saturacion_RAM);
			subpanel_PARAMETROS_PC.add(slider_actividad_DD);
		panel_RECURSOS_SISTEMA.add("Center",subpanel_PARAMETROS_RED);
			subpanel_PARAMETROS_RED.add(slider_ancho_banda_consumido);
			subpanel_PARAMETROS_RED.add(slider_numero_conexiones);
		panel_RECURSOS_SISTEMA.add("South",subpanel_recursos_sistema_INVISIBLE);
			subpanel_recursos_sistema_INVISIBLE.add(slider_INVISIBLE1);
			subpanel_recursos_sistema_INVISIBLE.add(slider_INVISIBLE2);
			subpanel_recursos_sistema_INVISIBLE.add(slider_INVISIBLE3);

		slider_saturacion_CPU.addChangeListener(new Slider_Listener());
		slider_saturacion_RAM.addChangeListener(new Slider_Listener());
		slider_actividad_DD.addChangeListener(new Slider_Listener());
		slider_ancho_banda_consumido.addChangeListener(new Slider_Listener());
		slider_numero_conexiones.addChangeListener(new Slider_Listener());
	}

	private void creacion_panel_datos_PELIGROSIDAD_CONEXION() {
		subpanel_PARAMETROS_PROCEDENCIA_Y_FIN.setBorder(BorderFactory.createTitledBorder(borde_subpaneles,"
Parametros de procedencia y fin ",TitledBorder.CENTER,TitledBorder.TOP));
		subpanel_PARAMETROS_TEMPORALES.setBorder(BorderFactory.createTitledBorder(borde_subpaneles,"
Parametros temporales ",TitledBorder.CENTER,TitledBorder.TOP));
		panel_PELIGROSIDAD_CONEXION.add("North",subpanel_PARAMETROS_PROCEDENCIA_Y_FIN);
			subpanel_PARAMETROS_PROCEDENCIA_Y_FIN.add(slider_origen_anormal);
			subpanel_PARAMETROS_PROCEDENCIA_Y_FIN.add(slider_destino_anormal);
		panel_PELIGROSIDAD_CONEXION.add("Center",subpanel_PARAMETROS_TEMPORALES);
			subpanel_PARAMETROS_TEMPORALES.add(slider_duracion_conexion);
			subpanel_PARAMETROS_TEMPORALES.add(slider_horario_conexion);
		panel_PELIGROSIDAD_CONEXION.add("South",subpanel_peligrosidad_conexion_INVISIBLE);
			subpanel_peligrosidad_conexion_INVISIBLE.add(slider_INVISIBLE4);
			subpanel_peligrosidad_conexion_INVISIBLE.add(slider_INVISIBLE5);
			subpanel_peligrosidad_conexion_INVISIBLE.add(slider_INVISIBLE6);
			subpanel_peligrosidad_conexion_INVISIBLE.add(slider_INVISIBLE7);

		//Asociamos los sliders a sus gestores de eventos
		slider_origen_anormal.addChangeListener(new Slider_Listener());
		slider_destino_anormal.addChangeListener(new Slider_Listener());
		slider_duracion_conexion.addChangeListener(new Slider_Listener());
		slider_horario_conexion.addChangeListener(new Slider_Listener());
	}

	private void creacion_panel_datos_VULNERABILIDAD_SISTEMA() {
		subpanel_SEGURIDAD_INTERNA.setBorder(BorderFactory.createTitledBorder(borde_subpaneles," Seguridad Interna
",TitledBorder.CENTER,TitledBorder.TOP));
		subpanel_SERVICIOS_USADOS.setBorder(BorderFactory.createTitledBorder(borde_subpaneles," Servicios Usados
",TitledBorder.CENTER,TitledBorder.TOP));
		panel_VULNERABILIDAD_SISTEMA.add("North",subpanel_SEGURIDAD_INTERNA);
			subpanel_SEGURIDAD_INTERNA.add(slider_modificaciones_SO);
			subpanel_SEGURIDAD_INTERNA.add(slider_copia_ficheros);
		panel_VULNERABILIDAD_SISTEMA.add("Center",subpanel_NUMERO_PROCESOS_ACTIVOS);
			subpanel_NUMERO_PROCESOS_ACTIVOS.add(slider_numero_procesos_activos);
			subpanel_NUMERO_PROCESOS_ACTIVOS.add(slider_INVISIBLE8);
		panel_VULNERABILIDAD_SISTEMA.add("South",subpanel_SERVICIOS_USADOS);
```

```
                        subpanel_SERVICIOS_USADOS.add(slider_telnet);
                        subpanel_SERVICIOS_USADOS.add(slider_web);
                        subpanel_SERVICIOS_USADOS.add(slider_ftp);
                        subpanel_SERVICIOS_USADOS.add(slider_email);
                        subpanel_SERVICIOS_USADOS.add(slider_dns);
                        subpanel_SERVICIOS_USADOS.add(slider_login);

                slider_modificaciones_SO.addChangeListener(new Slider_Listener());
                slider_copia_ficheros.addChangeListener(new Slider_Listener());
                slider_numero_procesos_activos.addChangeListener(new Slider_Listener());
                slider_telnet.addChangeListener(new Slider_Listener());
                slider_web.addChangeListener(new Slider_Listener());
                slider_ftp.addChangeListener(new Slider_Listener());
                slider_email.addChangeListener(new Slider_Listener());
                slider_dns.addChangeListener(new Slider_Listener());
                slider_login.addChangeListener(new Slider_Listener());
        }

        //*************************************************************************************************
        // CREACION DEL PANEL DE NIVEL DE SEGURIDAD DEL SISTEMA DETECTOR DE ANOMALIAS (FIABILIDAD,
SENSIBILIDAD) *
        //*************************************************************************************************

        private void creacion_panel_NIVEL_SEGURIDAD_SISTEMA_DETECTOR_ANOMALIAS() {
                Color color_fondo = Color.cyan; //LIGHT_GRAY;
                TitledBorder titulo = BorderFactory.createTitledBorder(borde_subpaneles," NIVEL DE SEGURIDAD
",TitledBorder.CENTER,TitledBorder.TOP);
                panel_NIVEL_SEGURIDAD_SISTEMA_DETECTOR_ANOMALIAS.setBorder(titulo);
                panel_NIVEL_SEGURIDAD_SISTEMA_DETECTOR_ANOMALIAS.setBackground(color_fondo);
                panel_NIVEL_SEGURIDAD_SISTEMA_DETECTOR_ANOMALIAS.add(slider_nivel_fiabilidad);
                panel_NIVEL_SEGURIDAD_SISTEMA_DETECTOR_ANOMALIAS.add(slider_nivel_sensibilidad);
                slider_nivel_fiabilidad.setBackground(color_fondo);
                slider_nivel_sensibilidad.setBackground(color_fondo);
                //Asociamos los sliders a sus gestores de eventos
                slider_nivel_fiabilidad.addChangeListener(new Slider_Listener());
                slider_nivel_sensibilidad.addChangeListener(new Slider_Listener());
        }

        //*************************************************************************
        // CREACION DEL PANEL DE HERRAMIENTAS (BARRA DE PROGRESO, COMENTARIOS) *
        //*************************************************************************

        private void creacion_panel_herramientas() {
                panel_herramientas.add("North",barra_progreso);
                        barra_progreso.setStringPainted(true); //muestra en la barra de progreso el valor numerico correspondiente
                        barra_progreso.setString("OBSERVE AQUI EL VALOR DE LA VARIABLE MODIFICADA");
                panel_herramientas.add("Center",definiciones_datos_sistema_experto);
                        definiciones_datos_sistema_experto.setLineWrap(true);
                        definiciones_datos_sistema_experto.setWrapStyleWord(true);
                        definiciones_datos_sistema_experto.setRows(2);
                        definiciones_datos_sistema_experto.setEditable(false);
                        definiciones_datos_sistema_experto.setFont(new Font("Courier",Font.ITALIC+Font.BOLD,12));
                        definiciones_datos_sistema_experto.setText("OBSERVE AQUI LA DEFINICION DE LA VARIABLE
MODIFICADA");
                        definiciones_datos_sistema_experto.setBackground(Color.yellow);
                        definiciones_datos_sistema_experto.setForeground(Color.blue);
        }

        //*************************************************************************
        // CONSTRUCTOR DEL OBJETO Tinterfaz_xfuzzy (OBJETO INTERFAZ DE LA APLICACION) *
        //*************************************************************************

        Tinterfaz_xfuzzy() {
                addWindowListener(new WindowAdapter() {
                        public void windowClosing(WindowEvent e) {
                                System.exit(0);
                        }
                });
                setTitle("DETECTOR DE ANOMALIAS version 1.1."); //Titulo de la ventana de la aplicacion
                setSize(650,680); //Determinamos las dimensiones de la ventana (ancho x alto)
                setResizable(false); //Dimensiones de la ventana no modificables
                admon_disenio=(JPanel)getContentPane(); //Administrador de diseño a traves del cual se añaden el resto de
componentes al "JPanel"
                //CENTRAR EL JFRAME EN LA PANTALLA
                Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
                Dimension size = getSize();
```

```
		screenSize.height = screenSize.height/2;
		screenSize.width = screenSize.width/2;
		size.height = size.height/2;
		size.width = size.width/2;
		int y = screenSize.height - size.height;
		int x = screenSize.width - size.width;
		setLocation(x, y);
		//DEFINICION DEL MENU DE LA APLICACION
		creacion_menu();
		//COLOCACION SOBRE EL FRAME DE LOS OBJETOS QUE LO COMPONEN
		admon_disenio.setLayout(new BorderLayout());
		admon_disenio.add("North",panel_herramientas);
			creacion_panel_herramientas();
		admon_disenio.add("Center",panel_datos);
			creacion_panel_datos_RECURSOS_SISTEMA();
			creacion_panel_datos_PELIGROSIDAD_CONEXION();
			creacion_panel_datos_VULNERABILIDAD_SISTEMA();
			panel_datos.addTab( " Recursos del Sistema ", panel_RECURSOS_SISTEMA);
			panel_datos.addTab( " Peligrosidad de la Conexion ", panel_PELIGROSIDAD_CONEXION);
			panel_datos.addTab( " Vulnerabilidad del Sistema ", panel_VULNERABILIDAD_SISTEMA);
		admon_disenio.add("South",panel_NIVEL_SEGURIDAD_SISTEMA_DETECTOR_ANOMALIAS);
			creacion_panel_NIVEL_SEGURIDAD_SISTEMA_DETECTOR_ANOMALIAS();
	}
};

//*****************************************
// CLASE QUE SIRVE DE "PROGRAMA PRINCIPAL" *
//*****************************************

class interfaz
{
	public static void main(String[] args)
	{
		Tparametros_anomalia parametros_anomalia = new Tparametros_anomalia();
		Tinterfaz_xfuzzy i = new Tinterfaz_xfuzzy();
		i.show();
	}
}
```

<FozzimA.java>

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//     Fuzzy Inference Engine fozzimA      //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

public class fozzimA implements FuzzyInferenceEngine {

 //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
 //      Membership function of an input variable       //
 //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

 private abstract class InnerMembershipFunction {
  double min, max, step;
  abstract double param(int i);
  double center() { return 0; }
  double basis() { return 0; }
  abstract double isEqual(double x);

  double isSmallerOrEqual(double x) {
   double degree=0, mu;
   for(double y=max; y>=x ; y-=step) if((mu = isEqual(y))>degree) degree=mu;
   return degree;
  }

  double isGreaterOrEqual(double x) {
   double degree=0, mu;
   for(double y=min; y<=x ; y+=step) if((mu = isEqual(y))>degree) degree=mu;
   return degree;
  }

  double isEqual(MembershipFunction mf) {
   if(mf instanceof FuzzySingleton)
    { return isEqual( ((FuzzySingleton) mf).getValue()); }
   if((mf instanceof OutputMembershipFunction) &&
      ((OutputMembershipFunction) mf).isDiscrete() ) {
    double[][] val = ((OutputMembershipFunction) mf).getDiscreteValues();
    double deg = 0;
    for(int i=0; i<val.length; i++){
     double mu = isEqual(val[i][0]);
     double minmu = (mu<val[i][1] ? mu : val[i][1]);
     if( deg<minmu ) deg = minmu;
    }
    return deg;
   }
   double mu1,mu2,minmu,degree=0;
   for(double x=min; x<=max; x+=step){
    mu1 = mf.compute(x);
    mu2 = isEqual(x);
    minmu = (mu1<mu2 ? mu1 : mu2);
    if( degree<minmu ) degree = minmu;
   }
   return degree;
  }

  double isGreaterOrEqual(MembershipFunction mf) {
   if(mf instanceof FuzzySingleton)
    { return isGreaterOrEqual( ((FuzzySingleton) mf).getValue()); }
   if((mf instanceof OutputMembershipFunction) &&
      ((OutputMembershipFunction) mf).isDiscrete() ) {
    double[][] val = ((OutputMembershipFunction) mf).getDiscreteValues();
    double deg = 0;
    for(int i=0; i<val.length; i++){
     double mu = isGreaterOrEqual(val[i][0]);
     double minmu = (mu<val[i][1] ? mu : val[i][1]);
     if( deg<minmu ) deg = minmu;
    }
    return deg;
   }
   double mu1,mu2,minmu,degree=0,greq=0;
   for(double x=min; x<=max; x+=step){
    mu1 = mf.compute(x);
    mu2 = isEqual(x);
    if( mu2>greq ) greq = mu2;
```

```
  if( mu1<greq ) minmu = mu1; else minmu = greq;
  if( degree<minmu ) degree = minmu;
 }
 return degree;
}

double isSmallerOrEqual(MembershipFunction mf) {
 if(mf instanceof FuzzySingleton)
  { return isSmallerOrEqual( ((FuzzySingleton) mf).getValue()); }
 if((mf instanceof OutputMembershipFunction) &&
    ((OutputMembershipFunction) mf).isDiscrete() ) {
  double[][] val = ((OutputMembershipFunction) mf).getDiscreteValues();
  double deg = 0;
  for(int i=0; i<val.length; i++){
   double mu = isSmallerOrEqual(val[i][0]);
   double minmu = (mu<val[i][1] ? mu : val[i][1]);
   if( deg<minmu ) deg = minmu;
  }
  return deg;
 }
 double mu1,mu2,minmu,degree=0,smeq=0;
 for(double x=max; x>=min; x-=step){
  mu1 = mf.compute(x);
  mu2 = isEqual(x);
  if( mu2>smeq ) smeq = mu2;
  if( mu1<smeq ) minmu = mu1; else minmu = smeq;
  if( degree<minmu ) degree = minmu;
 }
 return degree;
}

double isGreater(MembershipFunction mf, InnerOperatorset op) {
 if(mf instanceof FuzzySingleton)
  { return op.not(isSmallerOrEqual( ((FuzzySingleton) mf).getValue())); }
 if((mf instanceof OutputMembershipFunction) &&
    ((OutputMembershipFunction) mf).isDiscrete() ) {
  double[][] val = ((OutputMembershipFunction) mf).getDiscreteValues();
  double deg = 0;
  for(int i=0; i<val.length; i++){
   double mu = op.not(isSmallerOrEqual(val[i][0]));
   double minmu = (mu<val[i][1] ? mu : val[i][1]);
   if( deg<minmu ) deg = minmu;
  }
  return deg;
 }
 double mu1,mu2,minmu,gr,degree=0,smeq=0;
 for(double x=max; x>=min; x-=step){
  mu1 = mf.compute(x);
  mu2 = isEqual(x);
  if( mu2>smeq ) smeq = mu2;
  gr = op.not(smeq);
  minmu = ( mu1<gr ? mu1 : gr);
  if( degree<minmu ) degree = minmu;
 }
 return degree;
}

double isSmaller(MembershipFunction mf, InnerOperatorset op) {
 if(mf instanceof FuzzySingleton)
  { return op.not(isGreaterOrEqual( ((FuzzySingleton) mf).getValue())); }
 if((mf instanceof OutputMembershipFunction) &&
    ((OutputMembershipFunction) mf).isDiscrete() ) {
  double[][] val = ((OutputMembershipFunction) mf).getDiscreteValues();
  double deg = 0;
  for(int i=0; i<val.length; i++){
   double mu = op.not(isGreaterOrEqual(val[i][0]));
   double minmu = (mu<val[i][1] ? mu : val[i][1]);
   if( deg<minmu ) deg = minmu;
  }
  return deg;
 }
 double mu1,mu2,minmu,sm,degree=0,greq=0;
 for(double x=min; x<=max; x+=step){
  mu1 = mf.compute(x);
  mu2 = isEqual(x);
  if( mu2>greq ) greq = mu2;
```

```
  sm = op.not(greq);
  minmu = ( mu1<sm ? mu1 : sm);
  if( degree<minmu ) degree = minmu;
 }
 return degree;
}

double isNotEqual(MembershipFunction mf, InnerOperatorset op) {
 if(mf instanceof FuzzySingleton)
  { return op.not(isEqual( ((FuzzySingleton) mf).getValue())); }
 if((mf instanceof OutputMembershipFunction) &&
    ((OutputMembershipFunction) mf).isDiscrete() ) {
  double[][] val = ((OutputMembershipFunction) mf).getDiscreteValues();
  double deg = 0;
  for(int i=0; i<val.length; i++){
   double mu = op.not(isEqual(val[i][0]));
   double minmu = (mu<val[i][1] ? mu : val[i][1]);
   if( deg<minmu ) deg = minmu;
  }
  return deg;
 }
 double mu1,mu2,minmu,degree=0;
 for(double x=min; x<=max; x+=step){
  mu1 = mf.compute(x);
  mu2 = op.not(isEqual(x));
  minmu = (mu1<mu2 ? mu1 : mu2);
  if( degree<minmu ) degree = minmu;
 }
 return degree;
}

double isApproxEqual(MembershipFunction mf, InnerOperatorset op) {
 if(mf instanceof FuzzySingleton)
  { return op.moreorless(isEqual( ((FuzzySingleton) mf).getValue())); }
 if((mf instanceof OutputMembershipFunction) &&
    ((OutputMembershipFunction) mf).isDiscrete() ) {
  double[][] val = ((OutputMembershipFunction) mf).getDiscreteValues();
  double deg = 0;
  for(int i=0; i<val.length; i++){
   double mu = op.moreorless(isEqual(val[i][0]));
   double minmu = (mu<val[i][1] ? mu : val[i][1]);
   if( deg<minmu ) deg = minmu;
  }
  return deg;
 }
 double mu1,mu2,minmu,degree=0;
 for(double x=min; x<=max; x+=step){
  mu1 = mf.compute(x);
  mu2 = op.moreorless(isEqual(x));
  minmu = (mu1<mu2 ? mu1 : mu2);
  if( degree<minmu ) degree = minmu;
 }
 return degree;
}

double isVeryEqual(MembershipFunction mf, InnerOperatorset op) {
 if(mf instanceof FuzzySingleton)
  { return op.very(isEqual( ((FuzzySingleton) mf).getValue())); }
 if((mf instanceof OutputMembershipFunction) &&
    ((OutputMembershipFunction) mf).isDiscrete() ) {
  double[][] val = ((OutputMembershipFunction) mf).getDiscreteValues();
  double deg = 0;
  for(int i=0; i<val.length; i++){
   double mu = op.very(isEqual(val[i][0]));
   double minmu = (mu<val[i][1] ? mu : val[i][1]);
   if( deg<minmu ) deg = minmu;
  }
  return deg;
 }
 double mu1,mu2,minmu,degree=0;
 for(double x=min; x<=max; x+=step){
  mu1 = mf.compute(x);
  mu2 = op.very(isEqual(x));
  minmu = (mu1<mu2 ? mu1 : mu2);
  if( degree<minmu ) degree = minmu;
 }
```

```
  return degree;
 }

 double isSlightlyEqual(MembershipFunction mf, InnerOperatorset op) {
  if(mf instanceof FuzzySingleton)
   { return op.slightly(isEqual( ((FuzzySingleton) mf).getValue())); }
  if((mf instanceof OutputMembershipFunction) &&
     ((OutputMembershipFunction) mf).isDiscrete() ) {
   double[][] val = ((OutputMembershipFunction) mf).getDiscreteValues();
   double deg = 0;
   for(int i=0; i<val.length; i++){
    double mu = op.slightly(isEqual(val[i][0]));
    double minmu = (mu<val[i][1] ? mu : val[i][1]);
    if( deg<minmu ) deg = minmu;
   }
   return deg;
  }
  double mu1,mu2,minmu,degree=0;
  for(double x=min; x<=max; x+=step){
   mu1 = mf.compute(x);
   mu2 = op.slightly(isEqual(x));
   minmu = (mu1<mu2 ? mu1 : mu2);
   if( degree<minmu ) degree = minmu;
  }
  return degree;
 }
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//          Abstract class of an operator set          //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private abstract class InnerOperatorset {
 abstract double and(double a, double b);
 abstract double or(double a, double b);
 abstract double also(double a, double b);
 abstract double imp(double a, double b);
 abstract double not(double a);
 abstract double very(double a);
 abstract double moreorless(double a);
 abstract double slightly(double a);
 abstract double defuz(OutputMembershipFunction mf);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//      Class for the conclusion of a fuzzy rule       //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private class InnerConclusion {
 private double degree;
 private InnerMembershipFunction mf;
 private InnerOperatorset op;

 InnerConclusion(double degree, InnerMembershipFunction mf, InnerOperatorset op) {
  this.op = op;
  this.degree = degree;
  this.mf = mf;
 }

 public double degree() { return degree; }
 public double compute(double x) { return op.imp(degree,mf.isEqual(x)); }
 public double center() { return mf.center(); }
 public double basis() { return mf.basis(); }
 public double param(int i) { return mf.param(i); }
 public double min() { return mf.min; }
 public double max() { return mf.max; }
 public double step() { return mf.step; }
 public boolean isSingleton() {
  return mf.getClass().getName().endsWith("MF_xfl_singleton");
 }
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//      Membership function of an output variable      //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
```

```
private class OutputMembershipFunction implements MembershipFunction {
 public InnerConclusion[] conc;
 public double[] input;
 private InnerOperatorset op;

 OutputMembershipFunction() {
  this.conc = new InnerConclusion[0];
 }

 public void set(int size, InnerOperatorset op, double[] input) {
  this.input = input;
  this.op = op;
  this.conc = new InnerConclusion[size];
 }

 public void set(int pos, double dg, InnerMembershipFunction imf) {
  this.conc[pos] = new InnerConclusion(dg,imf,op);
 }

 public double compute(double x) {
  double dom = conc[0].compute(x);
  for(int i=1; i<conc.length; i++) dom = op.also(dom,conc[i].compute(x));
  return dom;
 }

 public double defuzzify() {
  return op.defuz(this);
 }

 public double min() {
  return conc[0].min();
 }

 public double max() {
  return conc[0].max();
 }

 public double step() {
  return conc[0].step();
 }

 public boolean isDiscrete() {
  for(int i=0; i<conc.length; i++) if(!conc[i].isSingleton()) return false;
  return true;
 }

 public double[][] getDiscreteValues() {
  double[][] value = new double[conc.length][2];
  for(int i=0; i<conc.length; i++) {
   value[i][0] = conc[i].param(0);
   value[i][1] = conc[i].degree();
  }
  return value;
 }

}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//      Membership function MF_xfl_trapezoid      //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

 private class MF_xfl_trapezoid extends InnerMembershipFunction {
  double a;
  double b;
  double c;
  double d;

  MF_xfl_trapezoid( double min, double max, double step, double param[]){
   super.min = min;
   super.max = max;
   super.step = step;
   this.a = param[0];
   this.b = param[1];
   this.c = param[2];
   this.d = param[3];
  }
```

```
  double param(int _i) {
   switch(_i) {
    case 0: return a;
    case 1: return b;
    case 2: return c;
    case 3: return d;
    default: return 0;
   }
  }
  double isEqual(double x) {
   return (x<a || x>d? 0: (x<b? (x-a)/(b-a) : (x<c?1 : (d-x)/(d-c))));
  }
  double isGreaterOrEqual(double x) {
   return (x<a? 0 : (x>b? 1 : (x-a)/(b-a) ));
  }
  double isSmallerOrEqual(double x) {
   return (x<c? 1 : (x>d? 0 : (d-x)/(d-c) ));
  }
  double center() {
   return (b+c)/2;
  }
  double basis() {
   return (d-a);
  }
 }

//+------------------------------------------------------+//
//      Membership function MF_xfl_singleton      //
//+------------------------------------------------------+//

 private class MF_xfl_singleton extends InnerMembershipFunction {
  double a;

  MF_xfl_singleton( double min, double max, double step, double param[]){
   super.min = min;
   super.max = max;
   super.step = step;
   this.a = param[0];
  }
  double param(int _i) {
   switch(_i) {
    case 0: return a;
    default: return 0;
   }
  }
  double isEqual(double x) {
   return (x==a? 1 : 0);
  }
  double isGreaterOrEqual(double x) {
   return (x>=a? 1 : 0);
  }
  double isSmallerOrEqual(double x) {
   return (x<=a? 1 : 0);
  }
  double center() {
   return a;
  }
 }

//+------------------------------------------------------+//
//     Operator set OP_operatorset2      //
//+------------------------------------------------------+//

 private class OP_operatorset2 extends InnerOperatorset {
  double and(double a, double b) {
    return (a<b? a : b);
  }
  double or(double a, double b) {
    return (a>b? a : b);
  }
  double also(double a, double b) {
    return (a>b? a : b);
  }
  double imp(double a, double b) {
    return (a<b? a : b);
  }
```

```
 double not(double a) {
   return 1-a;
 }
 double very(double a) {
  double w = 2.0;
   return Math.pow(a,w);
 }
 double moreorless(double a) {
  double w = 0.5;
   return Math.pow(a,w);
 }
 double slightly(double a) {
   return 4*a*(1-a);
 }
double defuz(OutputMembershipFunction mf) {
  double min = mf.min();
  double max = mf.max();
  double step = mf.step();
 double out=min, maximum = 0;
 for(double x=min; x<=max; x+=step) {
  double m = mf.compute(x);
  if(m>=maximum) { maximum = m; out = x; }
 }
 return out;
 }
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//     Operator set OP__default_      //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

 private class OP__default_ extends InnerOperatorset {
  double and(double a, double b) {
    return (a<b? a : b);
  }
  double or(double a, double b) {
    return (a>b? a : b);
  }
  double also(double a, double b) {
    return (a>b? a : b);
  }
  double imp(double a, double b) {
    return (a<b? a : b);
  }
  double not(double a) {
    return 1-a;
  }
  double very(double a) {
   double w = 2.0;
    return Math.pow(a,w);
  }
  double moreorless(double a) {
   double w = 0.5;
    return Math.pow(a,w);
  }
  double slightly(double a) {
    return 4*a*(1-a);
  }
 double defuz(OutputMembershipFunction mf) {
   double min = mf.min();
   double max = mf.max();
   double step = mf.step();
  double num=0, denom=0;
  for(double x=min; x<=max; x+=step) {
   double m = mf.compute(x);
   num += x*m;
   denom += m;
  }
  if(denom==0) return (min+max)/2;
  return num/denom;
  }
 }

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Type TP_tiempo  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
```

```
private class TP_tiempo {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_bajo[] = { -12.5,0.0,25.0,37.5 };
 double _p_medio[] = { 25.0,37.5,62.5,75.0 };
 double _p_alto[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid bajo = new MF_xfl_trapezoid(min,max,step,_p_bajo);
 MF_xfl_trapezoid medio = new MF_xfl_trapezoid(min,max,step,_p_medio);
 MF_xfl_trapezoid alto = new MF_xfl_trapezoid(min,max,step,_p_alto);
}

//+------------------------------------------------------+//
//  Type TP_NumConex  //
//+------------------------------------------------------+//

private class TP_NumConex {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_muypocas[] = { -7.142857142857143,0.0,14.285714285714286,21.42857142857143 };
 double _p_pocas[] = { 14.285714285714286,21.42857142857143,35.714285714285715,42.85714285714286 };
 double _p_normales[] = { 35.714285714285715,42.85714285714286,57.142857142857146,64.28571428571429 };
 double _p_muchas[] = { 57.142857142857146,64.28571428571429,78.57142857142857,85.71428571428572 };
 double _p_demasiadas[] = { 78.57142857142857,85.71428571428572,100.0,107.14285714285715 };
 MF_xfl_trapezoid muypocas = new MF_xfl_trapezoid(min,max,step,_p_muypocas);
 MF_xfl_trapezoid pocas = new MF_xfl_trapezoid(min,max,step,_p_pocas);
 MF_xfl_trapezoid normales = new MF_xfl_trapezoid(min,max,step,_p_normales);
 MF_xfl_trapezoid muchas = new MF_xfl_trapezoid(min,max,step,_p_muchas);
 MF_xfl_trapezoid demasiadas = new MF_xfl_trapezoid(min,max,step,_p_demasiadas);
}

//+------------------------------------------------------+//
//  Type TP_probabilidadanomalia  //
//+------------------------------------------------------+//

private class TP_probabilidadanomalia {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_poca[] = { -12.5,0.0,25.0,37.5 };
 double _p_normal[] = { 25.0,37.5,62.5,75.0 };
 double _p_alta[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid poca = new MF_xfl_trapezoid(min,max,step,_p_poca);
 MF_xfl_trapezoid normal = new MF_xfl_trapezoid(min,max,step,_p_normal);
 MF_xfl_trapezoid alta = new MF_xfl_trapezoid(min,max,step,_p_alta);
}

//+------------------------------------------------------+//
//  Type TP_probabilidadfalsos  //
//+------------------------------------------------------+//

private class TP_probabilidadfalsos {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_poca[] = { -12.5,0.0,25.0,37.5 };
 double _p_normal[] = { 25.0,37.5,62.5,75.0 };
 double _p_alta[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid poca = new MF_xfl_trapezoid(min,max,step,_p_poca);
 MF_xfl_trapezoid normal = new MF_xfl_trapezoid(min,max,step,_p_normal);
 MF_xfl_trapezoid alta = new MF_xfl_trapezoid(min,max,step,_p_alta);
}

//+------------------------------------------------------+//
//  Type TP_nivelseguridad  //
//+------------------------------------------------------+//

private class TP_nivelseguridad {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_bajo[] = { -12.5,0.0,25.0,37.5 };
 double _p_medio[] = { 25.0,37.5,62.5,75.0 };
 double _p_alto[] = { 62.5,75.0,100.0,112.5 };
```

```
 MF_xfl_trapezoid bajo = new MF_xfl_trapezoid(min,max,step,_p_bajo);
 MF_xfl_trapezoid medio = new MF_xfl_trapezoid(min,max,step,_p_medio);
 MF_xfl_trapezoid alto = new MF_xfl_trapezoid(min,max,step,_p_alto);
}

//+------------------------------------------------------+//
//  Type TP_anchodebandacons  //
//+------------------------------------------------------+//

private class TP_anchodebandacons {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_muypoco[] = { -7.142857142857143,0.0,14.285714285714286,21.42857142857143 };
 double _p_poco[] = { 14.285714285714286,21.42857142857143,35.714285714285715,42.85714285714286 };
 double _p_normal[] = { 35.714285714285715,42.85714285714286,57.142857142857146,64.28571428571429 };
 double _p_mucho[] = { 57.142857142857146,64.28571428571429,78.57142857142857,85.71428571428572 };
 double _p_demasiado[] = { 78.57142857142857,85.71428571428572,100.0,107.14285714285715 };
 MF_xfl_trapezoid muypoco = new MF_xfl_trapezoid(min,max,step,_p_muypoco);
 MF_xfl_trapezoid poco = new MF_xfl_trapezoid(min,max,step,_p_poco);
 MF_xfl_trapezoid normal = new MF_xfl_trapezoid(min,max,step,_p_normal);
 MF_xfl_trapezoid mucho = new MF_xfl_trapezoid(min,max,step,_p_mucho);
 MF_xfl_trapezoid demasiado = new MF_xfl_trapezoid(min,max,step,_p_demasiado);
}

//+------------------------------------------------------+//
//  Type TP_nivelsaturacionred  //
//+------------------------------------------------------+//

private class TP_nivelsaturacionred {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_muypocasaturacion[] = { -7.142857142857143,0.0,14.285714285714286,21.42857142857143 };
 double _p_pocasaturacion[] = { 14.285714285714286,21.42857142857143,35.714285714285715,42.85714285714286 };
 double _p_normalsaturacion[] = { 35.714285714285715,42.85714285714286,57.142857142857146,64.28571428571429 };
 double _p_muchasaturacion[] = { 57.142857142857146,64.28571428571429,78.57142857142857,85.71428571428572 };
 double _p_demasiadasaturacion[] = { 78.57142857142857,85.71428571428572,100.0,107.14285714285715 };
 MF_xfl_trapezoid muypocasaturacion = new MF_xfl_trapezoid(min,max,step,_p_muypocasaturacion);
 MF_xfl_trapezoid pocasaturacion = new MF_xfl_trapezoid(min,max,step,_p_pocasaturacion);
 MF_xfl_trapezoid normalsaturacion = new MF_xfl_trapezoid(min,max,step,_p_normalsaturacion);
 MF_xfl_trapezoid muchasaturacion = new MF_xfl_trapezoid(min,max,step,_p_muchasaturacion);
 MF_xfl_trapezoid demasiadasaturacion = new MF_xfl_trapezoid(min,max,step,_p_demasiadasaturacion);
}

//+------------------------------------------------------+//
//  Type TP_horario  //
//+------------------------------------------------------+//

private class TP_horario {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_hora0a5[] = { -7.142857142857143,0.0,14.285714285714286,21.42857142857143 };
 double _p_hora6a11[] = { 14.285714285714286,21.42857142857143,35.714285714285715,42.85714285714286 };
 double _p_hora11a16[] = { 35.714285714285715,42.85714285714286,57.142857142857146,64.28571428571429 };
 double _p_hora16a21[] = { 57.142857142857146,64.28571428571429,78.57142857142857,85.71428571428572 };
 double _p_hora21a24[] = { 78.57142857142857,85.71428571428572,100.0,107.14285714285715 };
 MF_xfl_trapezoid hora0a5 = new MF_xfl_trapezoid(min,max,step,_p_hora0a5);
 MF_xfl_trapezoid hora6a11 = new MF_xfl_trapezoid(min,max,step,_p_hora6a11);
 MF_xfl_trapezoid hora11a16 = new MF_xfl_trapezoid(min,max,step,_p_hora11a16);
 MF_xfl_trapezoid hora16a21 = new MF_xfl_trapezoid(min,max,step,_p_hora16a21);
 MF_xfl_trapezoid hora21a24 = new MF_xfl_trapezoid(min,max,step,_p_hora21a24);
}

//+------------------------------------------------------+//
//  Type TP_consumoCPU  //
//+------------------------------------------------------+//

private class TP_consumoCPU {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_bajo[] = { -12.5,0.0,25.0,37.5 };
 double _p_medio[] = { 25.0,37.5,62.5,75.0 };
 double _p_alto[] = { 62.5,75.0,100.0,112.5 };
```

```
 MF_xfl_trapezoid bajo = new MF_xfl_trapezoid(min,max,step,_p_bajo);
 MF_xfl_trapezoid medio = new MF_xfl_trapezoid(min,max,step,_p_medio);
 MF_xfl_trapezoid alto = new MF_xfl_trapezoid(min,max,step,_p_alto);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Type TP_consumoRAM  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private class TP_consumoRAM {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_bajo[] = { -12.5,0.0,25.0,37.5 };
 double _p_medio[] = { 25.0,37.5,62.5,75.0 };
 double _p_alto[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid bajo = new MF_xfl_trapezoid(min,max,step,_p_bajo);
 MF_xfl_trapezoid medio = new MF_xfl_trapezoid(min,max,step,_p_medio);
 MF_xfl_trapezoid alto = new MF_xfl_trapezoid(min,max,step,_p_alto);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Type TP_actividadDiscoDuro  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private class TP_actividadDiscoDuro {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_bajo[] = { -12.5,0.0,25.0,37.5 };
 double _p_medio[] = { 25.0,37.5,62.5,75.0 };
 double _p_alto[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid bajo = new MF_xfl_trapezoid(min,max,step,_p_bajo);
 MF_xfl_trapezoid medio = new MF_xfl_trapezoid(min,max,step,_p_medio);
 MF_xfl_trapezoid alto = new MF_xfl_trapezoid(min,max,step,_p_alto);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Type TP_numerodepuertos  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private class TP_numerodepuertos {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_menosde100[] = { -12.5,0.0,25.0,37.5 };
 double _p_entre100y1000[] = { 25.0,37.5,62.5,75.0 };
 double _p_masde1000[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid menosde100 = new MF_xfl_trapezoid(min,max,step,_p_menosde100);
 MF_xfl_trapezoid entre100y1000 = new MF_xfl_trapezoid(min,max,step,_p_entre100y1000);
 MF_xfl_trapezoid masde1000 = new MF_xfl_trapezoid(min,max,step,_p_masde1000);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Type TP_origenAnormal  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private class TP_origenAnormal {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_muypoco[] = { -7.142857142857143,0.0,14.285714285714286,21.42857142857143 };
 double _p_poco[] = { 14.285714285714286,21.42857142857143,35.714285714285715,42.85714285714286 };
 double _p_normal[] = { 35.714285714285715,42.85714285714286,57.142857142857146,64.28571428571429 };
 double _p_mucho[] = { 57.142857142857146,64.28571428571429,78.57142857142857,85.71428571428572 };
 double _p_demasiado[] = { 78.57142857142857,85.71428571428572,100.0,107.14285714285715 };
 MF_xfl_trapezoid muypoco = new MF_xfl_trapezoid(min,max,step,_p_muypoco);
 MF_xfl_trapezoid poco = new MF_xfl_trapezoid(min,max,step,_p_poco);
 MF_xfl_trapezoid normal = new MF_xfl_trapezoid(min,max,step,_p_normal);
 MF_xfl_trapezoid mucho = new MF_xfl_trapezoid(min,max,step,_p_mucho);
 MF_xfl_trapezoid demasiado = new MF_xfl_trapezoid(min,max,step,_p_demasiado);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Type TP_destinoAnormal  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
```

```
private class TP_destinoAnormal {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_muypoco[] = { -7.142857142857143,0.0,14.285714285714286,21.42857142857143 };
 double _p_poco[] = { 14.285714285714286,21.42857142857143,35.714285714285715,42.85714285714286 };
 double _p_normal[] = { 35.714285714285715,42.85714285714286,57.142857142857146,64.28571428571429 };
 double _p_mucho[] = { 57.142857142857146,64.28571428571429,78.57142857142857,85.71428571428572 };
 double _p_demasiado[] = { 78.57142857142857,85.71428571428572,100.0,107.14285714285715 };
 MF_xfl_trapezoid muypoco = new MF_xfl_trapezoid(min,max,step,_p_muypoco);
 MF_xfl_trapezoid poco = new MF_xfl_trapezoid(min,max,step,_p_poco);
 MF_xfl_trapezoid normal = new MF_xfl_trapezoid(min,max,step,_p_normal);
 MF_xfl_trapezoid mucho = new MF_xfl_trapezoid(min,max,step,_p_mucho);
 MF_xfl_trapezoid demasiado = new MF_xfl_trapezoid(min,max,step,_p_demasiado);
}

//+------------------------------------------------------+//
//  Type TP_fiabilidad  //
//+------------------------------------------------------+//

private class TP_fiabilidad {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_bajo[] = { -12.5,0.0,25.0,37.5 };
 double _p_medio[] = { 25.0,37.5,62.5,75.0 };
 double _p_alto[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid bajo = new MF_xfl_trapezoid(min,max,step,_p_bajo);
 MF_xfl_trapezoid medio = new MF_xfl_trapezoid(min,max,step,_p_medio);
 MF_xfl_trapezoid alto = new MF_xfl_trapezoid(min,max,step,_p_alto);
}

//+------------------------------------------------------+//
//  Type TP_sensibilidad  //
//+------------------------------------------------------+//

private class TP_sensibilidad {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_bajo[] = { -12.5,0.0,25.0,37.5 };
 double _p_medio[] = { 25.0,37.5,62.5,75.0 };
 double _p_alto[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid bajo = new MF_xfl_trapezoid(min,max,step,_p_bajo);
 MF_xfl_trapezoid medio = new MF_xfl_trapezoid(min,max,step,_p_medio);
 MF_xfl_trapezoid alto = new MF_xfl_trapezoid(min,max,step,_p_alto);
}

//+------------------------------------------------------+//
//  Type TP_cambiosSO  //
//+------------------------------------------------------+//

private class TP_cambiosSO {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_bajo[] = { -12.5,0.0,25.0,37.5 };
 double _p_medio[] = { 25.0,37.5,62.5,75.0 };
 double _p_alto[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid bajo = new MF_xfl_trapezoid(min,max,step,_p_bajo);
 MF_xfl_trapezoid medio = new MF_xfl_trapezoid(min,max,step,_p_medio);
 MF_xfl_trapezoid alto = new MF_xfl_trapezoid(min,max,step,_p_alto);
}

//+------------------------------------------------------+//
//  Type TP_copiaredinterna  //
//+------------------------------------------------------+//

private class TP_copiaredinterna {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_bajo[] = { -12.5,0.0,25.0,37.5 };
 double _p_medio[] = { 25.0,37.5,62.5,75.0 };
 double _p_alto[] = { 62.5,75.0,100.0,112.5 };
```

```
 MF_xfl_trapezoid bajo = new MF_xfl_trapezoid(min,max,step,_p_bajo);
 MF_xfl_trapezoid medio = new MF_xfl_trapezoid(min,max,step,_p_medio);
 MF_xfl_trapezoid alto = new MF_xfl_trapezoid(min,max,step,_p_alto);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Type TP_estadointerno  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private class TP_estadointerno {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_malo[] = { -12.5,0.0,25.0,37.5 };
 double _p_normal[] = { 25.0,37.5,62.5,75.0 };
 double _p_bueno[] = { 62.5,75.0,100.0,112.5 };
 MF_xfl_trapezoid malo = new MF_xfl_trapezoid(min,max,step,_p_malo);
 MF_xfl_trapezoid normal = new MF_xfl_trapezoid(min,max,step,_p_normal);
 MF_xfl_trapezoid bueno = new MF_xfl_trapezoid(min,max,step,_p_bueno);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Type TP_binario  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private class TP_binario {
 private double min = 0.0;
 private double max = 1.0;
 private double step = 1.0;
 double _p_no[] = { 0.0 };
 double _p_si[] = { 1.0 };
 MF_xfl_singleton no = new MF_xfl_singleton(min,max,step,_p_no);
 MF_xfl_singleton si = new MF_xfl_singleton(min,max,step,_p_si);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Type TP_ProcEnEjec  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private class TP_ProcEnEjec {
 private double min = 0.0;
 private double max = 100.0;
 private double step = 1.0101010101010102;
 double _p_muypoco[] = { -7.142857142857143,0.0,14.285714285714286,21.42857142857143 };
 double _p_poco[] = { 14.285714285714286,21.42857142857143,35.714285714285715,42.85714285714286 };
 double _p_normal[] = { 35.714285714285715,42.85714285714286,57.142857142857146,64.28571428571429 };
 double _p_mucho[] = { 57.142857142857146,64.28571428571429,78.57142857142857,85.71428571428572 };
 double _p_demasiado[] = { 78.57142857142857,85.71428571428572,100.0,107.14285714285715 };
 MF_xfl_trapezoid muypoco = new MF_xfl_trapezoid(min,max,step,_p_muypoco);
 MF_xfl_trapezoid poco = new MF_xfl_trapezoid(min,max,step,_p_poco);
 MF_xfl_trapezoid normal = new MF_xfl_trapezoid(min,max,step,_p_normal);
 MF_xfl_trapezoid mucho = new MF_xfl_trapezoid(min,max,step,_p_mucho);
 MF_xfl_trapezoid demasiado = new MF_xfl_trapezoid(min,max,step,_p_demasiado);
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_red  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private MembershipFunction[] RL_red(MembershipFunction anchobanda, MembershipFunction numeroconexiones) {
 double _rl;
 double _input[] = new double[2];
 if(anchobanda instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) anchobanda).getValue();
 if(numeroconexiones instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) numeroconexiones).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction saturacionred = new OutputMembershipFunction();
 saturacionred.set(25,_op,_input);
 TP_anchodebandacons _t_anchobanda = new TP_anchodebandacons();
 TP_NumConex _t_numeroconexiones = new TP_NumConex();
 TP_nivelsaturacionred _t_saturacionred = new TP_nivelsaturacionred();
 int _i_saturacionred=0;
 _rl = _op.and(_t_anchobanda.muypoco.isEqual(anchobanda),_t_numeroconexiones.muypocas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.muypocasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.muypoco.isEqual(anchobanda),_t_numeroconexiones.pocas.isEqual(numeroconexiones));
```

```
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.muypocasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.muypoco.isEqual(anchobanda),_t_numeroconexiones.normales.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.pocasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.muypoco.isEqual(anchobanda),_t_numeroconexiones.muchas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.normalsaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.muypoco.isEqual(anchobanda),_t_numeroconexiones.demasiadas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.demasiadasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.poco.isEqual(anchobanda),_t_numeroconexiones.muypocas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.pocasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.poco.isEqual(anchobanda),_t_numeroconexiones.pocas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.pocasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.poco.isEqual(anchobanda),_t_numeroconexiones.normales.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.normalsaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.poco.isEqual(anchobanda),_t_numeroconexiones.muchas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.muchasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.poco.isEqual(anchobanda),_t_numeroconexiones.demasiadas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.demasiadasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.normal.isEqual(anchobanda),_t_numeroconexiones.muypocas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.pocasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.normal.isEqual(anchobanda),_t_numeroconexiones.pocas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.normalsaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.normal.isEqual(anchobanda),_t_numeroconexiones.normales.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.muchasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.normal.isEqual(anchobanda),_t_numeroconexiones.muchas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.muchasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.normal.isEqual(anchobanda),_t_numeroconexiones.demasiadas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.demasiadasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.mucho.isEqual(anchobanda),_t_numeroconexiones.muypocas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.muchasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.mucho.isEqual(anchobanda),_t_numeroconexiones.pocas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.muchasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.mucho.isEqual(anchobanda),_t_numeroconexiones.normales.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.muchasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.mucho.isEqual(anchobanda),_t_numeroconexiones.muchas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.muchasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.mucho.isEqual(anchobanda),_t_numeroconexiones.demasiadas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.demasiadasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.demasiado.isEqual(anchobanda),_t_numeroconexiones.muypocas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.demasiadasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.demasiado.isEqual(anchobanda),_t_numeroconexiones.pocas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.demasiadasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.demasiado.isEqual(anchobanda),_t_numeroconexiones.normales.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.demasiadasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.demasiado.isEqual(anchobanda),_t_numeroconexiones.muchas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.demasiadasaturacion); _i_saturacionred++;
 _rl = _op.and(_t_anchobanda.demasiado.isEqual(anchobanda),_t_numeroconexiones.demasiadas.isEqual(numeroconexiones));
 saturacionred.set(_i_saturacionred,_rl, _t_saturacionred.demasiadasaturacion); _i_saturacionred++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = saturacionred;
 return _output;
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_UsoRecursos  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

 private MembershipFunction[] RL_UsoRecursos(MembershipFunction cpu, MembershipFunction ram, MembershipFunction discoduro)
{
 double _rl;
 double _input[] = new double[3];
 if(cpu instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) cpu).getValue();
 if(ram instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) ram).getValue();
 if(discoduro instanceof FuzzySingleton)
  _input[2] = ((FuzzySingleton) discoduro).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction rendimiento = new OutputMembershipFunction();
 rendimiento.set(27,_op,_input);
 TP_consumoCPU _t_cpu = new TP_consumoCPU();
 TP_consumoRAM _t_ram = new TP_consumoRAM();
 TP_actividadDiscoDuro _t_discoduro = new TP_actividadDiscoDuro();
 TP_actividadDiscoDuro _t_rendimiento = new TP_actividadDiscoDuro();
 int _i_rendimiento=0;
 _rl = _op.and(_op.and(_t_cpu.bajo.isEqual(cpu),_t_ram.bajo.isEqual(ram)),_t_discoduro.bajo.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.bajo); _i_rendimiento++;
```

```
 _rl = _op.and(_op.and(_t_cpu.bajo.isEqual(cpu),_t_ram.medio.isEqual(ram)),_t_discoduro.bajo.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.bajo); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.bajo.isEqual(cpu),_t_ram.alto.isEqual(ram)),_t_discoduro.bajo.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.bajo); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.medio.isEqual(cpu),_t_ram.bajo.isEqual(ram)),_t_discoduro.bajo.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.bajo); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.medio.isEqual(cpu),_t_ram.medio.isEqual(ram)),_t_discoduro.bajo.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.medio.isEqual(cpu),_t_ram.alto.isEqual(ram)),_t_discoduro.bajo.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.alto.isEqual(cpu),_t_ram.bajo.isEqual(ram)),_t_discoduro.bajo.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.bajo); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.alto.isEqual(cpu),_t_ram.medio.isEqual(ram)),_t_discoduro.bajo.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.alto.isEqual(cpu),_t_ram.alto.isEqual(ram)),_t_discoduro.bajo.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.alto); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.bajo.isEqual(cpu),_t_ram.bajo.isEqual(ram)),_t_discoduro.medio.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.bajo); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.bajo.isEqual(cpu),_t_ram.medio.isEqual(ram)),_t_discoduro.medio.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.bajo.isEqual(cpu),_t_ram.alto.isEqual(ram)),_t_discoduro.medio.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.medio.isEqual(cpu),_t_ram.bajo.isEqual(ram)),_t_discoduro.medio.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.medio.isEqual(cpu),_t_ram.medio.isEqual(ram)),_t_discoduro.medio.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.medio.isEqual(cpu),_t_ram.alto.isEqual(ram)),_t_discoduro.medio.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.alto.isEqual(cpu),_t_ram.bajo.isEqual(ram)),_t_discoduro.medio.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.alto.isEqual(cpu),_t_ram.medio.isEqual(ram)),_t_discoduro.medio.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.alto.isEqual(cpu),_t_ram.alto.isEqual(ram)),_t_discoduro.medio.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.alto); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.bajo.isEqual(cpu),_t_ram.bajo.isEqual(ram)),_t_discoduro.alto.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.bajo); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.bajo.isEqual(cpu),_t_ram.medio.isEqual(ram)),_t_discoduro.alto.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.bajo.isEqual(cpu),_t_ram.alto.isEqual(ram)),_t_discoduro.alto.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.alto); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.medio.isEqual(cpu),_t_ram.bajo.isEqual(ram)),_t_discoduro.alto.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.medio.isEqual(cpu),_t_ram.medio.isEqual(ram)),_t_discoduro.alto.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.medio); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.medio.isEqual(cpu),_t_ram.alto.isEqual(ram)),_t_discoduro.alto.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.alto); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.alto.isEqual(cpu),_t_ram.bajo.isEqual(ram)),_t_discoduro.alto.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.alto); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.alto.isEqual(cpu),_t_ram.medio.isEqual(ram)),_t_discoduro.alto.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.alto); _i_rendimiento++;
 _rl = _op.and(_op.and(_t_cpu.alto.isEqual(cpu),_t_ram.alto.isEqual(ram)),_t_discoduro.alto.isEqual(discoduro));
 rendimiento.set(_i_rendimiento,_rl, _t_rendimiento.alto); _i_rendimiento++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = rendimiento;
 return _output;
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_saturaciongeneral  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private MembershipFunction[] RL_saturaciongeneral(MembershipFunction red, MembershipFunction servidor) {
 double _rl;
 double _input[] = new double[2];
 if(red instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) red).getValue();
 if(servidor instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) servidor).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction saturacion = new OutputMembershipFunction();
 saturacion.set(9,_op,_input);
 TP_actividadDiscoDuro _t_red = new TP_actividadDiscoDuro();
 TP_actividadDiscoDuro _t_servidor = new TP_actividadDiscoDuro();
 TP_nivelseguridad _t_saturacion = new TP_nivelseguridad();
 int _i_saturacion=0;
 _rl = _op.and(_t_red.bajo.isEqual(red),_t_servidor.bajo.isEqual(servidor));
```

```
 saturacion.set(_i_saturacion,_rl, _t_saturacion.bajo); _i_saturacion++;
 _rl = _op.and(_t_red.bajo.isEqual(red),_t_servidor.medio.isEqual(servidor));
 saturacion.set(_i_saturacion,_rl, _t_saturacion.medio); _i_saturacion++;
 _rl = _op.and(_t_red.bajo.isEqual(red),_t_servidor.alto.isEqual(servidor));
 saturacion.set(_i_saturacion,_rl, _t_saturacion.alto); _i_saturacion++;
 _rl = _op.and(_t_red.medio.isEqual(red),_t_servidor.bajo.isEqual(servidor));
 saturacion.set(_i_saturacion,_rl, _t_saturacion.medio); _i_saturacion++;
 _rl = _op.and(_t_red.medio.isEqual(red),_t_servidor.medio.isEqual(servidor));
 saturacion.set(_i_saturacion,_rl, _t_saturacion.medio); _i_saturacion++;
 _rl = _op.and(_t_red.medio.isEqual(red),_t_servidor.alto.isEqual(servidor));
 saturacion.set(_i_saturacion,_rl, _t_saturacion.alto); _i_saturacion++;
 _rl = _op.and(_t_red.alto.isEqual(red),_t_servidor.bajo.isEqual(servidor));
 saturacion.set(_i_saturacion,_rl, _t_saturacion.alto); _i_saturacion++;
 _rl = _op.and(_t_red.alto.isEqual(red),_t_servidor.medio.isEqual(servidor));
 saturacion.set(_i_saturacion,_rl, _t_saturacion.alto); _i_saturacion++;
 _rl = _op.and(_t_red.alto.isEqual(red),_t_servidor.alto.isEqual(servidor));
 saturacion.set(_i_saturacion,_rl, _t_saturacion.alto); _i_saturacion++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = saturacion;
 return _output;
}

//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_procedenciayfin  //
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private MembershipFunction[] RL_procedenciayfin(MembershipFunction origen, MembershipFunction destino) {
 double _rl;
 double _input[] = new double[2];
 if(origen instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) origen).getValue();
 if(destino instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) destino).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction direccionamientoanormal = new OutputMembershipFunction();
 direccionamientoanormal.set(25,_op,_input);
 TP_origenAnormal _t_origen = new TP_origenAnormal();
 TP_destinoAnormal _t_destino = new TP_destinoAnormal();
 TP_destinoAnormal _t_direccionamientoanormal = new TP_destinoAnormal();
 int _i_direccionamientoanormal=0;
 _rl = _op.and(_t_origen.muypoco.isEqual(origen),_t_destino.muypoco.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.muypoco); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.muypoco.isEqual(origen),_t_destino.poco.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.poco); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.muypoco.isEqual(origen),_t_destino.normal.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.normal); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.muypoco.isEqual(origen),_t_destino.mucho.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.normal); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.muypoco.isEqual(origen),_t_destino.demasiado.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.demasiado);
_i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.poco.isEqual(origen),_t_destino.muypoco.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.poco); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.poco.isEqual(origen),_t_destino.poco.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.poco); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.poco.isEqual(origen),_t_destino.normal.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.normal); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.poco.isEqual(origen),_t_destino.mucho.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.mucho); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.poco.isEqual(origen),_t_destino.demasiado.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.demasiado);
_i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.normal.isEqual(origen),_t_destino.muypoco.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.normal); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.normal.isEqual(origen),_t_destino.poco.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.normal); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.normal.isEqual(origen),_t_destino.normal.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.normal); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.normal.isEqual(origen),_t_destino.mucho.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.normal); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.normal.isEqual(origen),_t_destino.demasiado.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.demasiado);
_i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.mucho.isEqual(origen),_t_destino.muypoco.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.normal); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.mucho.isEqual(origen),_t_destino.poco.isEqual(destino));
```

```
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.mucho); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.mucho.isEqual(origen),_t_destino.normal.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.mucho); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.mucho.isEqual(origen),_t_destino.mucho.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.mucho); _i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.mucho.isEqual(origen),_t_destino.demasiado.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.demasiado);
_i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.demasiado.isEqual(origen),_t_destino.muypoco.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.demasiado);
_i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.demasiado.isEqual(origen),_t_destino.poco.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.demasiado);
_i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.demasiado.isEqual(origen),_t_destino.normal.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.demasiado);
_i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.demasiado.isEqual(origen),_t_destino.mucho.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.demasiado);
_i_direccionamientoanormal++;
 _rl = _op.and(_t_origen.demasiado.isEqual(origen),_t_destino.demasiado.isEqual(destino));
 direccionamientoanormal.set(_i_direccionamientoanormal,_rl, _t_direccionamientoanormal.demasiado);
_i_direccionamientoanormal++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = direccionamientoanormal;
 return _output;
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_anomaliatemporal  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private MembershipFunction[] RL_anomaliatemporal(MembershipFunction duracion, MembershipFunction horarioconexion) {
 double _rl;
 double _input[] = new double[2];
 if(duracion instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) duracion).getValue();
 if(horarioconexion instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) horarioconexion).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction horarioraro = new OutputMembershipFunction();
 horarioraro.set(15,_op,_input);
 TP_tiempo _t_duracion = new TP_tiempo();
 TP_horario _t_horarioconexion = new TP_horario();
 TP_fiabilidad _t_horarioraro = new TP_fiabilidad();
 int _i_horarioraro=0;
 _rl = _op.and(_t_duracion.bajo.isEqual(duracion),_t_horarioconexion.hora0a5.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.medio); _i_horarioraro++;
 _rl = _op.and(_t_duracion.bajo.isEqual(duracion),_t_horarioconexion.hora6a11.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.bajo); _i_horarioraro++;
 _rl = _op.and(_t_duracion.bajo.isEqual(duracion),_t_horarioconexion.hora11a16.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.bajo); _i_horarioraro++;
 _rl = _op.and(_t_duracion.bajo.isEqual(duracion),_t_horarioconexion.hora16a21.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.bajo); _i_horarioraro++;
 _rl = _op.and(_t_duracion.bajo.isEqual(duracion),_t_horarioconexion.hora21a24.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.medio); _i_horarioraro++;
 _rl = _op.and(_t_duracion.medio.isEqual(duracion),_t_horarioconexion.hora0a5.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.alto); _i_horarioraro++;
 _rl = _op.and(_t_duracion.medio.isEqual(duracion),_t_horarioconexion.hora6a11.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.medio); _i_horarioraro++;
 _rl = _op.and(_t_duracion.medio.isEqual(duracion),_t_horarioconexion.hora11a16.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.bajo); _i_horarioraro++;
 _rl = _op.and(_t_duracion.medio.isEqual(duracion),_t_horarioconexion.hora16a21.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.medio); _i_horarioraro++;
 _rl = _op.and(_t_duracion.medio.isEqual(duracion),_t_horarioconexion.hora21a24.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.alto); _i_horarioraro++;
 _rl = _op.and(_t_duracion.alto.isEqual(duracion),_t_horarioconexion.hora0a5.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.alto); _i_horarioraro++;
 _rl = _op.and(_t_duracion.alto.isEqual(duracion),_t_horarioconexion.hora6a11.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.medio); _i_horarioraro++;
 _rl = _op.and(_t_duracion.alto.isEqual(duracion),_t_horarioconexion.hora11a16.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.medio); _i_horarioraro++;
 _rl = _op.and(_t_duracion.alto.isEqual(duracion),_t_horarioconexion.hora16a21.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.medio); _i_horarioraro++;
 _rl = _op.and(_t_duracion.alto.isEqual(duracion),_t_horarioconexion.hora21a24.isEqual(horarioconexion));
 horarioraro.set(_i_horarioraro,_rl, _t_horarioraro.alto); _i_horarioraro++;
```

```
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = horarioraro;
 return _output;
}

//+------------------------------------------------------+//
// Rulebase RL_nivelseguridad //
//+------------------------------------------------------+//

private MembershipFunction[] RL_nivelseguridad(MembershipFunction nivelfiab, MembershipFunction nivelsensi) {
 double _rl;
 double _input[] = new double[2];
 if(nivelfiab instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) nivelfiab).getValue();
 if(nivelsensi instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) nivelsensi).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction nivelsegur = new OutputMembershipFunction();
 nivelsegur.set(9,_op,_input);
 TP_fiabilidad _t_nivelfiab = new TP_fiabilidad();
 TP_sensibilidad _t_nivelsensi = new TP_sensibilidad();
 TP_nivelseguridad _t_nivelsegur = new TP_nivelseguridad();
 int _i_nivelsegur=0;
 _rl = _op.and(_t_nivelfiab.bajo.isEqual(nivelfiab),_t_nivelsensi.bajo.isEqual(nivelsensi));
 nivelsegur.set(_i_nivelsegur,_rl, _t_nivelsegur.bajo); _i_nivelsegur++;
 _rl = _op.and(_t_nivelfiab.bajo.isEqual(nivelfiab),_t_nivelsensi.medio.isEqual(nivelsensi));
 nivelsegur.set(_i_nivelsegur,_rl, _t_nivelsegur.bajo); _i_nivelsegur++;
 _rl = _op.and(_t_nivelfiab.bajo.isEqual(nivelfiab),_t_nivelsensi.alto.isEqual(nivelsensi));
 nivelsegur.set(_i_nivelsegur,_rl, _t_nivelsegur.bajo); _i_nivelsegur++;
 _rl = _op.and(_t_nivelfiab.medio.isEqual(nivelfiab),_t_nivelsensi.bajo.isEqual(nivelsensi));
 nivelsegur.set(_i_nivelsegur,_rl, _t_nivelsegur.bajo); _i_nivelsegur++;
 _rl = _op.and(_t_nivelfiab.medio.isEqual(nivelfiab),_t_nivelsensi.medio.isEqual(nivelsensi));
 nivelsegur.set(_i_nivelsegur,_rl, _t_nivelsegur.medio); _i_nivelsegur++;
 _rl = _op.and(_t_nivelfiab.medio.isEqual(nivelfiab),_t_nivelsensi.alto.isEqual(nivelsensi));
 nivelsegur.set(_i_nivelsegur,_rl, _t_nivelsegur.medio); _i_nivelsegur++;
 _rl = _op.and(_t_nivelfiab.alto.isEqual(nivelfiab),_t_nivelsensi.bajo.isEqual(nivelsensi));
 nivelsegur.set(_i_nivelsegur,_rl, _t_nivelsegur.bajo); _i_nivelsegur++;
 _rl = _op.and(_t_nivelfiab.alto.isEqual(nivelfiab),_t_nivelsensi.medio.isEqual(nivelsensi));
 nivelsegur.set(_i_nivelsegur,_rl, _t_nivelsegur.medio); _i_nivelsegur++;
 _rl = _op.and(_t_nivelfiab.alto.isEqual(nivelfiab),_t_nivelsensi.alto.isEqual(nivelsensi));
 nivelsegur.set(_i_nivelsegur,_rl, _t_nivelsegur.alto); _i_nivelsegur++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = nivelsegur;
 return _output;
}

//+------------------------------------------------------+//
// Rulebase RL_seguridadredinterna //
//+------------------------------------------------------+//

private MembershipFunction[] RL_seguridadredinterna(MembershipFunction modificacionesinternas, MembershipFunction
accesosinternos) {
 double _rl;
 double _input[] = new double[2];
 if(modificacionesinternas instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) modificacionesinternas).getValue();
 if(accesosinternos instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) accesosinternos).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction seginterna = new OutputMembershipFunction();
 seginterna.set(9,_op,_input);
 TP_cambiosSO _t_modificacionesinternas = new TP_cambiosSO();
 TP_copiaredinterna _t_accesosinternos = new TP_copiaredinterna();
 TP_nivelseguridad _t_seginterna = new TP_nivelseguridad();
 int _i_seginterna=0;
 _rl = _op.and(_t_modificacionesinternas.bajo.isEqual(modificacionesinternas),_t_accesosinternos.bajo.isEqual(accesosinternos));
 seginterna.set(_i_seginterna,_rl, _t_seginterna.alto); _i_seginterna++;
 _rl = _op.and(_t_modificacionesinternas.bajo.isEqual(modificacionesinternas),_t_accesosinternos.medio.isEqual(accesosinternos));
 seginterna.set(_i_seginterna,_rl, _t_seginterna.medio); _i_seginterna++;
 _rl = _op.and(_t_modificacionesinternas.bajo.isEqual(modificacionesinternas),_t_accesosinternos.alto.isEqual(accesosinternos));
 seginterna.set(_i_seginterna,_rl, _t_seginterna.bajo); _i_seginterna++;
 _rl = _op.and(_t_modificacionesinternas.medio.isEqual(modificacionesinternas),_t_accesosinternos.bajo.isEqual(accesosinternos));
 seginterna.set(_i_seginterna,_rl, _t_seginterna.medio); _i_seginterna++;
 _rl = _op.and(_t_modificacionesinternas.medio.isEqual(modificacionesinternas),_t_accesosinternos.medio.isEqual(accesosinternos));
 seginterna.set(_i_seginterna,_rl, _t_seginterna.medio); _i_seginterna++;
 _rl = _op.and(_t_modificacionesinternas.medio.isEqual(modificacionesinternas),_t_accesosinternos.alto.isEqual(accesosinternos));
```

```
 seginterna.set(_i_seginterna,_rl, _t_seginterna.bajo); _i_seginterna++;
 _rl = _op.and(_t_modificacionesinternas.alto.isEqual(modificacionesinternas),_t_accesosinternos.bajo.isEqual(accesosinternos));
 seginterna.set(_i_seginterna,_rl, _t_seginterna.bajo); _i_seginterna++;
 _rl = _op.and(_t_modificacionesinternas.alto.isEqual(modificacionesinternas),_t_accesosinternos.medio.isEqual(accesosinternos));
 seginterna.set(_i_seginterna,_rl, _t_seginterna.bajo); _i_seginterna++;
 _rl = _op.and(_t_modificacionesinternas.alto.isEqual(modificacionesinternas),_t_accesosinternos.alto.isEqual(accesosinternos));
 seginterna.set(_i_seginterna,_rl, _t_seginterna.bajo); _i_seginterna++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = seginterna;
 return _output;
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_final  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

 private MembershipFunction[] RL_final(MembershipFunction saturaciongeneral, MembershipFunction nivelseguridad,
MembershipFunction vulnerabilidad, MembershipFunction peligrosidad) {
 double _rl;
 double _input[] = new double[4];
 if(saturaciongeneral instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) saturaciongeneral).getValue();
 if(nivelseguridad instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) nivelseguridad).getValue();
 if(vulnerabilidad instanceof FuzzySingleton)
  _input[2] = ((FuzzySingleton) vulnerabilidad).getValue();
 if(peligrosidad instanceof FuzzySingleton)
  _input[3] = ((FuzzySingleton) peligrosidad).getValue();
 OP_operatorset2 _op = new OP_operatorset2();
 OutputMembershipFunction salida_anomalia = new OutputMembershipFunction();
 salida_anomalia.set(81,_op,_input);
 TP_nivelseguridad _t_saturaciongeneral = new TP_nivelseguridad();
 TP_nivelseguridad _t_nivelseguridad = new TP_nivelseguridad();
 TP_nivelseguridad _t_vulnerabilidad = new TP_nivelseguridad();
 TP_nivelseguridad _t_peligrosidad = new TP_nivelseguridad();
 TP_probabilidadanomalia _t_salida_anomalia = new TP_probabilidadanomalia();
 int _i_salida_anomalia=0;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelse
guridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
 salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelse
guridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
 salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelse
guridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
 salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelse
guridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
 salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelse
guridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
 salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelse
guridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
 salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelse
guridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
 salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelse
guridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
 salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelse
guridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
 salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
 _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
```

```
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.bajo.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
```

```
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nive
lseguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nive
lseguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nive
lseguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.alta); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nive
lseguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nive
lseguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nive
lseguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nive
lseguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nive
lseguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nive
lseguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
```

```
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.alta); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.medio.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelseg
uridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelseg
uridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelseg
uridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelseg
uridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelseg
uridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelseg
uridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelseg
uridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelseg
uridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.bajo.isEqual(saturaciongeneral)),_t_nivelseg
uridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.alta); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
```

```
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.medio.isEqual(saturaciongeneral)),_t_nivels
eguridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.alta); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.bajo.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.alta); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.medio.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.alta); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.bajo.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.poca); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.medio.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_t_peligrosidad.alto.isEqual(peligrosidad),_t_saturaciongeneral.alto.isEqual(saturaciongeneral)),_t_nivelseg
uridad.alto.isEqual(nivelseguridad)),_t_vulnerabilidad.alto.isEqual(vulnerabilidad));
  salida_anomalia.set(_i_salida_anomalia,_rl, _t_salida_anomalia.normal); _i_salida_anomalia++;
  MembershipFunction[] _output = new MembershipFunction[1];
  _output[0] = new FuzzySingleton(salida_anomalia.defuzzify());
  return _output;
 }

 //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
 //  Rulebase RL_ServiciosUsados  //
 //+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

 private MembershipFunction[] RL_ServiciosUsados(MembershipFunction telnet, MembershipFunction web, MembershipFunction ftp,
MembershipFunction email, MembershipFunction dns, MembershipFunction login) {
  double _rl;
  double _input[] = new double[6];
  if(telnet instanceof FuzzySingleton)
   _input[0] = ((FuzzySingleton) telnet).getValue();
  if(web instanceof FuzzySingleton)
   _input[1] = ((FuzzySingleton) web).getValue();
  if(ftp instanceof FuzzySingleton)
   _input[2] = ((FuzzySingleton) ftp).getValue();
```

```
  if(email instanceof FuzzySingleton)
    _input[3] = ((FuzzySingleton) email).getValue();
  if(dns instanceof FuzzySingleton)
    _input[4] = ((FuzzySingleton) dns).getValue();
  if(login instanceof FuzzySingleton)
    _input[5] = ((FuzzySingleton) login).getValue();
  OP__default_ _op = new OP__default_();
  OutputMembershipFunction nivel_inseguridad = new OutputMembershipFunction();
  nivel_inseguridad.set(59,_op,_input);
  TP_binario _t_telnet = new TP_binario();
  TP_binario _t_web = new TP_binario();
  TP_binario _t_ftp = new TP_binario();
  TP_binario _t_email = new TP_binario();
  TP_binario _t_dns = new TP_binario();
  TP_binario _t_login = new TP_binario();
  TP_nivelseguridad _t_nivel_inseguridad = new TP_nivelseguridad();
  int _i_nivel_inseguridad=0;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isEqu
al(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isEq
ual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
```

```
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
```

```
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.bajo); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isE
qual(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.no.isE
qual(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isEqu
al(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.no.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isEqu
al(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
```

```
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.medio); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.si.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isEqu
al(email)),_t_dns.si.isEqual(dns)),_t_login.no.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.no.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.no.isEqual(ftp)),_t_email.si.isEq
ual(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.no.isEq
ual(email)),_t_dns.si.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  _rl =
_op.and(_op.and(_op.and(_op.and(_op.and(_t_telnet.si.isEqual(telnet),_t_web.no.isEqual(web)),_t_ftp.si.isEqual(ftp)),_t_email.si.isEqu
al(email)),_t_dns.no.isEqual(dns)),_t_login.si.isEqual(login));
  nivel_inseguridad.set(_i_nivel_inseguridad,_rl, _t_nivel_inseguridad.alto); _i_nivel_inseguridad++;
  MembershipFunction[] _output = new MembershipFunction[1];
  _output[0] = nivel_inseguridad;
  return _output;
 }

 //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
 //  Rulebase RL_Peligrosidad  //
 //++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

 private MembershipFunction[] RL_Peligrosidad(MembershipFunction direccion, MembershipFunction anomaliatemporal) {
  double _rl;
  double _input[] = new double[2];
  if(direccion instanceof FuzzySingleton)
   _input[0] = ((FuzzySingleton) direccion).getValue();
  if(anomaliatemporal instanceof FuzzySingleton)
   _input[1] = ((FuzzySingleton) anomaliatemporal).getValue();
  OP__default_ _op = new OP__default_();
  OutputMembershipFunction peligrosidad = new OutputMembershipFunction();
  peligrosidad.set(15,_op,_input);
  TP_destinoAnormal _t_direccion = new TP_destinoAnormal();
  TP_fiabilidad _t_anomaliatemporal = new TP_fiabilidad();
  TP_nivelseguridad _t_peligrosidad = new TP_nivelseguridad();
  int _i_peligrosidad=0;
  _rl = _op.and(_t_direccion.muypoco.isEqual(direccion),_t_anomaliatemporal.bajo.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.bajo); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.muypoco.isEqual(direccion),_t_anomaliatemporal.medio.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.bajo); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.muypoco.isEqual(direccion),_t_anomaliatemporal.alto.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.bajo); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.poco.isEqual(direccion),_t_anomaliatemporal.bajo.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.bajo); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.poco.isEqual(direccion),_t_anomaliatemporal.medio.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.bajo); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.poco.isEqual(direccion),_t_anomaliatemporal.alto.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.medio); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.normal.isEqual(direccion),_t_anomaliatemporal.bajo.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.bajo); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.normal.isEqual(direccion),_t_anomaliatemporal.medio.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.medio); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.normal.isEqual(direccion),_t_anomaliatemporal.alto.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.alto); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.mucho.isEqual(direccion),_t_anomaliatemporal.bajo.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.medio); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.mucho.isEqual(direccion),_t_anomaliatemporal.medio.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.medio); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.mucho.isEqual(direccion),_t_anomaliatemporal.alto.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.alto); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.demasiado.isEqual(direccion),_t_anomaliatemporal.bajo.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.alto); _i_peligrosidad++;
  _rl = _op.and(_t_direccion.demasiado.isEqual(direccion),_t_anomaliatemporal.medio.isEqual(anomaliatemporal));
  peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.alto); _i_peligrosidad++;
```

```
 _rl = _op.and(_t_direccion.demasiado.isEqual(direccion),_t_anomaliatemporal.alto.isEqual(anomaliatemporal));
 peligrosidad.set(_i_peligrosidad,_rl, _t_peligrosidad.alto); _i_peligrosidad++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = peligrosidad;
 return _output;
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_estadointerno  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++//

private MembershipFunction[] RL_estadointerno(MembershipFunction numeroprocesos, MembershipFunction seginterna) {
 double _rl;
 double _input[] = new double[2];
 if(numeroprocesos instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) numeroprocesos).getValue();
 if(seginterna instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) seginterna).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction estado = new OutputMembershipFunction();
 estado.set(15,_op,_input);
 TP_ProcEnEjec _t_numeroprocesos = new TP_ProcEnEjec();
 TP_nivelseguridad _t_seginterna = new TP_nivelseguridad();
 TP_estadointerno _t_estado = new TP_estadointerno();
 int _i_estado=0;
 _rl = _op.and(_t_numeroprocesos.muypoco.isEqual(numeroprocesos),_t_seginterna.bajo.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.normal); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.muypoco.isEqual(numeroprocesos),_t_seginterna.medio.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.normal); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.muypoco.isEqual(numeroprocesos),_t_seginterna.alto.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.bueno); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.poco.isEqual(numeroprocesos),_t_seginterna.bajo.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.normal); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.poco.isEqual(numeroprocesos),_t_seginterna.medio.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.normal); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.poco.isEqual(numeroprocesos),_t_seginterna.alto.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.bueno); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.normal.isEqual(numeroprocesos),_t_seginterna.bajo.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.malo); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.normal.isEqual(numeroprocesos),_t_seginterna.medio.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.normal); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.normal.isEqual(numeroprocesos),_t_seginterna.alto.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.bueno); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.mucho.isEqual(numeroprocesos),_t_seginterna.bajo.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.malo); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.mucho.isEqual(numeroprocesos),_t_seginterna.medio.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.normal); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.mucho.isEqual(numeroprocesos),_t_seginterna.alto.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.normal); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.demasiado.isEqual(numeroprocesos),_t_seginterna.bajo.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.malo); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.demasiado.isEqual(numeroprocesos),_t_seginterna.medio.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.malo); _i_estado++;
 _rl = _op.and(_t_numeroprocesos.demasiado.isEqual(numeroprocesos),_t_seginterna.alto.isEqual(seginterna));
 estado.set(_i_estado,_rl, _t_estado.malo); _i_estado++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = estado;
 return _output;
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_Vulnerabilidad  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++//

private MembershipFunction[] RL_Vulnerabilidad(MembershipFunction estado, MembershipFunction nivel_inseguridad) {
 double _rl;
 double _input[] = new double[2];
 if(estado instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) estado).getValue();
 if(nivel_inseguridad instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) nivel_inseguridad).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction vulnerabilidad = new OutputMembershipFunction();
 vulnerabilidad.set(9,_op,_input);
 TP_estadointerno _t_estado = new TP_estadointerno();
 TP_nivelseguridad _t_nivel_inseguridad = new TP_nivelseguridad();
```

```
 TP_nivelseguridad _t_vulnerabilidad = new TP_nivelseguridad();
 int _i_vulnerabilidad=0;
 _rl = _op.and(_t_estado.malo.isEqual(estado),_t_nivel_inseguridad.bajo.isEqual(nivel_inseguridad));
 vulnerabilidad.set(_i_vulnerabilidad,_rl, _t_vulnerabilidad.medio); _i_vulnerabilidad++;
 _rl = _op.and(_t_estado.malo.isEqual(estado),_t_nivel_inseguridad.medio.isEqual(nivel_inseguridad));
 vulnerabilidad.set(_i_vulnerabilidad,_rl, _t_vulnerabilidad.medio); _i_vulnerabilidad++;
 _rl = _op.and(_t_estado.malo.isEqual(estado),_t_nivel_inseguridad.alto.isEqual(nivel_inseguridad));
 vulnerabilidad.set(_i_vulnerabilidad,_rl, _t_vulnerabilidad.alto); _i_vulnerabilidad++;
 _rl = _op.and(_t_estado.normal.isEqual(estado),_t_nivel_inseguridad.bajo.isEqual(nivel_inseguridad));
 vulnerabilidad.set(_i_vulnerabilidad,_rl, _t_vulnerabilidad.bajo); _i_vulnerabilidad++;
 _rl = _op.and(_t_estado.normal.isEqual(estado),_t_nivel_inseguridad.medio.isEqual(nivel_inseguridad));
 vulnerabilidad.set(_i_vulnerabilidad,_rl, _t_vulnerabilidad.medio); _i_vulnerabilidad++;
 _rl = _op.and(_t_estado.normal.isEqual(estado),_t_nivel_inseguridad.alto.isEqual(nivel_inseguridad));
 vulnerabilidad.set(_i_vulnerabilidad,_rl, _t_vulnerabilidad.alto); _i_vulnerabilidad++;
 _rl = _op.and(_t_estado.bueno.isEqual(estado),_t_nivel_inseguridad.bajo.isEqual(nivel_inseguridad));
 vulnerabilidad.set(_i_vulnerabilidad,_rl, _t_vulnerabilidad.bajo); _i_vulnerabilidad++;
 _rl = _op.and(_t_estado.bueno.isEqual(estado),_t_nivel_inseguridad.medio.isEqual(nivel_inseguridad));
 vulnerabilidad.set(_i_vulnerabilidad,_rl, _t_vulnerabilidad.bajo); _i_vulnerabilidad++;
 _rl = _op.and(_t_estado.bueno.isEqual(estado),_t_nivel_inseguridad.alto.isEqual(nivel_inseguridad));
 vulnerabilidad.set(_i_vulnerabilidad,_rl, _t_vulnerabilidad.medio); _i_vulnerabilidad++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = vulnerabilidad;
 return _output;
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_moduloFalsos  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private MembershipFunction[] RL_moduloFalsos(MembershipFunction nivelseguridad, MembershipFunction salida_anomalia) {
 double _rl;
 double _input[] = new double[2];
 if(nivelseguridad instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) nivelseguridad).getValue();
 if(salida_anomalia instanceof FuzzySingleton)
  _input[1] = ((FuzzySingleton) salida_anomalia).getValue();
 OP__default_ _op = new OP__default_();
 OutputMembershipFunction salida_falsos = new OutputMembershipFunction();
 salida_falsos.set(9,_op,_input);
 TP_nivelseguridad _t_nivelseguridad = new TP_nivelseguridad();
 TP_probabilidadanomalia _t_salida_anomalia = new TP_probabilidadanomalia();
 TP_probabilidadfalsos _t_salida_falsos = new TP_probabilidadfalsos();
 int _i_salida_falsos=0;
 _rl = _op.and(_t_nivelseguridad.bajo.isEqual(nivelseguridad),_t_salida_anomalia.poca.isEqual(salida_anomalia));
 salida_falsos.set(_i_salida_falsos,_rl, _t_salida_falsos.poca); _i_salida_falsos++;
 _rl = _op.and(_t_nivelseguridad.bajo.isEqual(nivelseguridad),_t_salida_anomalia.normal.isEqual(salida_anomalia));
 salida_falsos.set(_i_salida_falsos,_rl, _t_salida_falsos.poca); _i_salida_falsos++;
 _rl = _op.and(_t_nivelseguridad.bajo.isEqual(nivelseguridad),_t_salida_anomalia.alta.isEqual(salida_anomalia));
 salida_falsos.set(_i_salida_falsos,_rl, _t_salida_falsos.normal); _i_salida_falsos++;
 _rl = _op.and(_t_nivelseguridad.medio.isEqual(nivelseguridad),_t_salida_anomalia.poca.isEqual(salida_anomalia));
 salida_falsos.set(_i_salida_falsos,_rl, _t_salida_falsos.poca); _i_salida_falsos++;
 _rl = _op.and(_t_nivelseguridad.medio.isEqual(nivelseguridad),_t_salida_anomalia.normal.isEqual(salida_anomalia));
 salida_falsos.set(_i_salida_falsos,_rl, _t_salida_falsos.normal); _i_salida_falsos++;
 _rl = _op.and(_t_nivelseguridad.medio.isEqual(nivelseguridad),_t_salida_anomalia.alta.isEqual(salida_anomalia));
 salida_falsos.set(_i_salida_falsos,_rl, _t_salida_falsos.alta); _i_salida_falsos++;
 _rl = _op.and(_t_nivelseguridad.alto.isEqual(nivelseguridad),_t_salida_anomalia.poca.isEqual(salida_anomalia));
 salida_falsos.set(_i_salida_falsos,_rl, _t_salida_falsos.normal); _i_salida_falsos++;
 _rl = _op.and(_t_nivelseguridad.alto.isEqual(nivelseguridad),_t_salida_anomalia.normal.isEqual(salida_anomalia));
 salida_falsos.set(_i_salida_falsos,_rl, _t_salida_falsos.alta); _i_salida_falsos++;
 _rl = _op.and(_t_nivelseguridad.alto.isEqual(nivelseguridad),_t_salida_anomalia.alta.isEqual(salida_anomalia));
 salida_falsos.set(_i_salida_falsos,_rl, _t_salida_falsos.alta); _i_salida_falsos++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = salida_falsos;
 return _output;
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//  Rulebase RL_nada  //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

private MembershipFunction[] RL_nada(MembershipFunction salida_anomalia) {
 double _rl;
 double _input[] = new double[1];
 if(salida_anomalia instanceof FuzzySingleton)
  _input[0] = ((FuzzySingleton) salida_anomalia).getValue();
 OP__default_ _op = new OP__default_();
```

```
 OutputMembershipFunction salida = new OutputMembershipFunction();
 salida.set(3,_op,_input);
 TP_probabilidadanomalia _t_salida_anomalia = new TP_probabilidadanomalia();
 TP_probabilidadanomalia _t_salida = new TP_probabilidadanomalia();
 int _i_salida=0;
 _rl = _t_salida_anomalia.poca.isEqual(salida_anomalia);
 salida.set(_i_salida,_rl, _t_salida.poca); _i_salida++;
 _rl = _t_salida_anomalia.normal.isEqual(salida_anomalia);
 salida.set(_i_salida,_rl, _t_salida.normal); _i_salida++;
 _rl = _t_salida_anomalia.alta.isEqual(salida_anomalia);
 salida.set(_i_salida,_rl, _t_salida.alta); _i_salida++;
 MembershipFunction[] _output = new MembershipFunction[1];
 _output[0] = salida;
 return _output;
}

//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
//             Fuzzy Inference Engine               //
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//

public double[] crispInference(double[] _input) {
 MembershipFunction saturacionCPU = new FuzzySingleton(_input[0]);
 MembershipFunction saturacionRAM = new FuzzySingleton(_input[1]);
 MembershipFunction saturacionDD = new FuzzySingleton(_input[2]);
 MembershipFunction numeroconexiones = new FuzzySingleton(_input[3]);
 MembershipFunction anchodebandaconsumido = new FuzzySingleton(_input[4]);
 MembershipFunction tiempoconectado = new FuzzySingleton(_input[5]);
 MembershipFunction horadeconexion = new FuzzySingleton(_input[6]);
 MembershipFunction origen = new FuzzySingleton(_input[7]);
 MembershipFunction destino = new FuzzySingleton(_input[8]);
 MembershipFunction nivelfiabilidad = new FuzzySingleton(_input[9]);
 MembershipFunction nivelsensibilidad = new FuzzySingleton(_input[10]);
 MembershipFunction modificacionesenSO = new FuzzySingleton(_input[11]);
 MembershipFunction copiaficheros = new FuzzySingleton(_input[12]);
 MembershipFunction numprocesos = new FuzzySingleton(_input[13]);
 MembershipFunction telnet = new FuzzySingleton(_input[14]);
 MembershipFunction web = new FuzzySingleton(_input[15]);
 MembershipFunction ftp = new FuzzySingleton(_input[16]);
 MembershipFunction email = new FuzzySingleton(_input[17]);
 MembershipFunction dns = new FuzzySingleton(_input[18]);
 MembershipFunction login = new FuzzySingleton(_input[19]);
 MembershipFunction salida_anomalia;
 MembershipFunction salida_falsos;
 MembershipFunction i0;
 MembershipFunction i1;
 MembershipFunction a;
 MembershipFunction b;
 MembershipFunction c;
 MembershipFunction i3;
 MembershipFunction e;
 MembershipFunction h;
 MembershipFunction f;
 MembershipFunction g;
 MembershipFunction i;
 MembershipFunction i4;
 MembershipFunction[] _call;
 _call = RL_UsoRecursos(saturacionCPU,saturacionRAM,saturacionDD); i0=_call[0];
 _call = RL_red(anchodebandaconsumido,numeroconexiones); i1=_call[0];
 _call = RL_saturaciongeneral(i1,i0); a=_call[0];
 _call = RL_procedenciayfin(origen,destino); b=_call[0];
 _call = RL_anomaliatemporal(tiempoconectado,horadeconexion); c=_call[0];
 _call = RL_nivelseguridad(nivelfiabilidad,nivelsensibilidad); i3=_call[0];
 _call = RL_seguridadredinterna(modificacionesenSO,copiaficheros); e=_call[0];
 _call = RL_estadointerno(numprocesos,e); h=_call[0];
 _call = RL_ServiciosUsados(telnet,web,ftp,email,dns,login); f=_call[0];
 _call = RL_Peligrosidad(b,c); g=_call[0];
 _call = RL_Vulnerabilidad(h,f); i=_call[0];
 _call = RL_final(a,i3,i,g); i4=_call[0];
 _call = RL_moduloFalsos(i3,i4); salida_falsos=_call[0];
 _call = RL_nada(i4); salida_anomalia=_call[0];
 double _output[] = new double[2];
 if(salida_anomalia instanceof FuzzySingleton)
  _output[0] = ((FuzzySingleton) salida_anomalia).getValue();
 else _output[0] = ((OutputMembershipFunction) salida_anomalia).defuzzify();
 if(salida_falsos instanceof FuzzySingleton)
  _output[1] = ((FuzzySingleton) salida_falsos).getValue();
```

```
 else _output[1] = ((OutputMembershipFunction) salida_falsos).defuzzify();
 return _output;
}

public double[] crispInference(MembershipFunction[] _input) {
 MembershipFunction saturacionCPU = _input[0];
 MembershipFunction saturacionRAM = _input[1];
 MembershipFunction saturacionDD = _input[2];
 MembershipFunction numeroconexiones = _input[3];
 MembershipFunction anchodebandaconsumido = _input[4];
 MembershipFunction tiempoconectado = _input[5];
 MembershipFunction horadeconexion = _input[6];
 MembershipFunction origen = _input[7];
 MembershipFunction destino = _input[8];
 MembershipFunction nivelfiabilidad = _input[9];
 MembershipFunction nivelsensibilidad = _input[10];
 MembershipFunction modificacionesenSO = _input[11];
 MembershipFunction copiaficheros = _input[12];
 MembershipFunction numprocesos = _input[13];
 MembershipFunction telnet = _input[14];
 MembershipFunction web = _input[15];
 MembershipFunction ftp = _input[16];
 MembershipFunction email = _input[17];
 MembershipFunction dns = _input[18];
 MembershipFunction login = _input[19];
 MembershipFunction salida_anomalia;
 MembershipFunction salida_falsos;
 MembershipFunction i0;
 MembershipFunction i1;
 MembershipFunction a;
 MembershipFunction b;
 MembershipFunction c;
 MembershipFunction i3;
 MembershipFunction e;
 MembershipFunction h;
 MembershipFunction f;
 MembershipFunction g;
 MembershipFunction i;
 MembershipFunction i4;
 MembershipFunction[] _call;
 _call = RL_UsoRecursos(saturacionCPU,saturacionRAM,saturacionDD); i0=_call[0];
 _call = RL_red(anchodebandaconsumido,numeroconexiones); i1=_call[0];
 _call = RL_saturaciongeneral(i1,i0); a=_call[0];
 _call = RL_procedenciayfin(origen,destino); b=_call[0];
 _call = RL_anomaliatemporal(tiempoconectado,horadeconexion); c=_call[0];
 _call = RL_nivelseguridad(nivelfiabilidad,nivelsensibilidad); i3=_call[0];
 _call = RL_seguridadredinterna(modificacionesenSO,copiaficheros); e=_call[0];
 _call = RL_estadointerno(numprocesos,e); h=_call[0];
 _call = RL_ServiciosUsados(telnet,web,ftp,email,dns,login); f=_call[0];
 _call = RL_Peligrosidad(b,c); g=_call[0];
 _call = RL_Vulnerabilidad(h,f); i=_call[0];
 _call = RL_final(a,i3,i,g); i4=_call[0];
 _call = RL_moduloFalsos(i3,i4); salida_falsos=_call[0];
 _call = RL_nada(i4); salida_anomalia=_call[0];
 double _output[] = new double[2];
 if(salida_anomalia instanceof FuzzySingleton)
  _output[0] = ((FuzzySingleton) salida_anomalia).getValue();
 else _output[0] = ((OutputMembershipFunction) salida_anomalia).defuzzify();
 if(salida_falsos instanceof FuzzySingleton)
  _output[1] = ((FuzzySingleton) salida_falsos).getValue();
 else _output[1] = ((OutputMembershipFunction) salida_falsos).defuzzify();
 return _output;
}

public MembershipFunction[] fuzzyInference(double[] _input) {
 MembershipFunction saturacionCPU = new FuzzySingleton(_input[0]);
 MembershipFunction saturacionRAM = new FuzzySingleton(_input[1]);
 MembershipFunction saturacionDD = new FuzzySingleton(_input[2]);
 MembershipFunction numeroconexiones = new FuzzySingleton(_input[3]);
 MembershipFunction anchodebandaconsumido = new FuzzySingleton(_input[4]);
 MembershipFunction tiempoconectado = new FuzzySingleton(_input[5]);
 MembershipFunction horadeconexion = new FuzzySingleton(_input[6]);
 MembershipFunction origen = new FuzzySingleton(_input[7]);
 MembershipFunction destino = new FuzzySingleton(_input[8]);
 MembershipFunction nivelfiabilidad = new FuzzySingleton(_input[9]);
 MembershipFunction nivelsensibilidad = new FuzzySingleton(_input[10]);
```

```
 MembershipFunction modificacionesenSO = new FuzzySingleton(_input[11]);
 MembershipFunction copiaficheros = new FuzzySingleton(_input[12]);
 MembershipFunction numprocesos = new FuzzySingleton(_input[13]);
 MembershipFunction telnet = new FuzzySingleton(_input[14]);
 MembershipFunction web = new FuzzySingleton(_input[15]);
 MembershipFunction ftp = new FuzzySingleton(_input[16]);
 MembershipFunction email = new FuzzySingleton(_input[17]);
 MembershipFunction dns = new FuzzySingleton(_input[18]);
 MembershipFunction login = new FuzzySingleton(_input[19]);
 MembershipFunction salida_anomalia;
 MembershipFunction salida_falsos;
 MembershipFunction i0;
 MembershipFunction i1;
 MembershipFunction a;
 MembershipFunction b;
 MembershipFunction c;
 MembershipFunction i3;
 MembershipFunction e;
 MembershipFunction h;
 MembershipFunction f;
 MembershipFunction g;
 MembershipFunction i;
 MembershipFunction i4;
 MembershipFunction[] _call;
 _call = RL_UsoRecursos(saturacionCPU,saturacionRAM,saturacionDD); i0=_call[0];
 _call = RL_red(anchodebandaconsumido,numeroconexiones); i1=_call[0];
 _call = RL_saturaciongeneral(i1,i0); a=_call[0];
 _call = RL_procedenciayfin(origen,destino); b=_call[0];
 _call = RL_anomaliatemporal(tiempoconectado,horadeconexion); c=_call[0];
 _call = RL_nivelseguridad(nivelfiabilidad,nivelsensibilidad); i3=_call[0];
 _call = RL_seguridadredinterna(modificacionesenSO,copiaficheros); e=_call[0];
 _call = RL_estadointerno(numprocesos,e); h=_call[0];
 _call = RL_ServiciosUsados(telnet,web,ftp,email,dns,login); f=_call[0];
 _call = RL_Peligrosidad(b,c); g=_call[0];
 _call = RL_Vulnerabilidad(h,f); i=_call[0];
 _call = RL_final(a,i3,i,g); i4=_call[0];
 _call = RL_moduloFalsos(i3,i4); salida_falsos=_call[0];
 _call = RL_nada(i4); salida_anomalia=_call[0];
 MembershipFunction _output[] = new MembershipFunction[2];
 _output[0] = salida_anomalia;
 _output[1] = salida_falsos;
 return _output;
}

public MembershipFunction[] fuzzyInference(MembershipFunction[] _input) {
 MembershipFunction saturacionCPU = _input[0];
 MembershipFunction saturacionRAM = _input[1];
 MembershipFunction saturacionDD = _input[2];
 MembershipFunction numeroconexiones = _input[3];
 MembershipFunction anchodebandaconsumido = _input[4];
 MembershipFunction tiempoconectado = _input[5];
 MembershipFunction horadeconexion = _input[6];
 MembershipFunction origen = _input[7];
 MembershipFunction destino = _input[8];
 MembershipFunction nivelfiabilidad = _input[9];
 MembershipFunction nivelsensibilidad = _input[10];
 MembershipFunction modificacionesenSO = _input[11];
 MembershipFunction copiaficheros = _input[12];
 MembershipFunction numprocesos = _input[13];
 MembershipFunction telnet = _input[14];
 MembershipFunction web = _input[15];
 MembershipFunction ftp = _input[16];
 MembershipFunction email = _input[17];
 MembershipFunction dns = _input[18];
 MembershipFunction login = _input[19];
 MembershipFunction salida_anomalia;
 MembershipFunction salida_falsos;
 MembershipFunction i0;
 MembershipFunction i1;
 MembershipFunction a;
 MembershipFunction b;
 MembershipFunction c;
 MembershipFunction i3;
 MembershipFunction e;
 MembershipFunction h;
 MembershipFunction f;
```

```
 MembershipFunction g;
 MembershipFunction i;
 MembershipFunction i4;
 MembershipFunction[] _call;
 _call = RL_UsoRecursos(saturacionCPU,saturacionRAM,saturacionDD); i0=_call[0];
 _call = RL_red(anchodebandaconsumido,numeroconexiones); i1=_call[0];
 _call = RL_saturaciongeneral(i1,i0); a=_call[0];
 _call = RL_procedenciayfin(origen,destino); b=_call[0];
 _call = RL_anomaliatemporal(tiempoconectado,horadeconexion); c=_call[0];
 _call = RL_nivelseguridad(nivelfiabilidad,nivelsensibilidad); i3=_call[0];
 _call = RL_seguridadredinterna(modificacionesenSO,copiaficheros); e=_call[0];
 _call = RL_estadointerno(numprocesos,e); h=_call[0];
 _call = RL_ServiciosUsados(telnet,web,ftp,email,dns,login); f=_call[0];
 _call = RL_Peligrosidad(b,c); g=_call[0];
 _call = RL_Vulnerabilidad(h,f); i=_call[0];
 _call = RL_final(a,i3,i,g); i4=_call[0];
 _call = RL_moduloFalsos(i3,i4); salida_falsos=_call[0];
 _call = RL_nada(i4); salida_anomalia=_call[0];
 MembershipFunction _output[] = new MembershipFunction[2];
 _output[0] = salida_anomalia;
 _output[1] = salida_falsos;
 return _output;
}

}
```

<FuzzySingleton.java>

```
public class FuzzySingleton implements MembershipFunction {
 private double value;

 public FuzzySingleton(double value) { this.value = value; }
 public double getValue() { return this.value; }
 public double compute(double x) { return (x==value? 1.0: 0.0); }
}
```

<FuzzyInferenceEngine.java>

```
public interface FuzzyInferenceEngine {
 public double[] crispInference(double[] input);
 public double[] crispInference(MembershipFunction[] input);
 public MembershipFunction[] fuzzyInference(double[] input);
 public MembershipFunction[] fuzzyInference(MembershipFunction[] input);
}
```

<MemberShipFunction.java>

```
public interface MembershipFunction {
 public double compute(double x);
}
```