DESARROLLO DE UN MODELO DE BASE DE CONOCIMIENTOS EN ASPECTOS LEGALES
CASO BANCOMER - FDR

T E S I S

MAESTRIA EN ADMINISTRACION DE SISTEMAS DE INFORMACION

INSTITUTO TECNOLOGICO Y DE ESTUDIOS

SUPERIORES DE MONTERREY

POR

ARTURO LEE AQUINO

AGOSTO DE 1995

DESARROLLO DE UN MODELO DE BASE DE CONOCIMIENTOS EN ASPECTOS LEGALES
CASO BANCOMER - FDR

TESIS

MAESTRÍA EN ADMINISTRACIÓN DE SISTEMAS DE INFORMACIÓN

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

POR

ARTURO LEE AQUINO

AGOSTO DE 1995

DESARROLLO DE UN MODELO DE BASE DE CONOCIMIENTOS EN ASPECTOS LEGALES
CASO BANCOMER - FDR

POR

ARTURO LEE AQUINO

TESIS

Presentada a la División de Graduados e Investigación
Este Trabajo es Requisito Parcial para Obtener el Título de Maestro en Administración de Sistemas de Información

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

AGOSTO DE 1995

RECONOCIMIENTOS

El autor desea expresar su más sincero agradecimiento a todas las personas que aportaron su talento, tiempo, conocimientos y paciencia para la elaboración de este trabajo y durante el transcurso del programa de maestría del cual tuve el privilegio de formar parte.

En particular quiero agradecer a las siguientes personas que tuvieron una especial relevancia en la culminación de este, tan satisfactorio, esfuerzo para completar un grado de maestría: al Ing. José Luis Figueroa, cuyas revisiones y comentarios fueron de gran ayuda; al Ing. Raúl Ceballos, por la oportunidad de realizar este proyecto; a la Dra. Ma. Elena Morín por su interés y apoyo; y al Lic. Ralf Eder por su inestimable ayuda.

"Tendemos a pensar en el conocimiento como bueno por sí mismo; pero el conocimiento sólo es útil cuando podemos explotarlo para que nos ayude a alcanzar nuestras metas"

MARVIN MINSKY

" Todo nuestro conocimiento del mundo depende de nuestra habilidad para construir modelos de él"

JOHNSON-LAIRD

A:
M.E. & I.D.E.A (ellos saben quiénes son)

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

CAPÍTULO 1. INTRODUCCIÓN

1.1 INTRODUCCIÓN

Una parte esencial de muchos sistemas que presentan comportamiento "inteligente" es la denominada "Base de Conocimientos" (*Knowledge Base - KB*). Tal componente es la encargada de determinar, por medio de una serie de reglas heurísticas, la conclusión a la que llegará un experto, (i. e.) en un Sistema Experto; la de sugerir, interpretar y ayudar a manipular los modelos de Decisiones en un *Decision Support System* (estrategias de inferencia); la de guardar la experiencia en los procesos de una compañía (memoria de la organización), etc.

Sin embargo, el implementar dicha base requiere de dos procesos fundamentales -ambos llevados a cabo por el "Ingeniero del Conocimiento" (*Knowledge Engineer*)- : la extracción del conocimiento del experto, y su implementación en un medio computacional.

El segundo proceso ha tratado de ser resuelto por medio de algunas metodologías que intentan desarrollar una estructura manejable por un sistema. Tales metodologías son: Deducción Automática, Redes Semánticas, Tripletas OAV (Objeto, Atributo, Valor), *Frames*, *Scripts* y Sistemas Basados en Reglas. Todas ellas son adecuadas según el problema que se quiera resolver, el alcance y complejidad que se le quiera dar al sistema, o la oportunidad y claridad de las respuestas (las redes neuronales requieren "aprender" el problema por medio de iteraciones, y su interpretación no es sencilla).

El propósito del documento de tesis fue el proponer un modelo de Base de Conocimientos para un Sistema Experto para Tarjeta de Crédito, basado en Reglas de Producción, utilizando un método de representación del conocimiento, que sea adecuado a la problemática.

El dominio seleccionado para la aplicación del sistema es el modelado del contrato "*DATA PROCESSING SERVICE AGREEMENT*" establecido entre First Data Resources Inc. y Bancomer S.A. Dicho documento establece la normatividad (de la relación entre las organizaciones mencionadas) para la prestación del servicio de procesamiento de tarjetas de crédito de Bancomer, así como otros servicios relacionados.

Se obtuvo, al final de este trabajo, un modelo factible de implementarse en un sistema computacional, que muestra conocimiento sobre la legalidad de los aspectos contenidos en el contrato de proceso de tarjetas; esto es: que puede establecer conclusiones sobre preguntas acerca de la validez de ciertas acciones -o su exigencia, por parte de alguna de las partes- , en relación con lo establecido en el contrato. Es decir, que es capaz de

discriminar entre una violación a lo estipulado en una cláusula o confirmar que se ha guardado la integridad de la misma.

Las características deseables del modelo para la base de conocimiento son:

- Proporcionar un formato compatible o adaptable a la computadora.
- Mantener, tan exacto como sea posible, una correspondencia entre este formato y los hechos y reglas reales implícitos en el contrato.
- Establecer una representación que pueda ser fácilmente utilizada, recuperada, modificada y actualizada.

1.2 PLANTEAMIENTO DEL PROBLEMA

Para regular la relación entre dos personas (fisicas o morales), a fin de que se cumplan las obligaciones y responsabilidades pactadas, se establece un contrato entre las mismas. En julio de 1994 Bancomer firmó un contrato con la compañía First Data Resources (FDR) para que ésta se hiciera cargo del procesamiento de sus tarjetas de crédito y productos relacionados.

Como en cualquier relación de este tipo, las condiciones son extensas y variadas, y el uso de un gran número de cláusulas en el contrato que las contemplen, se vuelve necesario. Muchas veces estas cláusulas, a fin de no ser repetitivas, hacen mención a definiciones incluidas en un glosario; y, frecuentemente, remiten al lector a condiciones o políticas incluidas en otras cláusulas. También es importante considerar el factor de interpretación de quien consulta el contrato y la cantidad de datos que hay en él. No todos entienden de igual manera lo dicho en cierta cláusula, por lo que contar con un medio que siempre ofrece la misma respuesta bajo las mismas condiciones, es de gran valía.

La mayor parte de las veces, se consulta el contrato para investigar las siguientes cuestiones:

- Establecer la validez, falta o incurrencia en violaciones; de cualquiera de las cláusulas del contrato.
- Puntualizar las acciones exigibles a cualquiera de las partes, bajo ciertas condiciones dadas.
- Verificar las condiciones y algoritmos para las tarifas de los servicios.

El uso de un sistema computarizado que permita consultar el contrato y dar opinión de una manera estándar, concisa y rápida, permitirá el tomar decisiones más acertadas con relación a los aspectos legales de este convenio. Un "Sistema Experto" es una herramienta que se adapta para la solución de este problema; sin embargo, el problema principal para el desarrollo de tal sistema, está en la elaboración de la *KB* y, más aún, en la forma de representar el conocimiento para adquirirlo, modelarlo y desarrollar la base.

1.3 OBJETIVO

El objetivo del presente trabajo es el de proponer un modelo para una base de conocimientos que represente el "Contrato de Procesamiento de Tarjetas de Crédito de Bancomer S.A. por parte de la compañía First Data Resources Inc".

1.4 ORGANIZACIÓN DE LA TESIS

El presente documento tiene el objetivo de proponer un modelo de una *KB* de un dominio particular que se establece en el inciso anterior.

Se asumen ciertos conocimientos básicos, los cuales se han omitido en el presente trabajo. Específicamente se asume cierto conocimiento de términos computacionales y de Inteligencia Artificial. La mayoría de éstos se describen y en su primera aparición se escriben tanto en inglés como en español. Posteriormente y a manera de no ser repetitivo, se hace referencia a ellos tan sólo por sus siglas. También se hace un uso somero de términos comprendidos en lenguajes como *Prolog*.

Aunque estos conocimientos no son esenciales, sí serán útiles para una mejor aproximación a los desarrollos aquí expuestos.

El trabajo está organizado de tal manera que primero se dan las bases y fundamentos de todos los elementos involucrados en el uso de sistemas expertos y los componentes asociados a los mismos, y posteriormente se progresa sistemáticamente a través de descripciones de las formas más comunes de representación del conocimiento, pasando por una breve explicación del uso de dos *shells* de características distintas, a fin de apreciar las diferencias al momento de modelar conocimiento; para entrar de lleno al desarrollo de un modelo de la *KB* del contrato.

El alcance del trabajo está delimitado y acotado a la propuesta y elaboración de un modelo de una base de conocimientos, dejando aparte el desarrollo a detalle de tal base. El enfoque se particulariza a los ejemplos más representativos de los diferentes tipos de cláusulas que se encuentran contenidas en el contrato.

La introducción proporciona el planteamiento del problema y la justificación del enfoque adoptado en el documento, así como el objetivo primordial de esta tesis. El capítulo 2 proporciona un repaso general a los conceptos de sistemas expertos, desde su definición, estructura y clasificación, hasta sus ventajas y desventajas. También son tratados brevemente los tipos de conocimiento y mecanismos de inferencia encontrados en estos sistemas. Posteriormente se pasa revista a las formas de representación del conocimiento más comunes y se especifica cuáles son las más adecuadas a nuestro problema propuesto. El capítulo termina con un resumen de los sistemas Basados en Reglas, que son los más adecuados para resolver la problemática expuesta.

El capítulo 3 nos introduce en el uso de *shells* para Sistemas Expertos y explica como éstos encauzan el análisis de adquisición de conocimiento dependiendo de la forma en que se formulan las reglas en cada uno de ellos.

El capítulo 4 es propiamente el desarrollo del modelo de la *KB*, poniendo énfasis en la categorización de conceptos, clasificación de cláusulas y formulación de reglas basadas en estructuras de datos más flexibles. Es por ello que se presentan los diagramas de árboles de decisiones y redes de inferencia, y la codificación de las reglas en el formato de ambos *shells*

descritos, para los cuatro tipos de cláusulas ejemplos, a fin de tipificar la base de conocimiento del contrato.

El capítulo 5 nos da las conclusiones del trabajo y el curso a seguir a partir de los conceptos establecidos aquí. Por último se incluyen anexos conteniendo la formulación de reglas en formato estructurado y glosario de los términos más importantes y de uso particular en esta tesis.

CAPÍTULO 2. MARCO TEÓRICO

2.1 SISTEMAS EXPERTOS

Un Sistema Experto (SE) es un programa de computadora que exhibe, dentro de un dominio específico, un grado de *expertise* al resolver un problema, comparable a un experto humano.

Podemos distinguir tres componentes estructurales básicos en un SE:

- La Base de Hechos (como se suele denominar a la base de datos), que contiene el conocimiento declarativo, a nivel de datos, sobre el problema particular que en un momento dado se intenta resolver y sobre el estado del sistema en cada instante.
- La base de conocimientos (*KB*), formada por el conocimiento específico y procedural acerca de la clase de problemas en los que el sistema es experto.
- El Motor de Inferencia (*IE*), que controla el resto del sistema en sus funciones deductivas.

La *KB* contiene una descripción, o modelo, de lo que sabemos (para llegar a la solución de un problema dado). El *IE* contiene una descripción de lo que hacemos para desarrollar la solución. Mientras que la *KB* cambia de dominio a dominio, el *IE* es el mismo si estamos utilizando un mismo *shell*.

En la Fig. 2.1 se muestra un diagrama de los componentes de un SE. Un *shell* de un SE incluye todos los componentes listados en la figura, menos la *KB*.

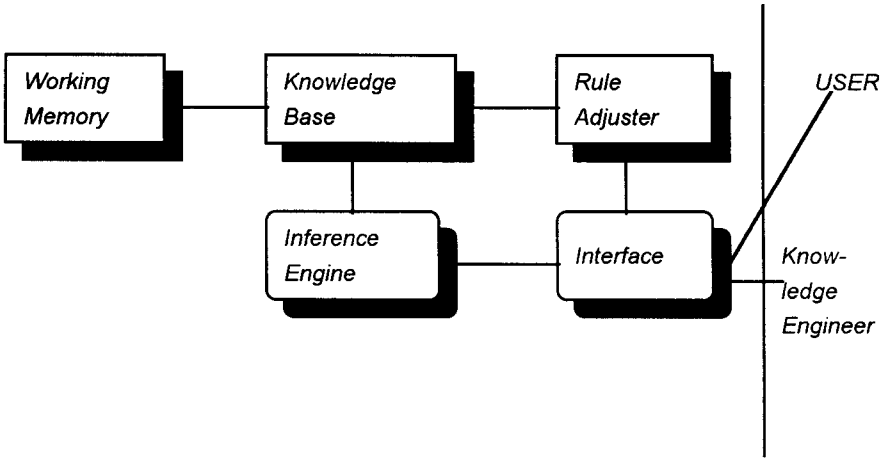


Fig. 2.1. Componentes de un sistema experto

El conocimiento que está comprendido dentro de un SE consiste de:

- Conocimiento *a priori*: Los hechos y reglas que son conocidos de un dominio antes de cualquier sesión consulta con el SE.
- Conocimiento inferido: Los hechos y reglas concernientes a un caso específico que son derivados durante y a la conclusión de una consulta con el sistema. En general, el conocimiento inferido consiste simplemente de nuevos hechos o conclusiones.

El conocimiento dentro de una *KB* es del primer tipo, esto es, hechos y reglas *a priori* de un dominio específico. El conocimiento dentro de la memoria de trabajo es dinámico, ya que cambia por cada problema planteado; y es del segundo tipo, esto es, conocimiento inferido acerca de un problema en particular bajo consideración.

2.1.1 Tipos de Aplicaciones de los Sistemas Expertos

Ten Dyke lista cuatro tipos de aplicaciones de los SE [Ignizio 91]:

Clase I: Caracterizados por la necesidad de seleccionar una solución de un conjunto de posibles alternativas casi bien definidas, tales como los problemas de diagnóstico. Esta clase coincide con los llamados: problemas de clasificación.

Clase II: Caracterizados por la necesidad de crear un plan o configuración , tales como la configuración de un sistema de cómputo y la formación de horarios y agendas. Esta clase coincide con los llamados: problemas de construcción.

Clase III: Caracterizados por la necesidad de verdadera creatividad. Tales problemas incluyen el diseño y aquellos donde la naturaleza del problema en sí, puede ser redefinida.

Clase IV: Caracterizados por aplicaciones que los humanos pueden manejar y las computadoras no. Entre esta clase de problemas se encuentran el reconocimiento de rostros, razonamiento por analogía y aprender a hablar.

Obviamente esta clasificación está basada en el grado de complejidad de los problemas planteados al SE. Entre más avanzamos en la clasificación, el SE requiere de más sofisticación para su implementación. Es por ello que la mayoría de las aplicaciones desarrolladas a la fecha involucran problemas de diagnóstico o del tipo de clasificación y diagnóstico.

En nuestro caso del contrato de FDR, la disertación sobre la validez de una acción conforme a un conjunto de condiciones es un problema típico de clasificación ya que de un conjunto de situaciones (condiciones) establece la validez o aplicación de una cláusula (conclusión). Una aplicación de la Clase II sería el pretender utilizar nuestro sistema para

que propusiera cursos de acción en base a una estrategia definida y limitada por la normatividad del contrato. Una aplicación interesante que, sin embargo, para poderla realizar, necesitaríamos contar con la experiencia de un experto humano (que no existe) que elabore planes y proyectos a partir de una estrategia apegada al contrato; y no la simple lectura y análisis de éste.

Es probable que en Bancomer exista un grupo de abogados expertos capaces de analizar y comprender todas las implicaciones e interpretaciones del contrato; pero se necesita, además, otro grupo de directivos con visión de negocio para plantear las alternativas que brinden una ventaja o una defensa, con respecto a una situación. Poniendo tal conjunto de individuos a trabajar juntos y extrayendo sus heurísticas al resolver un problema, tendríamos una aplicación de Clase II.

2.1.2 Problemas donde se aplican Sistemas Expertos

La problemática planteada en la sección 1.2, sobre la elaboración del modelo de una *KB* de un contrato, apunta a una resolución con un SE. Podríamos preguntarnos entonces si el área seleccionada en el presente trabajo, puede ser enfocada y resuelta por medio de esta tecnología. De esta manera, las preguntas a plantearse son: ¿qué tipos de problemas son susceptibles de atacarse con los SE?, ¿nuestro problema encaja en esta categoría?

Los problemas en los que se aplican SE son aquellos en los que:

1. Hay uso de criterio y no hay algoritmo.
2. Hay procesos de decisión rutinarios.
3. Existe conocimiento adquirible de un experto o de alguna otra fuente.
4. Hay información masiva.
5. Se puede liberar de trabajo a un especialista.
6. Se puede "democratizar" una pericia.
7. Hay una razonable rentabilidad.
8. Hay una problemática (sistematizable) interesante.
9. La tarea es fundamentalmente cognoscitiva.
10. No se requiere sentido común.
11. El dominio del conocimiento es valioso.
12. La tarea se resuelve en un plazo razonable.

Nuestro problema bien cumple con la mayoría de los puntos mencionados, ya que:

1. No existe un algoritmo para establecer la legalidad de una acción.
2. La consulta del contrato es un proceso rutinario y se debe revisar una gran cantidad de información.
3. En este caso, la fuente de conocimiento será el propio documento escrito del contrato.

4. La cantidad de datos e información contenida, si bien es manejable, es voluminosa y cuantiosa.
5. Por el momento no existe ningún especialista al que se pueda liberar de este trabajo debido a que las consultas las efectúa cualquier directivo interesado en algún punto.
6. No existe intención de "democratizar" la facultad de consultar al sistema.
7. La rentabilidad es alta debido a los bajos costos de desarrollo y a la gran ventaja que representa el poder evitar multas, retrasos, costos no considerados, etc., al no conocer las políticas y procedimientos establecidos en el acuerdo.
8. El tipo de aplicación, como ya dijimos, es de clasificación. La mayoría de los sistemas han sido enfocados a la identificación o verificación de una hipótesis a partir de datos preliminares. Por ejemplo: para concluir que una persona tiene gripe (hipótesis), se deben cumplir las condiciones (premisas) : hay fiebre, malestar corporal, inflamación de la laringe, etc.
   El objetivo de nuestro trabajo es la formulación de una *KB* que se obtendrá al construir (por deducción) las hipótesis de cada cláusula; y extrayendo las premisas al analizar el texto. Esto en sí mismo es original e interesante.
9. La tarea es eminentemente cognoscitiva, no existe un uso extensivo de fórmulas o algoritmos, sino más bien se hacen comparaciones y deducciones.
10. El conocimiento de sentido común y su respectivo razonamiento, están involucrados en la mayoría de las actividades inteligentes, tales como usar lenguaje natural, planeación, aprendizaje, visión de alto nivel, etc. Este es un problema muy complejo. El sentido común involucra muchos modos de razonamiento y una vasta cantidad de conocimiento con difíciles interacciones. Por ello, se ha omitido esta área en nuestro estudio, simplemente porque "no ha sido extensivamente estudiada en inteligencia artificial" [Davis, 1990]
11. El contar con conocimiento de este tipo representa una ventaja enorme para la toma de decisiones.
12. La consulta a un contrato para esclarecer una duda, es una tarea finita en tiempo, ya que no involucra problemas de explosión combinatoria. En el peor de los casos, se tomaría mucho tiempo tratando de encontrar todas las posibles anotaciones o relaciones que un conjunto de condiciones contemple. Y esto, es precisamente lo que nuestro sistema pretende ayudar a minimizar.

2.1.3 Ventajas de los Sistemas Expertos

Estas son algunas de las ventajas al utilizar SE:

- El SE está siempre disponible y al instante; y siempre ejecuta el mismo nivel de experiencia.
- Tiene acceso directo e instantáneo a las bases de datos necesarias, y no está atado a limitaciones, desviaciones e imperfecciones de los humanos.
- Es lógico, objetivo y consistente, así que no está desviado por los argumentos emocionales que pueden influenciar a un humano.
- El SE no olvida o comete errores matemáticos.
- Hace sus decisiones acorde a las metas de la compañía, en lugar de hacerlo conforme a decisiones que pudieran influenciar sus intereses personales.
- El SE multiplica el *expertise* de la compañía, esto es, está directamente disponible para todos los departamentos de la firma donde el acceso a los expertos humanos está limitado por consideraciones físicas o geográficas.
- El SE es, en sí mismo, un repositorio para almacenar el conocimiento de aquellos expertos, a partir de los cuales fue desarrollado. Es un banco de conocimientos de considerable valor y es un activo tangible y permanente en la empresa.

2.1.4 Desventajas de los Sistemas Expertos

También existen algunos puntos débiles a considerar cuando contemplamos el uso de esta tecnología. Entre las desventajas más relevantes se encuentran:

- Un experto humano tiene sentido común. Esto es, un ser humano entiende y aprecia ciertos factores culturales acerca de los cuales un SE puede, en ciertos momentos, aparecer excepcionalmente inocente.
- Los expertos humanos generalmente están conscientes del alcance y límites de su conocimiento, mientras que los SE actuales simplemente no conocen sus limitaciones. En otras palabras, ellos no saben que no saben.
- Cuando se enfrenta a una nueva situación, los expertos humanos pueden algunas veces desarrollar enfoques enteramente nuevos, brillantes y originales para resolver un problema. Específicamente,

el experto humano exhibe creatividad, mientras que un SE típico, está limitado a la *KB* desarrollada originalmente.
- El experto humano es simplemente más flexible que un programa de computadora.

2.1.5 Los Sistemas Expertos y los DSS

Actualmente, los diseñadores de *DSS* (*Decision Support Systems*) han empezado a incluir heurística - y SE - dentro de la arquitectura de los *DSS*. El sistema combinado de la Fig. 2.2 parecería proporcionar una extensión lógica, a la vez que una mejora al enfoque, ya sea del concepto de *DSS* o de SE. Esto es, permite la elección entre herramientas analíticas (modelos matemáticos y algoritmos) y heurísticas de experto (*KB* y estrategias de inferencia), dependiendo de las características del problema. De hecho, permite el uso de ambos enfoques al mismo problema - con las herramientas analíticas manejando la porción del problema atribuible a tal enfoque y el SE manejando el resto.

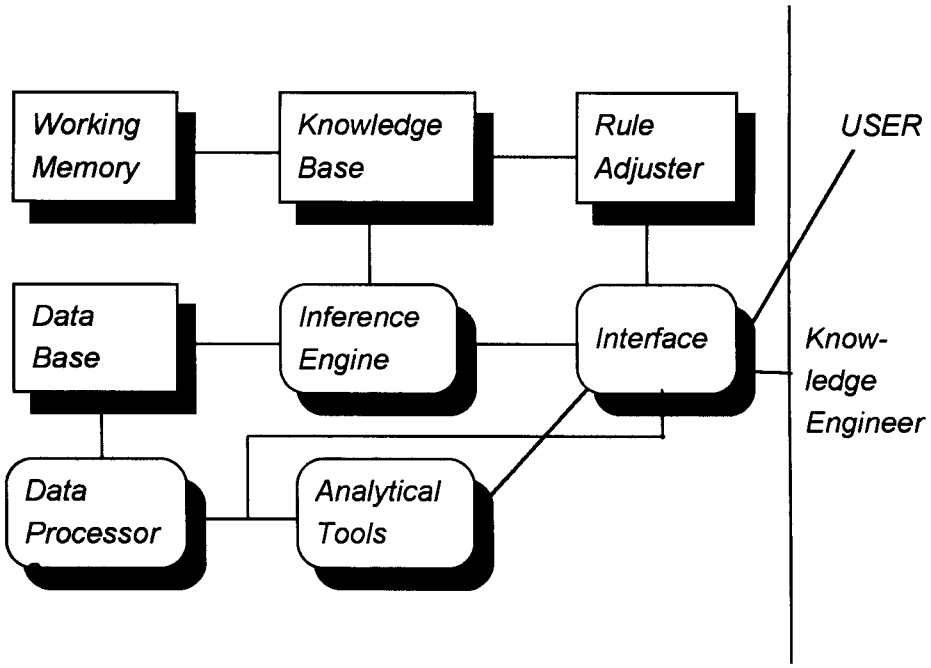


Fig. 2.2: Sistemas expertos y los *DSS*

2.2 TIPOS DE CONOCIMIENTO

Una *KB* contendrá, típicamente, dos tipos de conocimiento:

* Hechos
* Reglas

Los hechos dentro de una *KB* representan varios aspectos de un dominio específico que son sabidos antes (*prior*) de la sesión de consulta con un SE. Las Reglas dentro de la *KB* son simplemente heurísticas del tipo descrito anteriormente.

El conocimiento que se obtiene del estudio formal, o disponible en el dominio público (esto es, que se encuentra accesible en medios tales como: manuales, libros y cualquier otro medio de esta categoría - inclusive electrónico) se le denomina "Conocimiento Profundo" y forma sólo una parte de la *KB* del SE. Sin embargo, en adición a este conocimiento profundo, un experto desarrolla, a través de la experiencia, su propio conjunto de reglas heurísticas. Estas reglas heurísticas son llamadas "Conocimiento Adquirido".

El conocimiento obtenido, pues, del contrato, podemos catalogarlo como del primer tipo: "profundo", ya que se obtiene de un medio escrito accesible (en el sentido físico) a cualquier persona. No debe confundirse esto con el carácter confidencial del documento que hace que éste no sea accesible a cualquier persona, sino sólo a personal autorizado por ambas partes (Bancomer y FDR).

2.3 MECANISMOS DE INFERENCIA

Hacer inferencias es un concepto central en SE y puede significar dos cosas: usar razonamiento para encontrar cosas acerca de los hechos existentes o cosas que las reglas implican; o usar razonamiento para investigar conclusiones que son de interés y que pueden ser o no verdad. Estos usos son llamados "Encadenamiento hacia Adelante" (*Forward Chaining*) y "Encadenamiento hacia Atrás" (*Backward Chaining*) respectivamente.

2.3.1 Encadenamiento hacia Adelante

El encadenamiento hacia adelante consiste en partir de los hechos iniciales e ir aplicando las reglas hasta tratar de deducir la conclusión. La estrategia más sencilla es la de ir recorriendo las reglas desde la primera hasta llegar a una que pueda aplicarse, ampliar la base de hechos con la consecuencia, empezar de nuevo con la primera regla y así sucesivamente.

Los SE con encadenamiento hacia adelante son encontrados típicamente en actividades de planeación y diseño, donde el número de soluciones potenciales al problema es manejable; pero donde el número de datos que definen la solución inicial al problema son muchos.

2.3.2 Encadenamiento hacia Atrás

En el encadenamiento hacia atrás, generalmente se empieza con una conclusión en la que estamos interesados. Una que no es un hecho explícito, es decir, que no está en la memoria cuando iniciamos el sistema. Lo que queremos encontrar es ¿está este hecho implicado por los otros hechos y reglas que conocemos?, ¿hay algún patrón de razonamiento que podamos descubrir que establecerá este hecho como verdadero?

La forma en que un *IE* puede investigar esto es identificando todas las reglas que tienen el hecho en cuestión como conclusión. Entonces busca las premisas en todas las reglas. Si alguno de estos hechos está en la memoria de hechos verdaderos, entonces hemos determinado que el hecho bajo investigación es verdadero. Pero, si ninguna de las reglas puede ser usada directamente para determinar la conclusión bajo investigación debido a que las premisas en estas reglas también necesitan establecerse como verdaderas, entonces se debe volver hacia atrás y tratar de determinar la validez de todas las premisas en las reglas que pueden ser usadas para establecer la conclusión final.

Los SE con encadenamiento hacia atrás se encuentran frecuentemente asociados a actividades de diagnóstico o clasificación.

2.4 MODELOS DE REPRESENTACIÓN DEL CONOCIMIENTO

La Representación del Conocimiento es la ciencia de las estructuras de datos. La elección de la técnica para representar el conocimiento, condiciona lógicamente todo el diseño del sistema experto y, muy especialmente, los procedimientos de inferencia.
En general, una representación es un conjunto de convenciones sobre la forma de describir un tipo de cosas. Una vez que el problema se describe mediante una buena representación, el problema está casi resuelto.

Características deseables en las representaciones:

* Hacen explícitos los objetos y las restricciones importantes.
* Ponen de manifiesto las restricciones naturales.
* Agrupan los objetos y las relaciones.
* Suprimen los detalles insignificantes.
* Son transparentes: se entiende lo que se dice.
* Son completas y concisas.
* Se puede almacenar y recuperar la información que representan.
* Son calculables: se pueden crear mediante un procedimiento ya existente.

Una representación tiene cuatro partes fundamentales:

1. Una parte de léxico que determina qué símbolos están permitidos en el vocabulario de la representación.
2. Una parte estructural que describe las restricciones sobre la forma en que los símbolos pueden ordenarse.
3. Una parte operativa que especifica los procedimientos de acceso que permiten crear descripciones, modificarlas y responder a preguntas utilizándolas.
4. Una parte semántica que establece una forma de asociar el significado con las descripciones.

Revisaremos a continuación, muy brevemente, los diferentes modelos de representación del conocimiento:

2.4.1 Deducción Automática

En las técnicas de deducción automática se modela el conjunto de condiciones descriptivas de un problema y los mecanismos de transformación mediante fórmulas de cálculo de predicados. Un problema es resoluble si sus especificaciones son deducibles de las premisas del problema, en cuyo caso, la línea seguida en el proceso de deducciones es el método de resolución del problema. Este enfoque de la resolución inteligente de problemas exige contar con técnicas de deducción automática, por ello, una de las líneas de investigación más activas de esta etapa fue la de desarrollo de métodos apoyados en los resultados teóricos de la lógica de primer orden obtenidos hasta los años treinta. Como consecuencia de este proceso se formuló la regla universal de inferencia de resolución y unificación por Robinson en 1965, que permitía reducir el problema de deducción automática a una búsqueda de resoluciones entre cláusulas.

Sobre la idea de Robinson se realizaron aportaciones que permitieron , a principios de los setenta, desarrollar sistemas de deducción automática, que si bien, debido a la indecibilidad general de la lógica de primer orden, podrían no producir respuestas en algunos casos, en otros muchos podrían producir demostraciones de teoremas.

2.4.2 Proposiciones Lógicas

La lógica de proposiciones (o de enunciados) maneja variables proposicionales (a, b, c...) y conectivas (~a: "no a"; a^b "a y b"; a->b "si a entonces b", etc.). Unas reglas de formación de enunciados válidos definen rigurosamente el lenguaje, y se puede construir un sistema axiomático mediante la definición de unos axiomas y de unas reglas de transformación. Con ese lenguaje se pueden formalizar muchos de los procesos racionales mediante la utilización de un sistema de reglas de inferencia. Pero hay muchos otros procesos que escapan a esa posibilidad (por ejemplo, la mayoría de los silogismos clásicos).

La lógica de predicados, también llamada "Cuantificacional", introduce los elementos necesarios para tratar con razonamientos de este tipo, en los que intervienen propiedades de individuos y relaciones entre los individuos.

Una proposición es una oración que puede ser verdadera o falsa. Las proposiciones pueden ser encadenadas juntas con varios operadores (llamados conectivas lógicas) tales como *AND*, *OR*, *NOT* y *EQUIVALENT*. Las proposiciones encadenadas son llamadas compuestas.

Los elementos fundamentales del cálculo de predicados son: el objeto y el predicado. Un predicado es simplemente una frase acerca del objeto, o una relación que el objeto posee. Los predicados pueden referirse a más de un objeto y pueden ser combinados por medio del uso de conectivas lógicas.

Algunos ejemplos del uso del cálculo de predicados son:

* Mamífero(perro) - "el perro es un mamífero"
* Cuatro_Patas(perro) - "el perro tiene cuatro patas"
* Mamífero(pollo) - "el pollo es un mamífero"
* Hermana(María, Juan) - "María es hermana de Juan"

Usando el cálculo de predicados podemos representar proposiciones compuestas como "María es hermana de Juan y es prima de Pedro"

Hermana(María, Juan) *AND* Primo(María, Pedro)

También podemos representar varias relaciones, o reglas, por medio del cálculo de predicados. Por ejemplo, consideremos la heurística que establece que, si las tasas de interés suben, el precio de los bonos bajará. Usando el cálculo de predicados y la proposición lógica *OR*, podemos representar esta heurística como:

Baja(precio_bonos) *IF*
Suben(tasas_interés)

Aquí hemos usado un formato similar al empleado en *Prolog*. Una desventaja inmediata al uso de lógica en la representación de conocimiento es el hecho de que uno debe aprender algún lenguaje lógico de programación (ej. *Prolog*) para desarrollar sistemas expertos.

2.4.3 Tripletas OAV

Las Tripletas OAV (Objeto, Atributo, Valor) no sólo son una forma de representación de conocimiento en sí, sino que forman los elementos básicos de, virtualmente, cualquier otro enfoque de representación de conocimiento.

Cada tripleta OAV referencia a alguna entidad específica u objeto, y asociado con cada objeto hay un conjunto de atributos que sirven para caracterizar al objeto. En particular, los valores de una tripleta OAV pueden ser numéricas o simbólicas.

Ejemplo:

Objeto:	Avión
# de motores:	4
Tipo de motor:	Propela
Diseño de ala:	Convencional

Los atributos del ejemplo citado son: # de motores, tipo de motor, diseño de ala; y los valores de cada atributo son: 4, propela, convencional; respectivamente.

OAV se usa algunas veces para representar hechos conocidos, en lugar de una estructura lógica particular, como en una red semántica. En un SE, un programa puede simplemente agrupar información antes de almacenarla en la *KB* utilizando el formato OAV.

Las Tripletas OAV proporcionan una manera particularmente conveniente de representar ciertos hechos dentro de una *KB* y pueden ser extendidas para proporcionar los fundamentos para la representación de reglas heurísticas.

2.4.4 Redes Semánticas

Como métodos intermedios entre la búsqueda heurística y la deducción automática de resolución de problemas, aparecen los basados en "Redes Semánticas" introducidas en 1968 por Quillian para representar las relaciones, formulables en predicados, mediante grafos con arcos etiquetados con cada uno de los tipos de relación posibles. Los métodos de resolución basados en esta representación son procesos de búsqueda con restricciones asociadas a los tipos de arcos de la red.

Una red semántica puede pensarse como una red que está compuesta por múltiples tripletas OAV en forma de red, como se indica en la Fig. 2.3

Fig. 2.3 : Ejemplo de red semántica

Las redes semánticas pueden ser usadas para representar varios objetos y varios atributos por objeto. Así, el esquema de las redes semánticas proporciona un enfoque conveniente para la representación de asociaciones entre entidades. Fig. 2.4.

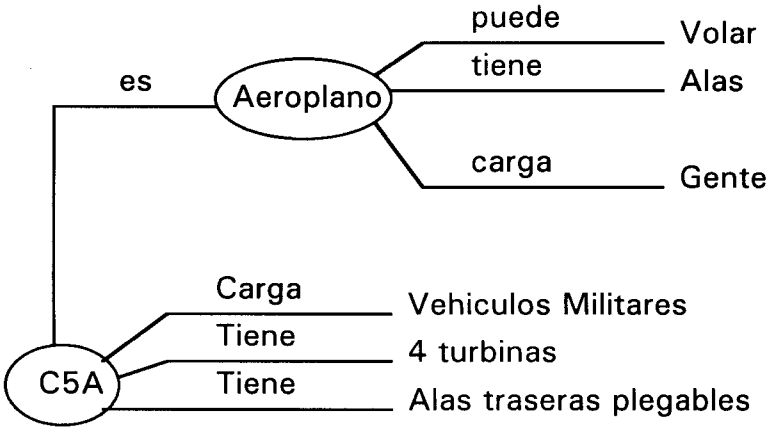


Fig. 2.4 : Red semántica con varios objetos

Debemos notar que las tripletas OAV son un subconjunto de las redes semánticas donde las únicas relaciones que pueden ser usadas son "ES" y "TIENE". Los nodos OAV de las redes semánticas, en cambio, pueden ser de tres tipos: Objetos, Atributos o Valores. Las redes semánticas son las formas más antiguas de representación de conocimiento en *AI*. Una red semántica es sólo una gráfica en la cual los nodos son objetos y descriptores, y los arcos son relaciones.

Ventajas de las Redes Semánticas

1) La estructura y enlaces de las redes semánticas pueden ser usadas para limitar el contexto en el cual las relaciones son examinadas. Esta habilidad de enfocar la atención permite a los procesos de inferencia limitar el examen de los sistemas de construcción en lugar de tener que seleccionar información relevante de un montón de relaciones.
2) Los procesos de inferencia pueden ser representados por patrones de activación dispersada de los enlaces entre los nodos de construcción. La utilización de tales métodos de inferencia puede simplificar la implementación de algoritmos paralelos para considerar conclusiones independientes.
3) La forma de la red semántica puede ser usada para guiar los procesos de inferencia para enfocarse en distinciones relevantes acordes a los enlaces activos más fuertes.
4) Las redes semánticas proporcionan la oportunidad de ejecutar razonamiento complejo acerca del conocimiento del dominio del experto, más allá del común uso de procesos de reconocimiento de patrones para hacer inferencias.
5) Los elementos básicos de las redes semánticas son los conceptos. Estos son objetos formales usados para representar objetos, atributos y relaciones del dominio de interés. Los conceptos pueden considerarse que representan el objeto intencional en lugar de una extensión de la entidad "mundo real". El número de estudios ha sido realizado para definir las estructuras de las redes a niveles detallados.

2.4.5 *Frames*

El uso de *frames* representa un enfoque alternativo que sirve para capturar la mayoría de las facilidades de una red semántica a la vez que proporciona aspectos adicionales. De hecho podemos pensar en las redes semánticas como un subconjunto del concepto de *frames*. El empleo de *frames* representa una manera particularmente robusta de presentar el conocimiento. Un *frame* contiene un objeto y compartimientos (*slots*) para cualquier información relacionada con el objeto. El contenido de tales *slots* son típicamente los atributos y los valores de los atributos, del objeto en particular. Sin embargo, en adición a almacenar valores para cada atributo, los *slots* pueden contener valores por omisión, apuntadores a otros *frames*, y conjuntos de reglas o procedimientos que pueden ser implementados.

Un *frame* podría verse así:

Frame Name : Dog
Slot : Value
Fleas : Yes
Color : Black
Name : Rover
# of Eyes : Default : 2
# of Legs : Default : 4
Owner :If needed READ License_Tag

Debemos notar que los *frames* van más allá del enfoque de las tripletas OAV de un simple valor. Con el *frame*, los valores pueden tener defaults, o los valores pueden ser (en esencia) llamadas a subrutinas. Es así porque algunos lenguajes de quinta generación como *SmallTalk* son considerados *Frame-Oriented*

La principal desventaja del uso de *frames* es, irónicamente, causada por la robustez, de tal modo de representación. Los *frames* tienen tantas capacidades que hacen su uso complejo. Jackson [1986] dice que: "mucha gente no acepta los sistemas basados en *frames* y objetos, ya que parecen apartarse de la lógica, y porque su flexibilidad en materia de contexto y control puede hacer su conducta difícil de predecir y entender". Aparte de tales desventajas, los *frames* pueden ser muy útiles, si no esenciales, en el diseño de SE de gran escala y complejos - particularmente aquellos que involucran una gran cantidad de datos *a priori* (ej. datos) y múltiples objetos.

2.4.6 *Scripts*

Los *scripts* son usados en programas de *AI* que tratan de explicar conducta. Muchos programas de *AI* leen y "entienden" nuevas historias justificando los roles de los personajes. Los *scripts* implícitamente contienen mucha información del dominio específico, haciendo entendible la tarea de entender un reporte financiero de alguna empresa.

Una preocupación común en todas estas técnicas es la de representar el conocimiento de la forma más estructurada posible, con el objetivo de facilitar su almacenamiento, modificación y búsqueda. En la mayoría de las representaciones estructuradas, se utilizan exclusivamente predicados binarios. La ventaja de utilizar solamente predicados binarios radica en la modularidad de la representación.

2.5 SISTEMAS BASADOS EN REGLAS

La forma más popular de representación de conocimiento dentro de los SE son los sistemas basados en reglas. Tales reglas son referidas como *IF-THEN*, o "Reglas de Producción". En su forma más sencilla, una regla de producción no es más que un par ordenado (A, B), que puede representarse en el lenguaje de la lógica de proposiciones como A --> B. Los elementos del par reciben distintos nombres: antecedente y consecuente, o premisa y conclusión, o condición y acción. El intérprete funciona siguiendo una regla de inferencia bien conocida en lógica, la llamada regla de Modus Ponens: del hecho A y de la regla A --> B se infiere la conclusión B.

En los sistemas basados en reglas, las reglas están arregladas en una lista. En su más simple variante, el control empieza en lo alto de la lista; las reglas son examinadas en secuencia hasta que se encuentra una para la cual la condición es verdadera. Entonces el sistema ejecuta la acción, cualesquiera que ésta sea. Posteriormente tratará de encontrar otra regla que ejecutar. Para hacer esto puede buscar empezando otra vez desde lo alto de la lista, o puede continuar buscando desde donde se quedó.

Entre las principales ventajas del uso de esta representación están:

1. La mayoría de los paquetes de desarrollo de SE existentes emplean bases de reglas.
2. Los paquetes desarrolladores de SE basados en reglas son normalmente menos caros que los que emplean modos alternativos de representación.
3. Las reglas representan un modo particularmente natural de representación del conocimiento. Consecuentemente, el tiempo requerido para aprender a desarrollar bases de reglas se minimiza.
4. Las reglas son transparentes, y ciertamente son mucho más transparentes que los modos de representación de conocimiento competidores: *Frames* y Redes Neuronales.
5. Las bases de reglas pueden ser relativamente fáciles de modificar. En particular, las adiciones, supresiones y revisiones a las bases de reglas, son un proceso sencillo.
6. Los SE basados en reglas pueden ser empleados para simular la mayoría de las características de los esquemas de representación de *frames* (esto no es necesariamente trivial).
7. La validación del contenido de un sistema basado en reglas (ej. la determinación de la integridad y consistencia de la representación) es relativamente un proceso simple.

Los sistemas basados en reglas pueden moverse hacia atrás o hacia adelante. Si el sistema funciona a partir de afirmaciones dadas hasta llegar a afirmaciones deducidas, exhibe encadenamiento hacia adelante. Si, por el contrario, el sistema forma hipótesis y usa las reglas para proceder hacia atrás en dirección a las afirmaciones que corroboran la

hipótesis, entonces muestra encadenamiento hacia atrás. ¿qué dirección es la mejor?, el problema mismo lo determina.

Siempre que las reglas sean tales que un conjunto típico de hechos pueda llevar a muchas conclusiones , el sistema exhibirá un alto grado de "amplitud de salida", y esto es un argumento para el encadenamiento hacia atrás. Por otro lado, siempre que las reglas sean tales que una hipótesis típica pueda conducir a muchas preguntas, el sistema mostrará un alto grado de "amplitud de entrada", y esto es un argumento a favor del encadenamiento hacia adelante.

CAPÍTULO 3. HERRAMIENTAS DE ADQUISICIÓN DEL CONOCIMIENTO

¿Cuál es la mejor manera de hacer un SE reusable y al mismo tiempo reducir su ciclos de desarrollo? Sería muy deseable mantener el conocimiento en un SE de tal manera que se maximice su independencia del *IE*. Si esto puede ser hecho, será posible construir nuevos SE sólo con cambiar la parte de conocimiento. El mecanismo de inferencia sería re-usado.

Tales requerimientos tuvieron eco al diseñar unos sistemas donde se puede insertar conocimiento y re-usarse la parte que no cambia. Esta parte se conoce como *shell* de SE.

Los *shells* se han convertido en los productos más populares en el mercado de *AI*. Tan pronto como la idea del *shell* fue establecida, se hizo obvio que con ciertas simples adiciones al concepto original, incrementaría su valor. La parte central de un *shell* es su mecanismo de inferencia para una clase particular de SE.

Otras características proporcionadas en estos sistemas son:

- Facilidades de explicación.
- Un editor para construir reglas.
- Facilidades de elaborar rutas para depurar reglas potenciales.
- Facilidades de interfase para proporcionar funciones tales como menú, gráficas o comunicaciones.
- Utilerías para optimizar el tiempo de corrida de los conjuntos de reglas.
- Mecanismos de ayuda en la construcción de reglas útiles que se puedan alimentar de la manera en que los expertos desean hablar sobre el tema.

La estrategia de razonamiento(*IE*) es el parámetro principal para seleccionar un *shell*. Debe ser adecuado al problema. Problemas de diagnóstico son resueltos con encadenamiento hacia atrás. Problemas de planeación y construcción con encadenamiento hacia adelante. Más adelante veremos la forma de clasificar el problema planteado en esta tesis y decidir el tipo de encadenamiento adecuado y, a su vez, el *shell* apropiado para modelarlo.

Podemos establecer los siguientes lineamientos para la selección del mecanismo de inferencia(*IE*) del *shell*.

- Si los hechos que se tiene o que se pueden establecer, conducen a un gran número de conclusiones, pero el número de formas para llegar a la conclusión particular en que estamos interesados es reducido, esto significa que hay más amplitud de salida que de entrada y por tanto se debe usar encadenamiento hacia atrás.
- Si el número de formas para llegar a la conclusión particular en la que se está interesado es grande, pero el número de conclusiones que se tiene probabilidad de obtener mediante los hechos es reducido, quiere decir que hay más amplitud de entrada que de salida y por tanto se debe usar encadenamiento hacia adelante.
- Si ninguna de las dos amplitudes domina, y aún no se han recolectado los datos, y sólo interesa saber si una de muchas conclusiones posibles es verdadera, entonces se debe usar encadenamiento hacia atrás.
- Si ya se tienen a mano todos los hechos que se pueden obtener, y deseamos saber todo lo que se pueda concluir a partir de tales hechos, se recomienda usar encadenamiento hacia adelante.

Para nuestro caso particular, hemos reducido el problema a probar una hipótesis a la vez, ya sea para comprobar si se ha incurrido en una falta (cláusulas de validez), se requiere de una acción (cláusulas de acciones exigibles), se recomienda un curso a seguir (cláusulas de procedimiento) o se establecen las características de un concepto, ya sea una situación o un ente (cláusulas de definición); por tanto se tiene un gran número de conclusiones, aunque se han dividido por cláusula del contrato, y el número de datos a manejar por cada consulta es reducido. Además, de cada cláusula sólo nos interesa saber una sola de las hipótesis (salvo en el caso de acciones exigibles, donde el número de conclusiones puede ser mayor, aunque siempre es manejable).

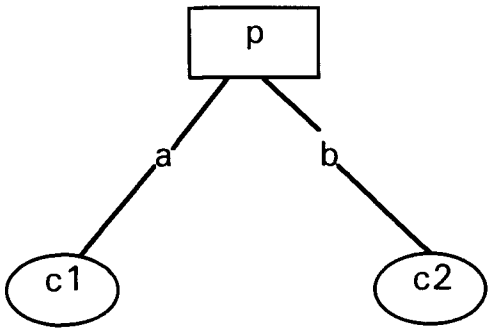
Lo recomendable en este caso es usar un *shell* que maneje encadenamiento hacia atrás, lo cual también es congruente con las características del sistema que pretendemos obtener, que es de tipo de diagnóstico, es decir, que primero establece una hipótesis y procede a probar las reglas que concluyan dicha hipótesis.

3.1 ÁRBOLES DE DECISIÓN

Una forma de representación gráfica y simple es la de los árboles de decisión. Éstos proporcionan una alternativa para la representación de las bases de reglas. Cada nodo del árbol representa una pregunta acerca del valor de un atributo o una conclusión. Cada rama que sale de un nodo-pregunta, representa uno de los posibles valores del atributo asociado. Los nodos-pregunta se representan por cuadros, mientras que las conclusiones se representan por círculos.

Fig. 3.1 Implicación Simple

IF p = a *THEN*
c1
ELSE
c2



Se transforma en:
IF p = a *THEN* c1
IF p = b *THEN* c2

Fig. 3.2 Implicación Disyuntiva

IF p = a *OR* p = b *THEN* c

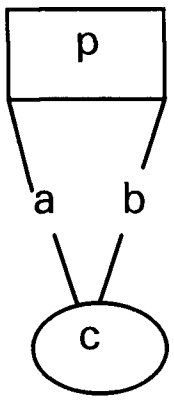


Fig. 3.3 Implicación Conjuntiva

IF p1= a *AND* p2 = b *THEN* e

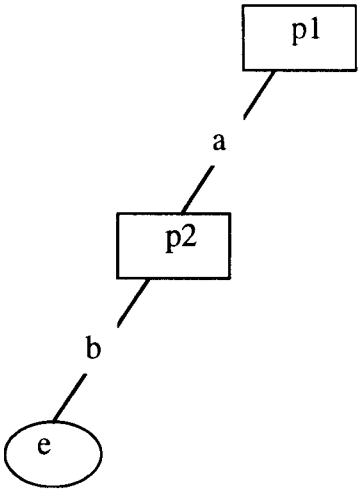
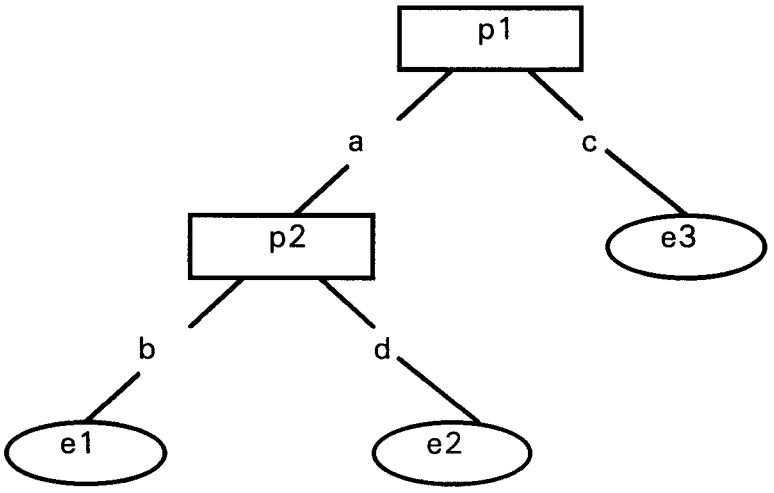


Fig. 3.4 Ejemplo de un conjunto de reglas:

IF p1 = c *THEN* e3
IF p1 = a *AND* p2 = b *THEN* e1
IF p1 = a *AND* p2 = d *THEN* e2



Para generar reglas a partir de los árboles de decisión se procede de la siguiente manera:

1. Identifique una conclusión que no haya sido considerada
2. Siga una cadena del nodo conclusión hacia atrás a través del árbol (hacia arriba) hasta el nodo raíz (el nodo en el nivel más alto del árbol).
3. En la cadena, los nodos con círculo son considerados nodos *THEN*, o cláusulas de conclusión, mientras que los nodos con cuadros son considerados nodos *IF* o cláusulas premisas.
4. Construya la regla de producción correspondiente para la cadena bajo consideración y repita este proceso para cada nodo conclusión.

Una cadena se define como la trayectoria de un nodo en el árbol a otro nodo, donde se atraviesan ramas en una sola dirección.

3.1.1 Modelo EXSYS

EXSYS es un medio ambiente para desarrollar SE basados en reglas. Las reglas se expresan en el formato *IF-THEN-ELSE*. Si todas las proposiciones de la parte *IF* de la regla son verdaderas, la regla se considera verdadera y es adicionada a lo que el programa conoce. Si cualquiera de las proposiciones de la parte *IF* es falsa, la regla se considera falsa y las proposiciones en la parte *ELSE* se vuelven verdaderas y, nuevamente, son sumadas a la memoria temporal de lo que el programa sabe. Esto quiere decir que cuando aparecen más de una premisa en cualquiera de las partes de la regla (*IF*, *THEN* o *ELSE*), éstas están relacionadas por conectivas *AND* implícitas. No existen operadores *OR* o *NOT*, así que éstos deben derivarse por medio de reglas adicionales.

Las partes *THEN* y *ELSE* pueden también contener posibles soluciones de las que EXSYS está tratando de encontrar. Estas soluciones son indicadas por una proposición seguida de la palabra "*-Probability*" y, a continuación, una razón que indica la probabilidad de la solución. EXSYS también tiene la posibilidad de instanciar por medio de asignaciones discretas 1, 0; o Verdadero y Falso.

El *shell* trabaja de manera que siempre intenta derivar información de otras reglas por medio de encadenamiento hacia atrás. También tiene facilidad de proporcionar explicación al usuario al teclear "*WHY*" en el momento que el sistema está solicitando información. Esto hará que se desplieguen las reglas asociadas a lo que está cuestionando, mostrando cuál es la inferencia que trata de aplicar.

3.2 REDES DE INFERENCIA

Otro enfoque gráfico para la representación de reglas son las llamadas Redes de Inferencia. Este enfoque requiere también de cierta nomenclatura para su utilización. Un cuadro representa una afirmación, esto es, la asociación del atributo de una premisa con un valor específico. Los círculos serán usados para representar conclusiones. Un círculo encerrado en un cuadro representa una conclusión intermedia. Finalmente las afirmaciones y las conclusiones intermedias pueden ser combinadas por conectivas lógicas (específicamente operadores *AND* y *OR*). Ver fig. 3.5

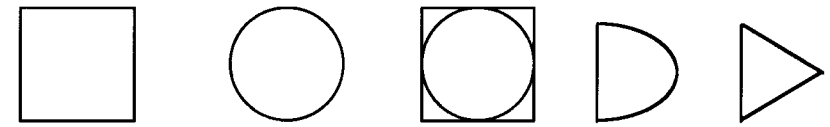


Fig 3.5 Símbolos de las redes de inferencia

3.2.1 Modelo *Prolog* (IMP *SHELL*)

Prolog por sí mismo es un *shell*, pero no muy flexible. Proporciona un mecanismo de inferencia y un medio ambiente de desarrollo competente. El conocimiento es mantenido aparte del mecanismo de inferencia. Todo el conocimiento está en forma de hechos y reglas *IF-THEN*. Su método de inferencia (*backward chaining*) está interconstruido en el sistema. A pesar de estas características, la mayoría considera a "*Prolog*" como un lenguaje de programación más que como un *shell*, así que se han construido (usando este lenguaje) algunos *shells* para mostrar la superioridad de este enfoque sobre el tradicional de programación, sea éste procedural o lógico.

IMP *Shell*

Este *shell* de SE se presenta para ejemplificar la forma en que se ha resuelto el problema de la adquisición de conocimiento, utilizando *Prolog* como herramienta fundamental. El *shell* en cuestión cuenta con algunas de las características deseables en esta clase de sistemas, sin embargo tiene algunas limitantes fuertes, de las que ya hablaremos más adelante en el comparativo de representaciones.

Las reglas se formulan por medio de un predicado especial denominado "IMP" (implicación). Ejemplo:

IMP(i, t, c, s1, e1, s2, e2, n) donde:

i =	tipo de implicación	a - *AND* o - *OR* s - *single* f - fórmula
t =	categoría	r - reversible n - no reversible
c =	conclusión	
s1 =	signo de la primera expresión lógica	
e1 =	primera expresión lógica	
s2 =	signo de la segunda expresión lógica	
e2 =	segunda expresión lógica	
n =	certidumbre de la implicación	

El primer argumento muestra el tipo de implicación (*AND*, *OR*, simple, fórmula). El segundo argumento indica si la regla es reversible (r) o no lo es (n). La conclusión a la que se llega es (c). Las dos piezas de evidencia son e1 y e2. Antes de cada evidencia se pone la palabra "neg" o "pos". Esto se usa para indicar el patrón de negación de la regla (si lo hay). La certidumbre de la implicación va al final. Esta sigue el modelo de cálculo similar a EMYCIN. En EMYCIN se usa un número en el rango -1, 0, +1 para cuantificar la certeza; así que, realmente, no se trata de una probabilidad de una evidencia. Los puntos extremos de este rango se definen así: +1 significa que el sistema está absolutamente seguro de que algo es cierto. -1 significa que cualquiera que sea la evidencia o conclusión probada, el sistema sabe que ésta es absolutamente falsa. El 0 significa que no hay conocimiento del punto en cuestión. Los números intermedios representan varios grados de credibilidad o desaprobación. Todo esto se conoce como "Certidumbre Bipolar".

Ahora explicaremos el mecanismo de reversibilidad de este sistema. Consideremos la aplicación de las siguientes reglas:

Regla 1:

IF es un carro último modelo *THEN*
tendrá convertidor catalítico

Supongamos que la certidumbre de la implicación es 0.9, lo cual significa una alta seguridad de que un auto nuevo tenga convertidor catalítico debido a los estándares impuestos. Si aplicamos esta regla a un auto antiguo (digamos que sabemos que es nuevo con una certeza de -0.8) entonces la certidumbre de la conclusión es -0.72, lo que significa

que casi estamos seguros de que el auto no tiene convertidor catalítico, y ésta sería una aplicación apropiada de la regla a un auto antiguo.

Regla 2:

IF es un carro último modelo *THEN*
tendrá estéreo

Asumiremos las mismas certidumbres que antes. Si aplicamos la regla a un auto antiguo concluiríamos que no tiene estéreo, y esta conclusión sería errónea. Esta regla sólo habla de autos último modelo; pero un SE simple, trataría de aplicarla en otras circunstancias ¿Cuál es la diferencia entre estas dos reglas? La primera regla retiene el sentido si la premisa y la conclusión son negadas; por ejemplo:

IF NOT es un carro último modelo *THEN*
NOT tendrá convertidor catalítico

La segunda regla no puede ser manipulada de esta forma. No parece razonable cuando aplicamos la negación a la premisa y conclusión.

IF NOT es un carro último modelo *THEN*
NOT tendrá estéreo

Todas las reglas a introducir en este *shell* deben caer en estas dos categorías, y las categorías son importantes. Las reglas de la primera clase se llaman reversibles. Una de sus características es que permanece aplicable para cualquier valor de certidumbre asociado con la premisa. La segunda clase de reglas son denominadas no reversibles. Una regla como ésta sólo es válida cuando la premisa tiene un valor de certidumbre positivo. Si la premisa tiene un valor negativo, la regla no debe ser usada.

Al diseñar las reglas uno puede probar si es reversible o no, negando de la manera vista, y ver si la regla aún tiene sentido.

El *shell* exige que se cumplan ciertas reglas declaradas en las siguientes restricciones:

Restricción 1

Las reglas deben tener una de estas tres formas:

Forma 1

IF (e) *THEN* (c) ct (n)
IMP(s, r, c, pos, e, *dummy*, *dummy*, n)
IF (*not* e) *THEN* (c) ct (n)
IMP(s, r ,c , *neg*, e, *dummy*, *dummy*, n)

Forma 2

IF (e1 *OR* e2) *THEN* (c) ct (n)
IMP(o, r, c, pos, e1, pos, e2, n)
IF ((*not* e1) *OR* e2) *THEN* (c) ct(n)
IMP(o, r, c, *neg*, e1, pos, e2, n)
IF (e1 *OR* (*not* e2)) *THEN* (c) ct(n)
IMP(o, r, c, pos, e1, *neg*, e2, n)
IF ((*not* e1) *OR* (*not* e2)) *THEN* (c) ct(n)
IMP(o, r, c, *neg*, e1, *neg*, e2, n)

Forma 3

IF (e1 *AND* e2) *THEN* (c) ct(n)
IMP(a, r, c, pos, e1, pos, e2, n)
IF (e1 *AND* (*not* e2)) *THEN* (c) ct(n)
IMP(a, r, c, pos, e1, *neg*, e2, n)
IF ((*not* e1) *AND* e2) *THEN* (c) ct(n)
IMP(a, r, c, *neg*, e1, pos, e2, n)
IF ((*not* e1) *AND* (*not* e2)) *THEN* (c) ct(n)
IMP(a, r, c, *neg*, e1, *neg*, e2, n)

Restricción 2

No se permiten más de 2 reglas que produzcan una conclusión particular.

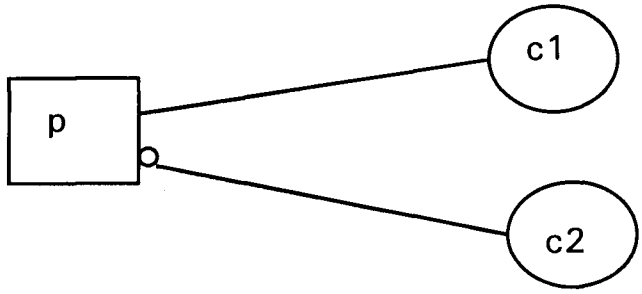
Restricción 3

La introducción de datos (estado inicial) y búsqueda de conclusiones deberá especificarse por medio de nodos especiales.

hypothesis_node ( ) - conclusiones
terminal_node ( ) - datos

Fig. 3.6 Implicación Simple

IF p *THEN* c1
ELSE c2



Se transforma en:

IF p *THEN* c1
IF NOT p *THEN* c2

Fig. 3.7 Implicación Disyuntiva

IF p *OR* q *THEN* c

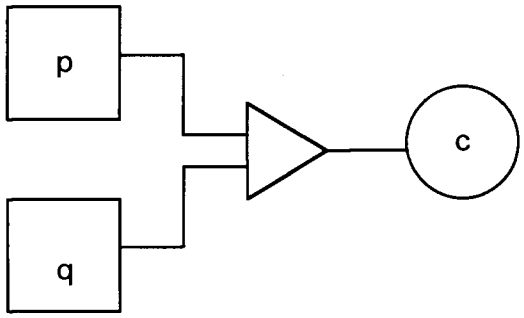


Fig. 3.8 Implicación Conjuntiva

IF p *AND* q *THEN* c

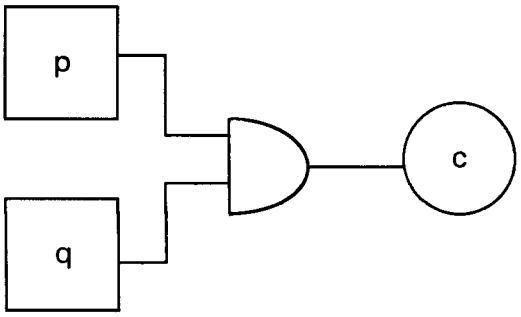
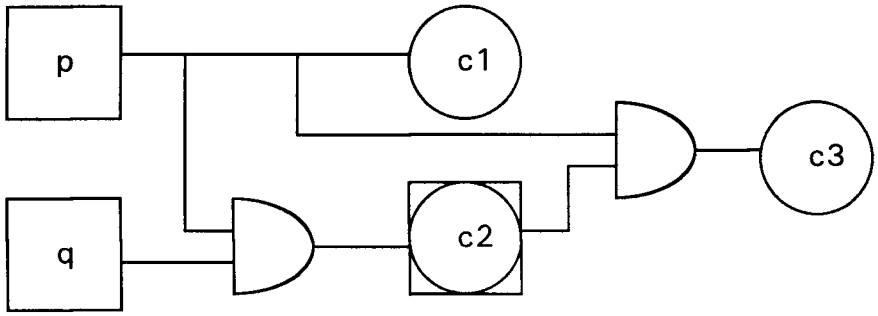


Fig. 3.9 Ejemplo de un conjunto de reglas:

IF p *THEN* c1
IF p *AND* q *THEN* c2
IF p *AND* c2 *THEN* c3



Al observar los diagramas de árboles de decisión y las redes de inferencia, inmediatamente notamos las diferencias entre ellos. Mientras que uno sigue la forma tradicional de asignar valores a los atributos (ej. *IF* p = a), las redes de inferencia prueban instancias de las premisas (ej. *IF* p). Estas diferencias se harán más evidentes al momento de formular las reglas de algunos ejemplos de cláusulas.

CAPÍTULO 4. BASE DE CONOCIMIENTO DEL CONTRATO

4.1 ESTRUCTURA DEL CONTRATO

A continuación se detallan los principales artículos de que se compone el contrato a fin de poder tipificar las cláusulas. Esta clasificación sirve para poder desarrollar con mayor facilidad, un modelo de la *KB*. Se ha mantenido su texto en inglés para evitar errores en la interpretación.

Estructura del Contrato:

Artículos

1. *Definitions*
2. *DP Services*
3. *Exclusivity*
4. *Payment for DP Services*
5. *Indemnification*
6. *Remedies; Limitation of Liability*
7. *Disclaimer of Warranties*
8. *Term of Agreement*
9. *Termination*
10. *Confidential Nature of Data*
11. *Representations and Warranties*
12. *Taxes*
13. *Dispute Resolution*
14. *General*

Exhibits

A. Descripción de los servicios
B. Programa de cargos y comisiones
   I. Cargos de Conversión
   II. Conceptos de cobro
   III. Gastos reembolsables
C. Plan de conversión
D. Estándares de *Performance*
E. Acuerdo de filiación
F1. Capacitación de conversión
F2. Capacitación de Conversión

4.2 CLASIFICACIÓN DE LAS CLÁUSULAS

Para mayor facilidad al momento de elaborar las reglas, se propone la siguiente clasificación de las cláusulas:

A).- de Definición
B).- de Validez
C).- de Acciones Exigibles
D).- de Procedimiento
E).- Combinadas

4.2.1 Descripción de los tipos de Cláusulas

A) Cláusulas de Definición

Propiamente son declaraciones que definen o especifican una cosa. Gran parte de las definiciones del contrato se localizan en el Artículo 1; aunque también se les puede encontrar a lo largo de todo el texto, como en el caso de la cláusula 4.1.b) que "define" la fórmula para calcular el pago mensual por los servicios prestados; y prácticamente todo el Artículo 11. Podemos considerar a las cláusulas de definición como las pautas para encontrar los objetos que están incluidos en el contrato o sus atributos. Atendiendo al texto de estas cláusulas, deberíamos ser capaces de configurar nuestros objetos del modelo, aunque en muchos casos esto no sea sencillo.

B) Cláusulas de Validez

Son estipulaciones que especifican las reglas o normas que se deben seguir y bajo qué condiciones se está faltando a lo acordado en dicha cláusula. Ejemplos de este tipo son las cláusulas 3 "*Exclusivity*", casi todo el Artículo 5 "*Indemnification*" y la 14.1 "*Assignment*"

C) Cláusulas de Acciones Exigibles o Acciones a Ejecutar

Especifican productos o acciones a requisitar cuando se ha cumplido cierta condición o el término de un periodo acordado; esto es, acciones que son disparadas cuando un conjunto de circunstancias se cumplen, como por ejemplo: el final de un año de proceso, después de instalado un nuevo servicio, etc. También incluye lo contrario, o sea, obligaciones a cumplir activadas por algún evento y acciones no exigibles como en un caso de fuerza mayor. (ver cláusula 2.7a "*EDP Audit Provisions*")

D) Cláusulas de Procedimiento

Cláusulas que definen la manera correcta de hacer una cosa. Por lo general especifican la secuencia de pasos a seguir para obtener algo. Ejemplo: cómo deben realizarse los cambios a las especificaciones del sistema.

E) Combinadas

Cláusulas que incluyen en su texto dos o más de las modalidades expuestas.

4.2.2 Catálogo de Cláusulas

Enumeraremos a continuación las categorías de cláusulas identificadas del contrato:

Artículo	Cláusula	Inciso	Tipo
1 Definitions			Definición
2 DP Services	*2.1 DP Services*		Definición
	2.2 Communications Links	a	Acciones exigibles
		b	Definición
		c	Validez
		d	Validez
	2.3 FDR NOAH Services	a	Definición
		b	Procedimiento
		c	Validez
		d	Procedimiento
		e	Procedimiento
		f	Validez
		g	Validez
	2.4 Conversion	a,b,c	Acciones exigibles
		d	Definición
		e,f,g	Acciones exigibles
		h	Procedimiento
		i	Acciones exigibles
		j	Acciones exigibles, Definición, Procedimiento
		k	Procedimiento
		l	Validez
		m	Definición
		n,o	Procedimiento
		p	Acciones exigibles

Artículo	Cláusula	Inciso	Tipo
		q	Procedimiento
		r,s	Acciones exigibles
		t	Definición, Procedimiento
	2.5 Account representative	a	Acciones exigibles
		b	Procedimiento
	2.6 Documentation and Training		Acciones exigibles
	2.7 EDP Audit Provisions; Backup Services		Acciones exigibles
	2.8 Correction of Errors		Acciones exigibles
	2.9 Reports; Record Retention	a	Acciones exigibles
		b	Procedimiento
	2.10 Management Meetings		Definición
	2.11 Performance Standards	a	Definición
		b,c	Validez
		d	Acciones exigibles
		e, f	Definición
	2.12 Interchange Settlement Services		Definición
	2.13 Enhancements; Mexican Requirements; U.S. Initiated Changes	x)	Acciones exigibles
		x) n)	Procedimiento
	2.14 Changes to the System Specifications		Procedimiento
	2.15 Notice of Adverse Event, Etc		Validez
	2.16 Business Continuity Plan; Security Procedures		Acciones exigibles
	2.17 Force Majeure	a	Validez
		b	Acciones exigibles
3 Exclusivity			Validez
4 Payment for DP Services	*4.1 Fees and Charges*	a	
		b	Definición
		c	Acciones exigibles
		d,e	Validez
	4.2 Price Changes		Procedimiento
	4.3 Method of Payment	a	Procedimiento

Artículo	Cláusula	Inciso	Tipo
		b,c	Acciones exigibles
	4.4 Overpayment		Acciones exigibles
	4.5 Interest		Acciones exigibles, Definición
	4.6 Payment of Reimbursable Expenses		Definición
5 Indemnification	*5.1 Indemnification by Customer*		Validez
	5.2 Indemnification by FDR		Validez
	5.3 Third Party Claims		Procedimiento
	5.4 Claims Period		Validez
6 Remedies; Limitation of Liability	*6.1 Remedies*		Definición
	6.2 Limitation on Liability		Validez
	6.3 No special Damages		Validez
7 Disclaimer of Warranties			Acciones exigibles
8 Term of Agreement	*8.1 Term*		Validez
	8.2 Timing of Obligations		Validez
9 Termination	*9.1 Termination by FDR for Default*		Validez
	9.2 Termination by Customer for Default		Validez
	9.3 Termination by Customer for Convenience		Validez, Acciones exigibles
	9.4 Termination by Customer for Force Majeure		Validez
	9.5 Termination by FDR for Force Majeure		Validez
	9.6 Termination by Customer for Adverse Event		Validez
	9.7 Termination by Customer for Major Devaluation		Validez, Acciones exigibles, Definición
	9.8 Termination by Customer for Major Taxes		Validez, Acciones exigibles
	9.9 General		Validez, Acciones exigibles
	9.10 Effect of Termination		Acciones exigibles
	9.11 Deconversion Assistance		Acciones exigibles
10 Confidential Nature of Data	*10.1 Customer's Propietary Information*		Definición

Artículo	Cláusula	Inciso	Tipo
	10.2 FDR's Propietary Information		Definición
	10.3 Confidentiality of Agreement		Validez
	10.4 Confidentiality		Validez
	10.5 Release of Information		Acciones exigibles
	10.6 Exclusions		Validez
	10.7 Limitations	a	Acciones exigibles
		b	Definición
	10.8 Remedy		Definición
11 Representations and Warranties	*11.1 FDR's Representation and Warranties*		Definición
	11.2 Customer's Representations and Warranties		Definición
12 Taxes	*12.1 Payment of Taxes*	a	Validez
		b,c,d	Definición
	12.2 Calculation and Payment of Sales Taxes		Acciones exigibles
	12.3 Return of Tax Benefits	a	Definición
		b	Acciones exigibles
	12.4 Cooperation		Definición
13 Dispute Resolution	*13.1 Arbitration*		Definición, Procedimiento
	13.2 Finality; Enforcement; Governance		Definición
	13.3 Injuctive Relief		Definición
	13.4 Continued Performance		Definición
14 General	*14.1 Assignment*		Validez
	14.2 Relationship of Parties		Validez
	14.3 Governing Law		Definición
	14.4 Notice		Procedimiento
	14.5 Headings and Interpretation		Definición
	14.6 Waiver		Definición
	14.7 Severability		Validez
	14.8 Risk of Loss		Definición
	14.9 Publicity		Acciones exigibles
	14.10 Entire Agreement		Validez

Tabla 4.2.1 Catálogo de Cláusulas

4.3 FORMULACIÓN DE REGLAS

Siendo el objetivo primordial del trabajo el proponer un modelo de conocimiento para la representación del contrato de FDR, dejaremos a un lado cualquier deliberación sobre los *shells* en general. No es la intención valorar la complejidad o funcionalidad de los *shells* existentes, ni tampoco si hay algunos otros con mayores facilidades que las de los aquí expuestos. Se han incluido aquí, con el objeto de ejemplificar la migración de una representación gráfica, conforme al modelo propuesto, a una susceptible de alimentarse en la computadora. También se utilizaron los *shells*, para realizar consultas ejemplo y verificar la validez del modelo, contra algunas de las cláusulas que a continuación se desarrollan:

Primeramente se presenta el texto original de la cláusula del contrato seguida de su representación en diagrama de árbol de decisión y de red de inferencia. A continuación se muestran las implementaciones en computadora de tales representaciones conforme a los *shells* utilizados. El modelo EXSYS y el IMP respectivamente.

Como primer paso, procedemos a identificar la categoría de la cláusula. Esto ha sido realizado a través de un estudio del contrato, tratando de identificar alguna clase que agrupe a las distintas cláusulas. Salvo una mejor opinión, el catálogo de cláusulas es lo que recomiendo para categorizarlas.

Una vez identificada la categoría (o clase) de la cláusula, su representación está basada, más que nada, en la forma de la hipótesis a comprobar. Las premisas de las reglas son, básicamente, el conjunto de condiciones, status, datos y eventos que concluyen la hipótesis. Con la posible excepción de las cláusulas de procedimiento, las cuales contienen preguntas referentes a una secuencia de eventos, (aunque no siempre), ningún tipo de cláusula contiene una fórmula específica para construir las preguntas en base al texto del contrato. Por tanto, no es un buen comienzo el representar la cláusula partiendo del tipo de preguntas; por el contrario, las hipótesis sí nos presentan un buen inicio para desarrollar las reglas.

Estos son los tipos de hipótesis que se deben buscar según el tipo de cláusulas:

CLÁUSULA	HIPÓTESIS
Definición	Aserción sobre un concepto
Validez	Acatamiento o violación de los estatutos acordados
Acciones exigibles	Acción a efectuar o solicitar. Acción susceptible de que nos sea exigida.
Procedimiento	Objetivo a lograr o conseguir

Tabla 4.3.1 Tipos de Hipótesis

Esto no es derivado, de ninguna manera, de un análisis de la semántica completa de las frases y oraciones contenidas en documentos de aspecto legal como contratos, estatutos y leyes. Sólo es una reducción del problema particular que estamos tratando, para enfocar una solución práctica producto de la revisión del documento.

Consideraciones del modelo EXSYS.

Se ha omitido el uso de cláusulas *ELSE* por las siguientes razones:

1. Un número considerable de paquetes de *software* de desarrollo de SE no permiten el uso de reglas *IF-THEN-ELSE*. El modelo EXSYS sí permite el uso de reglas que incluyan el *ELSE*; pero como aquí se pretende establecer un modelo general que pueda implementarse en diferentes plataformas (no solamente en un sistema como EXSYS) se tomó en cuenta este punto para no utilizarlas. Por supuesto que la tecnología y los nuevos desarrollos, harán cada vez más difundido el uso de cláusulas *ELSE*, las cuales pueden incluirse en futuros modelos.
2. La validación de reglas con *ELSE* es considerablemente más difícil que sus equivalentes *IF-THEN*.
3. En el proceso de inferencia, tales reglas (*IF-THEN-ELSE*) tenderán a alcanzar una conclusión siempre (la de la porción *THEN* o la de la porción *ELSE*). Esto puede llevar a resultados no anticipados.

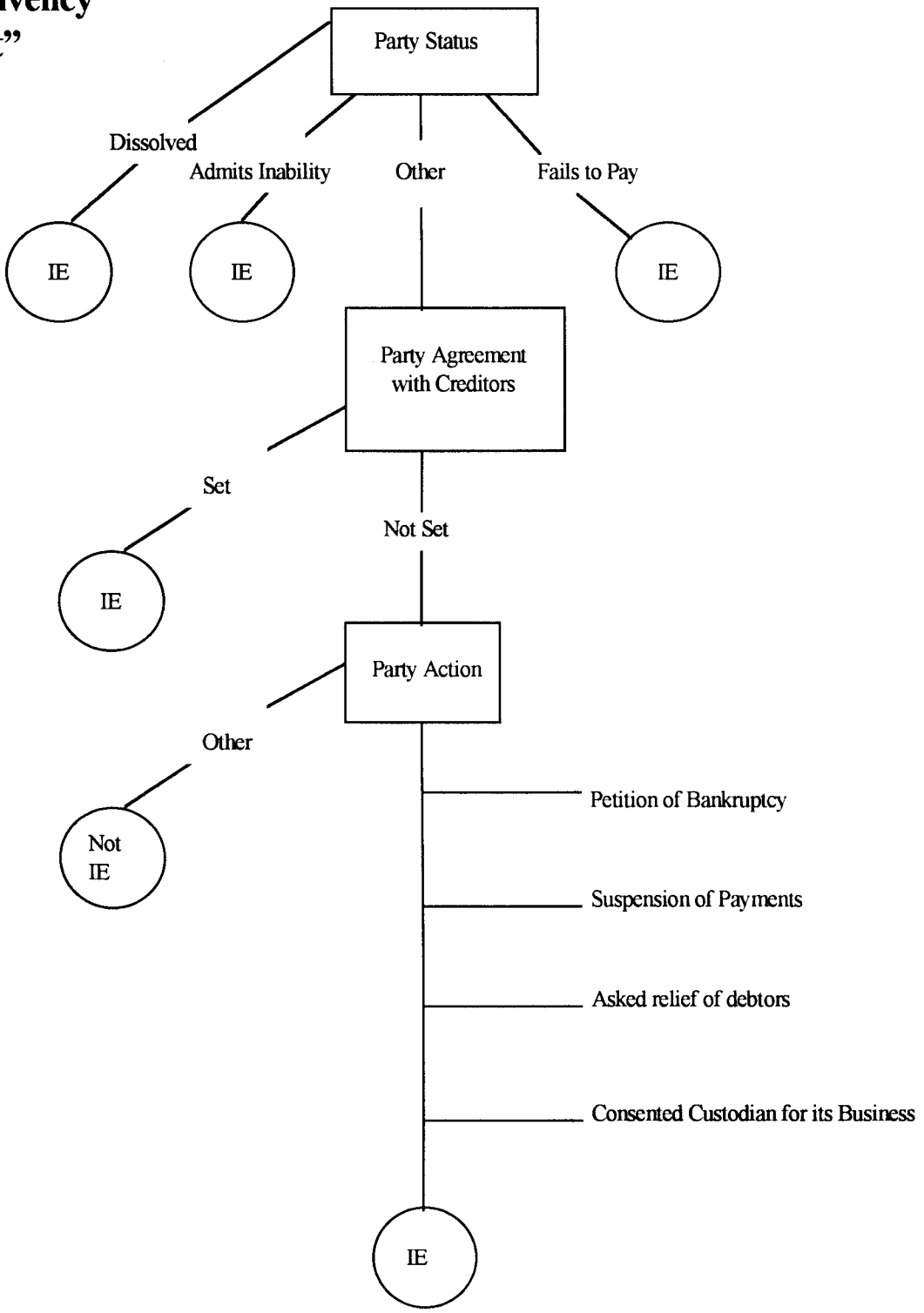
4.3.1 Cláusulas de Definición

"Insolvency Event".*Occurs, with respect to any Party, when such Party:*

1. *is dissolved, becomes insolvent, generally fails to pay or admits in writing its inability generally to pay its debts as they become due;*
2. *makes a general assignment, arrangement, or composition agreement with or for the benefit of its creditors; or*
3. *files a petition in bankruptcy, suspension of payments or institutes any action under applicable law for the relief of debtors or seeks or consents to the appointment of an administrator, receiver, custodian, or similar official for the winding up of its business (or has such a petition or action filed against it and such petition, action or appointment is not dismissed or stayed within thirty (30) calendar days).*

Las cláusulas de definición no presentan gran problema dado que las reglas son un conjunto de preguntas que verifican la existencia de los atributos que "definen" un concepto (objeto) del contrato.

1.- Definitions "Insolvency Event"



IE - Insolvency Event

Subject:
Conjunto de reglas que determinan el concepto de Insolvencia (Cláusula 1)

Author:
Arturo Lee

Starting text:
Contestar a las preguntas considerando el estado de alguna de las partes involucradas en el contrato (cliente, proveedor, etc.)

Ending text:
Para las condiciones establecidas

Uses all applicable rules in data derivations.

RULES:

RULE NUMBER: 1

IF:
Party Status is Dissolved
THEN:
Insolvency Event Occurs - Probability=1

RULE NUMBER: 2

IF:
Party Status is Admits inability
THEN:
Insolvency Event Occurs - Probability=1

RULE NUMBER: 3

IF:
Party Status is Fails to pay
THEN:
Insolvency Event Occurs - Probability=1

RULE NUMBER: 4

IF:
Party Status is Other
and Party Agreement with debtors is Set
THEN:

Insolvency Event Occurs - Probability=1

RULE NUMBER: 5

IF:
Party Status is Other
and Party Agreement with debtors is Not Set
and Party action is Petition of Bankruptcy
THEN:
Insolvency Event Occurs - Probability=1

RULE NUMBER: 6

IF:
Party Status is Other
and Party Agreement with debtors is Not Set
and Party action is Suspension of payments
THEN:
Insolvency Event Occurs - Probability=1

RULE NUMBER: 7

IF:
Party Status is Other
and Party Agreement with debtors is Not Set
and Party action is Asked relief of debtors
THEN:
Insolvency Event Occurs - Probability=1

RULE NUMBER: 8

IF:
Party Status is Other
and Party Agreement with debtors is Not Set
and Party action is Consented Custodian for its business
THEN:
Insolvency Event Occurs - Probability=1

RULE NUMBER: 9

IF:
Party Status is Other

and Party Agreement with debtors is Not Set
and Party action is Other
THEN:
Insolvency Event Occurs - Probability=0

QUALIFIERS:

1 Party Status is

Dissolved
Admits inability
Fails to pay
Other

Used in rule(s): 1 2 3 4 5 6
7 8 9

2 Party Agreement with debtors is

Set
Not Set

Used in rule(s): 4 5 6 7 8 9

3 Party action is

Petition of Bankruptcy
Suspension of payments
Asked relief of debtors
Consented Custodian for its business
Other

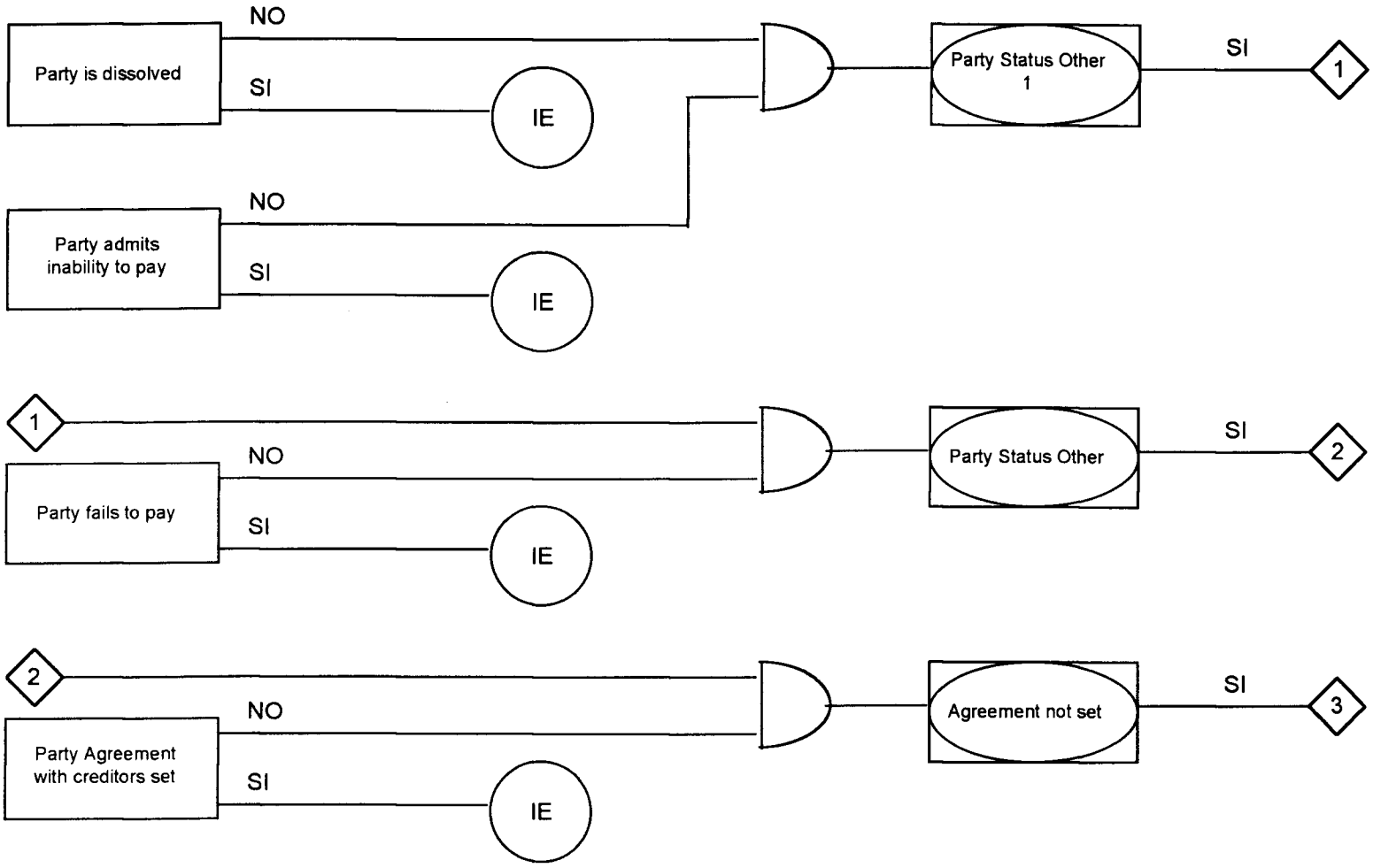
Used in rule(s): 5 6 7 8 9

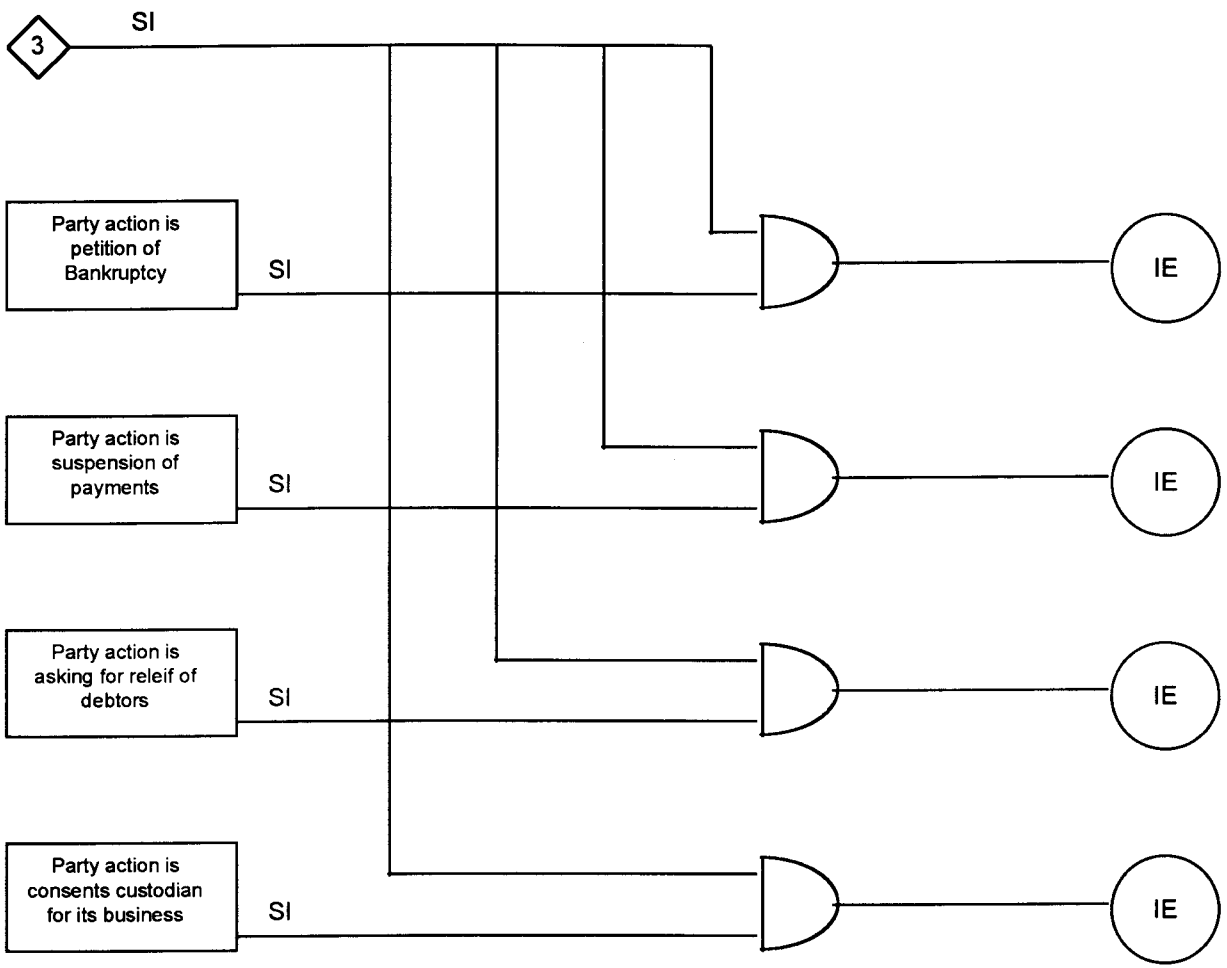
CHOICES:

1 Insolvency Event Occurs

Used in rule(s): (1) (2) (3) (4) (5) (6)
(7) (8) (9)

1.- Definitions "Insolvency Event"





continuación...1.- Definitions

Implementación de una cláusula de "definición" en modelo IMP

```
imp("s","n","Insolvency Event","pos","Party is Dissolved","dummy","dummy",1)
imp("s","n","Insolvency Event","pos","Party admits inability to pay","dummy","dummy",1)
imp("s","n","Insolvency Event","pos","Party fails to pay","dummy","dummy",1)
imp("s","n","Insolvency Event","pos","Party Agreement with creditors set","dummy","dummy",1)
imp("a","r","Party Status is Other 1","neg","Party is Dissolved","neg","Party admits inability to pay",1)
imp("a","r","Party Status is Other 2","pos","Party Status is Other 1","neg","Party fails to pay",1)
imp("a","r","Agreement not set","pos","Party Status is Other 2","neg","Party Agreement with creditors set",1)
imp("a","n","Insolvency Event","pos","Agreement not set","pos","Party Action is Petition of Bankruptcy",1)
imp("a","n","Insolvency Event","pos","Agreement not set","pos","Party Action is Suspension of Payments",1)
imp("a","n","Insolvency Event","pos","Agreement not set","pos","Party Action is asking relief of debtors",1)
imp("a","n","Insolvency Event","pos","Agreement not set","pos","Party Action is consents custodian for its business",1)
hypothesis_node("Insolvency Event")
terminal_node("Party is Dissolved")
terminal_node("Party admits inability to pay")
terminal_node("Party fails to pay")
terminal_node("Party Agreement with creditors set")
terminal_node("Party Action is Petition of Bankruptcy")
terminal_node("Party Action is Suspension of Payments")
terminal_node("Party Action is asking for relief of debtors")
terminal_node("Party Action is consents custodian for its business")
```

4.3.2 Cláusulas de Validez

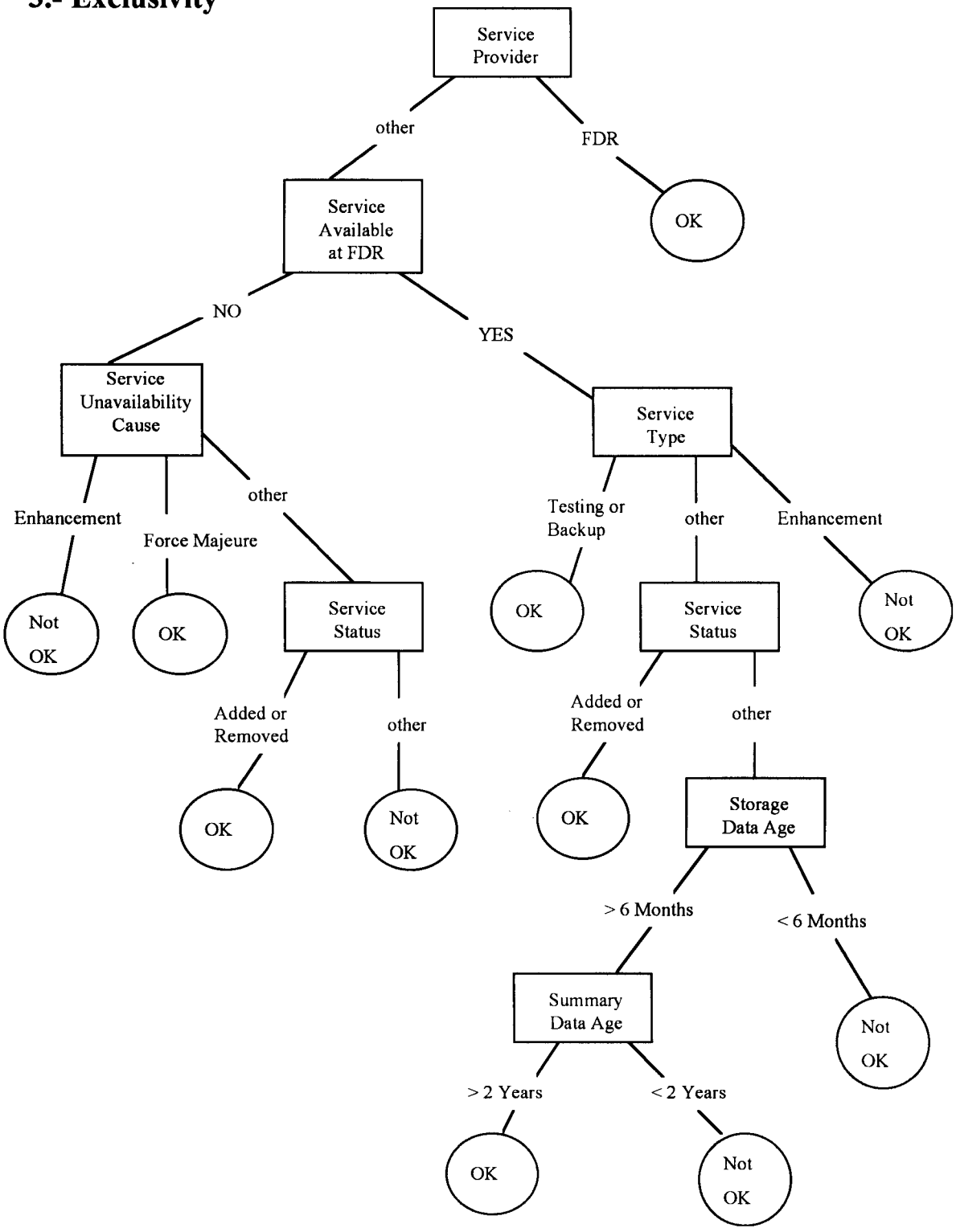
3.1 Sole and Exclusive Provider. *After the Conversion Date, (I) FDR shall be the sole and exclusive provider to Customer and its Affiliates of the DP Services described in the System Specifications, (ii) neither Customer nor any of its Affiliates shall perform, for itself, themselves or any other party, any of such DP Services, and (iii) neither Customer nor any of its Affiliates shall, pursuant to agreement or otherwise, have any third party perform or provide any of such DP Services for them; provided, however, that (A) if at any time FDR fails to provide any such services in accordance with the terms of this Agreement (regardless of whether or not such failure is excused due to the occurrence of Force Majeure Event) and such failure results, or is likely to result, in a material interruption in or disruption to a Permitted User's business activities or operations, then Customer and its Affiliates shall be free to perform such services from a third party for so long as FDR shall fail to provide such services; (B) the limitations contained in this Section 3.1 shall not apply to testing of any backup replacement system by Customer or its Affiliates; (C) the limitations contained in this Section 3.1 shall not apply to storage of, access to, and processing of, transaction data that is more than six (6) months and annual summary data that is more than two (2) years old; and (D) the limitations contained in this Section 3.1 shall not apply to any services that FDR adds to or removes from the System Specifications after the date hereof or to any Enhancement requested by Customer that FDR does not for any reason effect pursuant to Section 2.13. (b) [Exception for Merchant-related services -- to come]*

Aquí, lo que se pretende lograr es inferir si se ha incurrido en una violación a la cláusula (en este caso de "exclusividad"), para lo cual las hipótesis a verificar son:

Exclusivity is Keeped	EK	En el caso de que la exclusividad se ha respetado
Exclsivity is Broken	EB	Cuando la exclusividad no se ha conservado

La primera nos dice que nuestras acciones no provocan, provocaron o provocarán (según el tiempo de la consulta al sistema) una falta sobre el acuerdo de exclusividad de los servicios de proceso.

3.- Exclusivity



Subject:
Conjunto de reglas para inferir si se ha incurrido en una violación a la cláusula "3.- *Exclusivity*" (Exclusividad) del Contrato

Author:
Arturo Lee

Starting text:
Contestar a las preguntas considerando un tipo de Servicio. (ej. impresión de estados de cuenta, proceso de tarjetas, etc.)

Ending text:
Para las condiciones establecidas:

Uses all applicable rules in data derivations.

RULES:

RULE NUMBER: 1

IF:
Service Provider is FDR
THEN:
Exclusivity Keeped - Probability=1

RULE NUMBER: 2

IF:
Service Provider is Other
and Service available at FDR is False
and Service Unavailable Cause is Force Majeure
THEN:
Exclusivity Keeped - Probability=1

RULE NUMBER: 3

IF:
Service Provider is Other
and Service available at FDR is False
and Service Unavailable Cause is Other
and Service Status is Removed from FDR Catalog
THEN:

Exclusivity Keeped - Probability=1

RULE NUMBER: 4

IF:
Service Provider is Other
and Service available at FDR is False
and Service Unavailable Cause is Other
and Service Status is Added to FDR Catalog
THEN:
Exclusivity Keeped - Probability=1

RULE NUMBER: 5

IF:
Service Provider is Other
and Service available at FDR is False
and Service Unavailable Cause is Other
and Service Status is Other
THEN:
Exclusivity Broken - Probability=1

RULE NUMBER: 6

IF:
Service Provider is Other
and Service available at FDR is True
and Service Type is Testing
THEN:
Exclusivity Keeped - Probability=1

RULE NUMBER: 7

IF:
Service Provider is Other
and Service available at FDR is True
and Service Type is Backup
THEN:
Exclusivity Keeped - Probability=1

RULE NUMBER: 8

IF:

Service Provider is Other
and Service available at FDR is True
and Service Type is Other
and Service Status is Removed from FDR Catalog
THEN:
Exclusivity Keeped - Probability=1

RULE NUMBER: 9

IF:
Service Provider is Other
and Service available at FDR is True
and Service Type is Other
and Service Status is Added to FDR Catalog
THEN:
Exclusivity Keeped - Probability=1

RULE NUMBER: 10

IF:
Service Provider is Other
and Service available at FDR is True
and Service Type is Other
and Service Status is Current on Contract
and [STORAGE DATA AGE] > 6 MONTHS
and [SUMMARY DATA AGE] > 2 YEARS
THEN:
Exclusivity Keeped - Probability=1

RULE NUMBER: 11

IF:
Service Provider is Other
and Service available at FDR is True
and Service Type is Other
and Service Status is Current on Contract
and [STORAGE DATA AGE] > 6 MONTHS
and [STORAGE DATA AGE] < 2 YEARS
THEN:
Exclusivity Broken - Probability=1

RULE NUMBER: 12

IF:
Service Provider is Other

and Service available at FDR is True
and Service Type is Other
and Service Status is Current on Contract
and [STORAGE DATA AGE] < 6 MONTHS
THEN:
Exclusivity Broken - Probability=1

RULE NUMBER: 13

IF:
Service Provider is Other
and Service available at FDR is True
and Service Type is Requested Enhancement
THEN:
Exclusivity Broken - Probability=1

RULE NUMBER: 14

IF:
Service Provider is Other
and Service available at FDR is False
and Service Unavailable Cause is Requested Enhancement
THEN:
Exclusivity Broken - Probability=1

QUALIFIERS:

1 Service Provider is

FDR
Other

Used in rule(s): 1 2 3 4 5 6
7 8 9 10 11 12
13 14

2 Service available at FDR is

True
False

Used in rule(s): 2 3 4 5 6 7
8 9 10 11 12 13
14

3 Service Unavailable Cause is

Force Majeure
Other
Requested Enhancement

Used in rule(s): 2 3 4 5 14

4 Service Status is

Removed from FDR Catalog
Added to FDR Catalog
Other
Current on Contract

Used in rule(s): 3 4 5 8 9 10
11 12

5 Service Type is

Testing
Backup
Requested Enhancement
Other

Used in rule(s): 6 7 8 9 10 11
12 13

CHOICES:

1 Exclusivity Keeped

Used in rule(s): (1) (2) (3) (4) (6) (7)
(8) (9) (10)

2 Exclusivity Broken

Used in rule(s): (5) (11) (12) (13) (14)

FORMULAS:

1 [STORAGE DATA AGE] > 6 MONTHS

Used in rule(s): 10

2 [SUMMARY DATA AGE] > 2 YEARS

Used in rule(s): 10

3 [STORAGE DATA AGE] > 6 MONTHS

Used in rule(s): 11

4 [STORAGE DATA AGE] < 2 YEARS

Used in rule(s): 11

5 [STORAGE DATA AGE] < 6 MONTHS

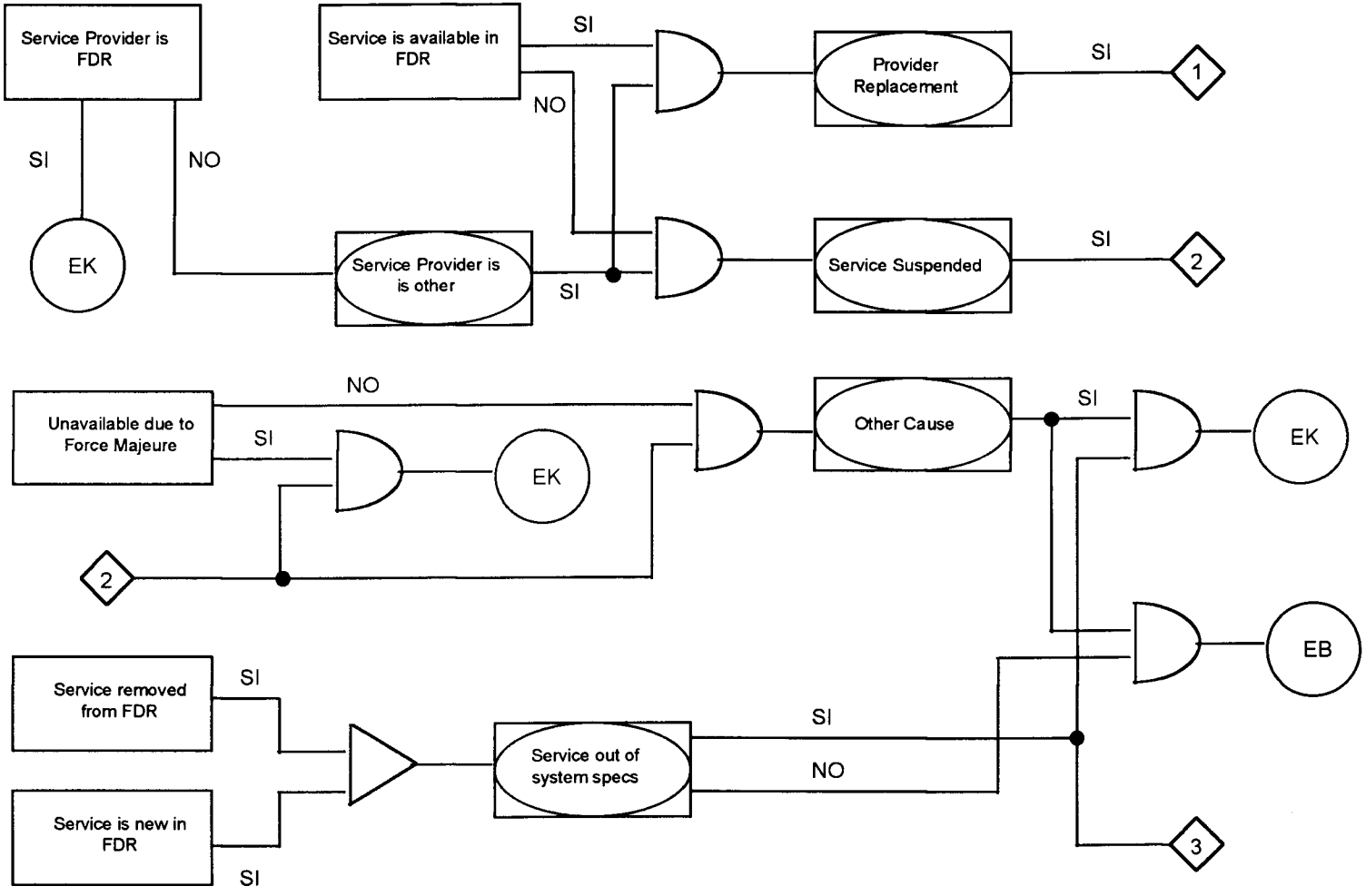
Used in rule(s): 12

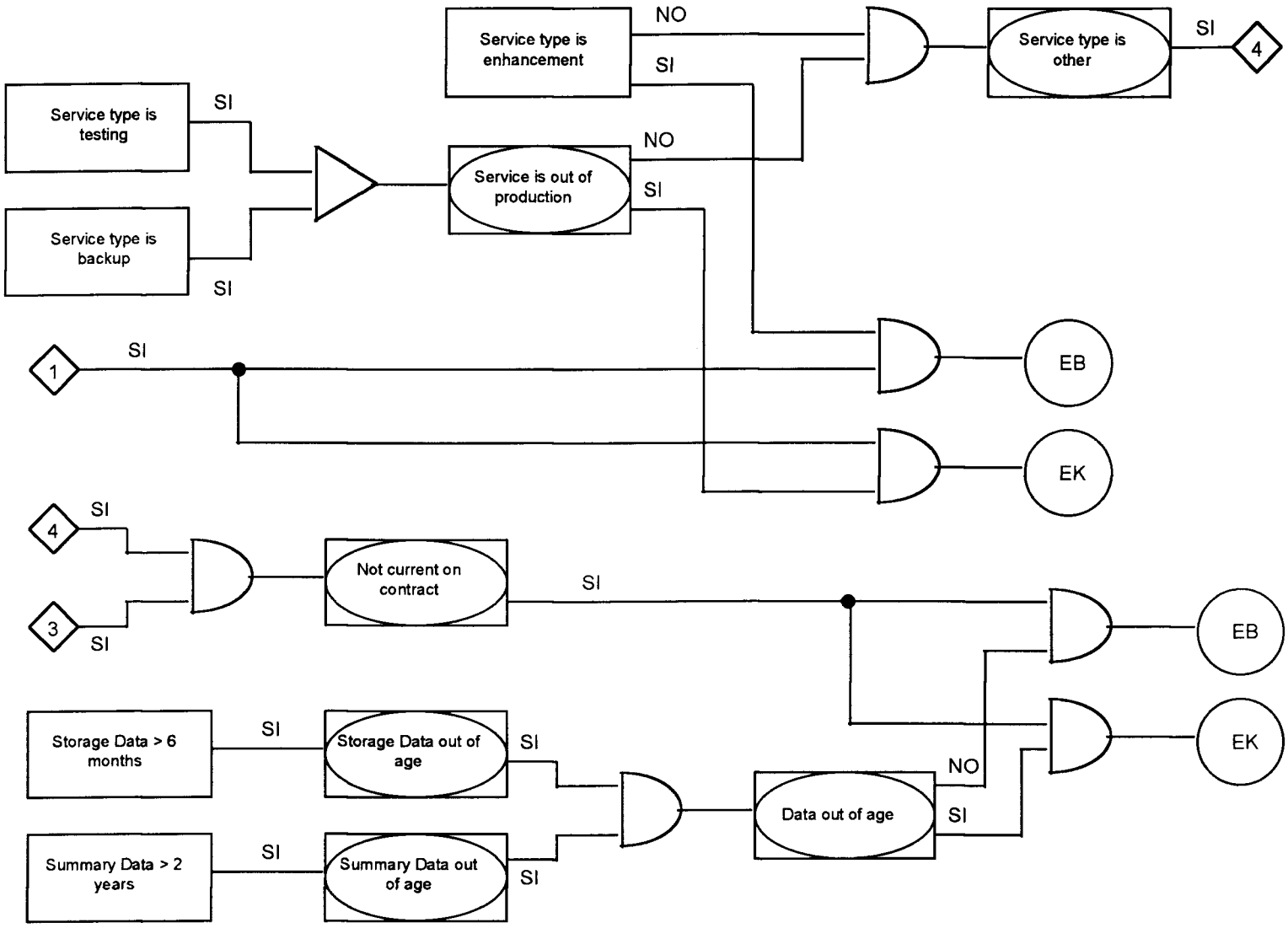
VARIABLES:

1 STORAGE DATA AGE
Edad de los datos almacenados en FDR.
Numeric variable

2 SUMMARY DATA AGE
Edad de los datos resumen y condensados
Numeric variable

3.- Exclusivity





continuación... 3.- Exclusivity

Implementación de una cláusula de "validez" en modelo IMP

```
imp("s","n","Exclusivity is Keeped","pos","Service Provider is FDR","dummy","dummy",1)
imp("a","n","Exclusivity is Keeped","pos","Service Suspended","pos","Unavailable due to Force Majeure",1)
imp("a","n","Exclusivity is Keeped","pos","Other Cause","pos","Service Out of System Specs",1)
imp("a","r","Exclusivity is Broken","pos","Other Cause","neg","Service Out of System Specs",1)
imp("a","n","Exclusivity is Keeped","pos","Provider Replacement","pos","Service Type is not Production",1)
imp("a","n","Exclusivity is Broken","pos","Provider Replacement","pos","Service Type is Enhancement",1)
imp("a","n","Exclusivity is Keeped","pos","Not Current on Contract","pos","Service Out of System Specs",1)
imp("a","n","Exclusivity is Keeped","neg","Not Current on Contract","pos","Data Out of Age",1)
imp("a","r","Exclusivity is Broken","neg","Not Current on Contract","neg","Data Out of Age",1)
imp("s","r","Service Provider is Other","neg","Service Provider is FDR","dummy","dummy",1)
imp("a","n","Provider Replacement","pos","Service Provider is Other","pos","Service is available in FDR",1)
imp("a","r","Service Suspended","pos","Service Provider is Other","neg","Service is available in FDR",1)
imp("a","r","Other Cause","pos","Service Suspended","neg","Unavailable due to Force Majeure",1)
imp("o","n","Service Out of System Specs","pos","Service is Removed from FDR","pos","Service is new in FDR",1)
imp("o","r","Service is not Production","pos","Service Type is Testing","pos","Service Type is Backup",1)
imp("a","r","Service Type is Other","neg","Service Type is not Production","neg","Service Type is Enhancement",1)
imp("a","n","Not Current on Contract","pos","Service Type is Other","pos","Service Out of System Specs",1)
imp("f","n","Storage Data Out of Age","pos","Storage_Data > 6.0 Months","dummy","dummy",1)
imp("f","n","Summary Data Out of Age","pos","Summary_Data > 2.0 Years","dummy","dummy",1)
imp("a","n","Data Out of Age","pos","Storage Data Out of Age","pos","Summary Data Out of Age",1)
hypothesis_node("Exclusivity is Keeped")
hypothesis_node("Exclusivity is Broken")
terminal_node("Service Provider is FDR")
terminal_node("Service is available in FDR")
terminal_node("Unavailable due to Force Majeure")
terminal_node("Service is Removed from FDR")
terminal_node("Service is new in FDR")
terminal_node("Service Type is Testing")
terminal_node("Service Type is Backup")
terminal_node("Service Type is Enhancement")
```

4.3.3 Cláusulas de Acciones Exigibles o Acciones a Ejecutar

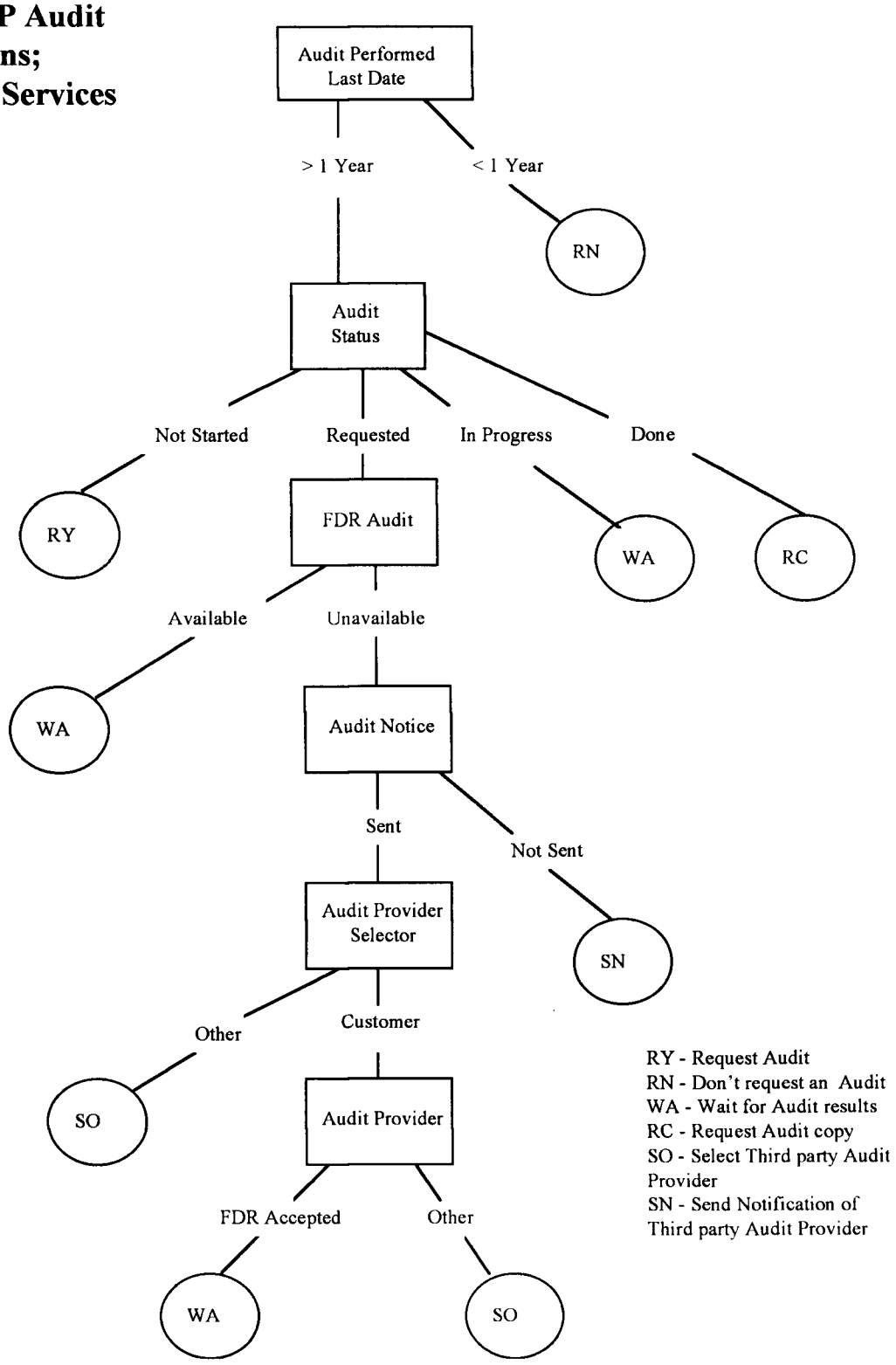
2.7 EDP Audit Provisions; Backup Services.

(a) At least annualy during the Term of this Agreement, FDR will provide for a third party selected by FDR that is a major accounting form of national reputation to perform an audit of the electronic data processing environment maintened by FDR to provide the DP Services contemplated under this Agreement. FDR promptly shall provide Customer with a copy of the results of each such audit. If FDR does not have the audit performed at least annualy, FDR upon reasonable advance notice will allow an independent third party, selected by Customer and approved by FDR (which approval will not be unreasonable withheld or delayed), to perform an audit of the electronic data processing environment maintained by FDR to provide the DP Services contemplated under this Agreement with FDR and Customer each being provided with a copy of the results of the audit to the extent permitted by applicable laws and regulations and FDR shall pay for such audit.

Para esta cláusula, se ha identificado que contiene las siguientes hipótesis que corresponden a las acciones o peticiones sobre las auditorías a los servicios prestados:

Request Audit	RY	Solicitar auditoría
Don't request Audit	RN	No es necesario solicitar auditoría
Wait for Audit results	WA	Espere por los resultados de la auditoría
Request audit copy	RC	Solicite copia de los resultados de la auditoría
Select third party audit provider	SO	Seleccione un tercero que efectúe la auditoría
Send notification of third party audit provider	SN	Envíe notificación a FDR de que seleccionará a un tercero que efectúe la auditoría

2.7a EDP Audit Provisions; Backup Services



RY - Request Audit
RN - Don't request an Audit
WA - Wait for Audit results
RC - Request Audit copy
SO - Select Third party Audit Provider
SN - Send Notification of Third party Audit Provider

Subject:

Conjunto de reglas que alertan o previenen sobre la exigencia de acciones específicas con respecto a las Provisiones de Auditoría y Servicios de Backup; cláusula "2.7a .- ***EDP Audit Provisions"***

Author:
Arturo Lee

Starting text:
Contestar a las preguntas considerando las fechas en las que se ha hecho (si aplica) un auditoría al servicio proporcionado por FDR

Ending text:
Para las condiciones establecidas, se recomienda:

Uses all applicable rules in data derivations.

RULES:

RULE NUMBER: 1

IF:
[TODAY] - [AUDIT LAST DATE] >= 10000
THEN:
Elapsed Time since Last Audit is > 1 Year
ELSE:
Elapsed Time since Last Audit is < 1 Year

RULE NUMBER: 2

IF:
Elapsed Time since Last Audit is < 1 Year
THEN:
Don't request an Audit - Probability=1

RULE NUMBER: 3

IF:

Elapsed Time since Last Audit is > 1 Year
and Audit Status is In Progress
THEN:
Wait for Audit Results - Probability=1

RULE NUMBER: 4

IF:
Elapsed Time since Last Audit is > 1 Year
and Audit Status is Done
THEN:
Request Audit Copy - Probability=1

RULE NUMBER: 5

IF:
Elapsed Time since Last Audit is > 1 Year
and Audit Status is Not Started
THEN:
Request an Audit - Probability=1

RULE NUMBER: 6

IF:
Elapsed Time since Last Audit is > 1 Year
and Audit Status is Requested
and Audit Performed by FDR is Unavailable
and Audit Notification Status is Not Sent
THEN:
Send Notification of Third Party needed - Probability=1

RULE NUMBER: 7

IF:
Elapsed Time since Last Audit is > 1 Year
and Audit Status is Requested
and Audit Performed by FDR is Unavailable
and Audit Notification Status is Sent
and Audit Provider selected by Other
THEN:
Select Third Party Audit Provider - Probability=1

RULE NUMBER: 8

IF:
Elapsed Time since Last Audit is > 1 Year
and Audit Status is Requested
and Audit Performed by FDR is Unavailable
and Audit Notification Status is Sent
and Audit Provider selected by Customer
and Audit Provider accepted by FDR No
THEN:
Select Third Party Audit Provider - Probability=1

RULE NUMBER: 9

IF:
Elapsed Time since Last Audit is > 1 Year
and Audit Status is Requested
and Audit Performed by FDR is Unavailable
and Audit Notification Status is Sent
and Audit Provider selected by Customer
and Audit Provider accepted by FDR Yes
THEN:
Wait for Audit Results - Probability=1

RULE NUMBER: 10

IF:
Elapsed Time since Last Audit is > 1 Year
and Audit Status is Requested
and Audit Performed by FDR is Available
THEN:
Wait for Audit Results - Probability=1

QUALIFIERS:

1 Elapsed Time since Last Audit is

> 1 Year
< 1 Year

Used in rule(s): (1) [1] 2 3 4 5
6 7 8 9 10

2 Audit Status is

Not Started
Requested
In Progress
Done

Used in rule(s): 3 4 5 6 7 8 9 10

3 Audit Performed by FDR is

Available
Unavailable

Used in rule(s): 6 7 8 9 10

4 Audit Notification Status is

Sent
Not Sent

Used in rule(s): 6 7 8 9

5 Audit Provider selected by

Customer
Other

Used in rule(s): 7 8 9

6 Audit Provider accepted by FDR

Yes
No

Used in rule(s): 8 9

CHOICES:

1 Request an Audit

Used in rule(s): (5)

2 Don't request an Audit

Used in rule(s): (2)

3 Wait for Audit Results

Used in rule(s): (3) (9) (10)

4 Request Audit Copy

Used in rule(s): (4)

5 Select Third Party Audit Provider

Used in rule(s): (7) (8)

6 Send Notification of Third Party needed

Used in rule(s): (6)

FORMULAS:

1 [TODAY] - [AUDIT LAST DATE] >=10000

Used in rule(s):

2 [TODAY] - [AUDIT LAST DATE] >= 10000

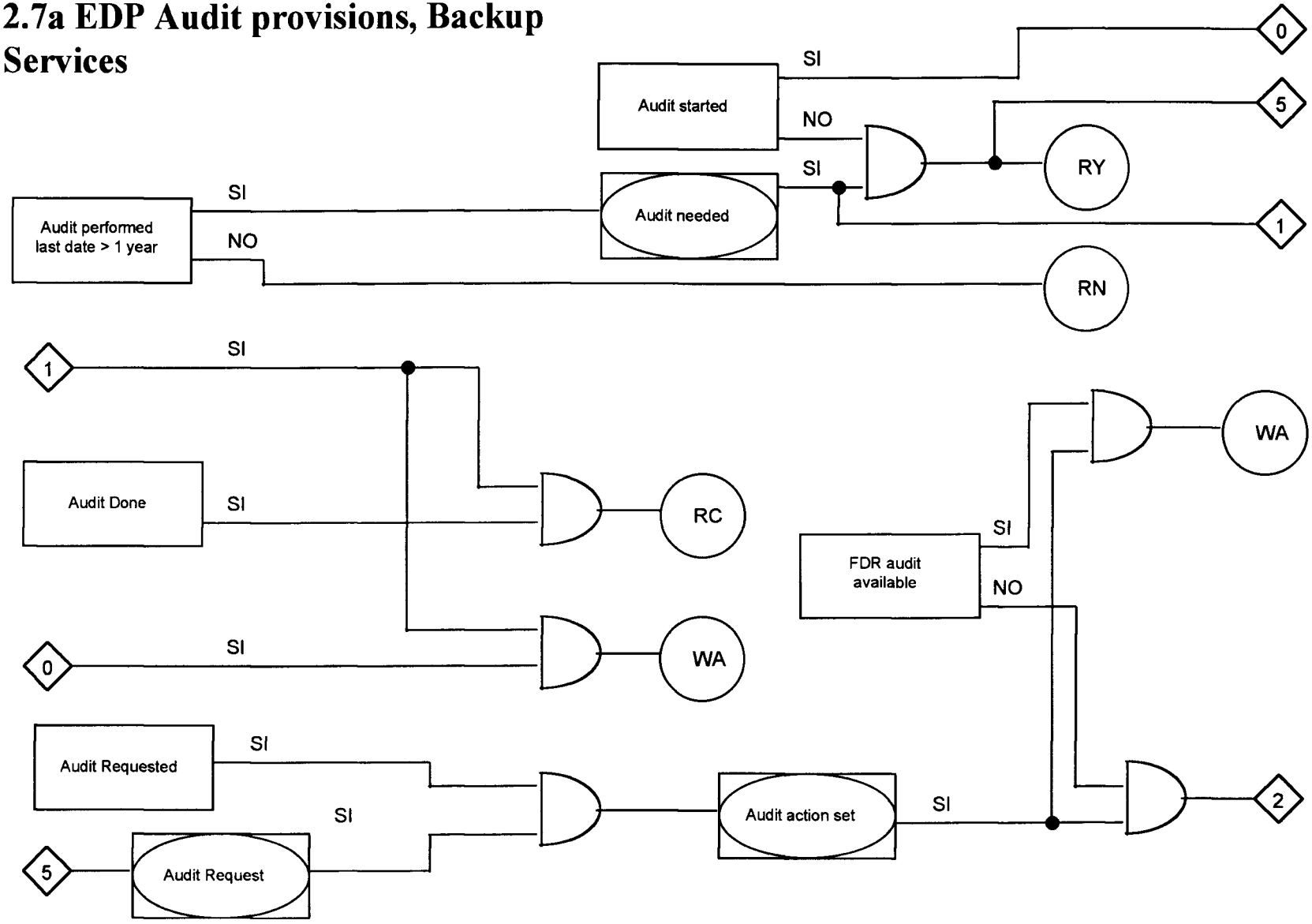
Used in rule(s): 1

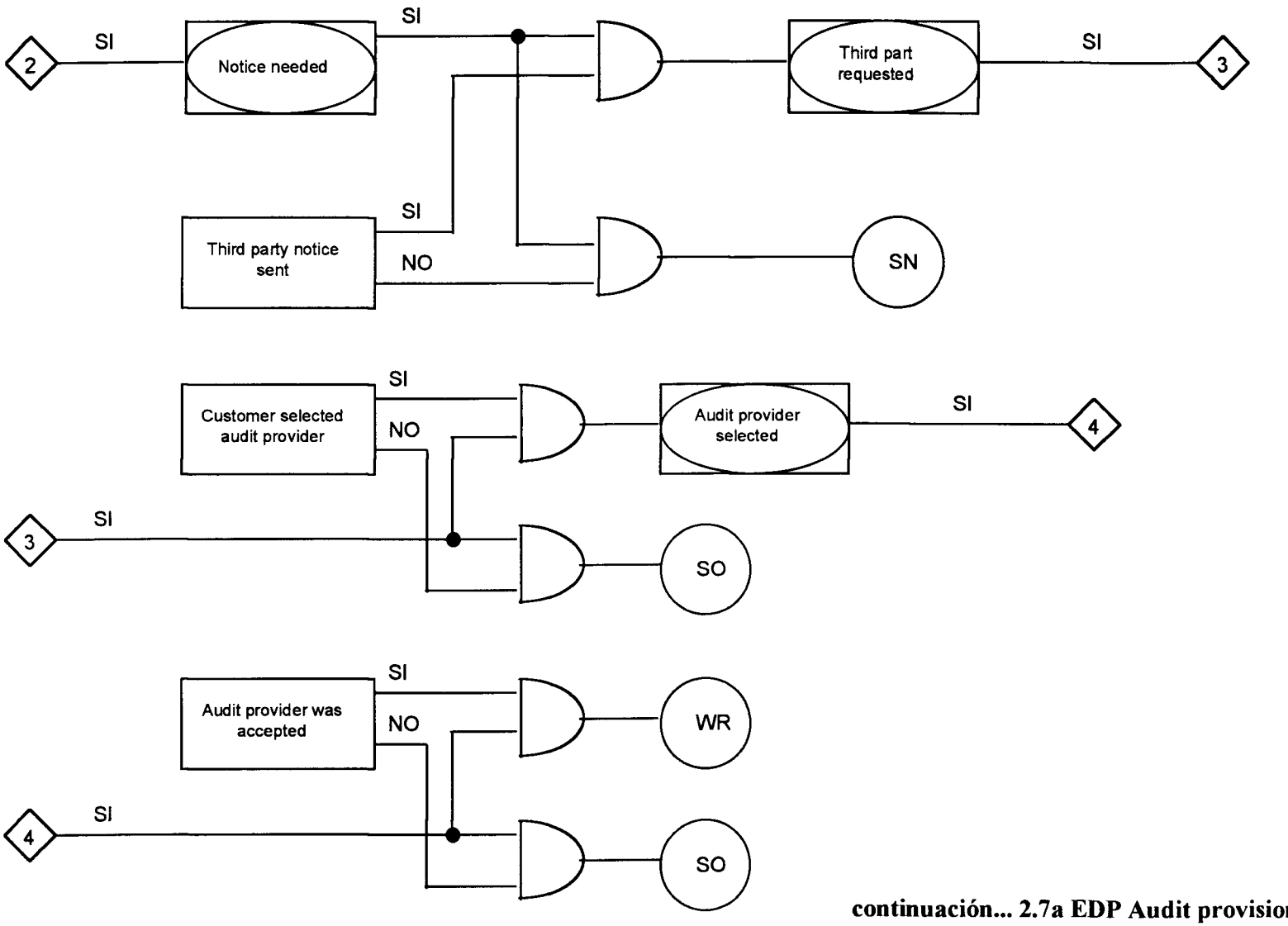
VARIABLES:

1 TODAY
Fecha Actual
Numeric variable

2 AUDIT LAST DATE
Fecha de la última Auditoría
Numeric variable

2.7a EDP Audit provisions, Backup Services





continuación... 2.7a EDP Audit provisions, Backup Services

Implementación de una cláusula de "acciones exigibles" en modelo IMP

```
imp("s","n","Audit Needed","pos","Audit Last Date > 1 Year","dummy","dummy",1)
imp("s","r","Don't Request Audit","neg","Audit Last Date > 1 Year","dummy","dummy",1)
imp("a","r","Request Audit","pos","Audit Needed","neg","Audit Started",1)
imp("a","n","Wait Audit Results","pos","Audit Needed","pos","Audit Started",1)
imp("a","n","Request Audit Copy","pos","Audit Needed","pos","Audit Done",1)
imp("a","r","Audit Request","pos","Audit Needed","neg","Audit Request",1)
imp("a","n","Audit Action Set","pos","Audit Requested","pos","Audit Needed",1)
imp("a","r","Notice Needed","pos","Audit Action Set","neg","FDR Audit Available",1)
imp("a","n","Third Party Requested","pos","Third Party Notice Sent","pos","Notice Needed",1)
imp("a","r","Send Third Party Audit Notification","pos","Notice Needed","neg","Third Party Notice Sent",1)
imp("a","n","Audit Provider Selected","pos","Third Party Requested","pos","Customer Selected Audit Provider",1)
imp("a","r","Select Third Party Audit Provider","pos","Third Party Requested","neg","Customer Selected Audit Provider",1)
imp("a","n","Wait Audit Results","pos","Audit Provider Selected","pos","Audit Provider was FDR Accepted",1)
imp("a","n","Request Payment from FDR","pos","Audit Provider Selected","pos","Audit Provider was FDR Accepted",1)
imp("a","r","Select Third Party Audit Provider","pos","Audit Provider Selected","neg","Audit Provider was FDR Accepted",1)
hypothesis_node("Request Audit")
hypothesis_node("Don't Request Audit")
hypothesis_node("Wait Audit Results")
hypothesis_node("Request Audit Copy")
hypothesis_node("Send Third Party Audit Notification")
hypothesis_node("Select Third Party Audit Provider")
hypothesis_node("Request Payment from FDR")
terminal_node("Audit Last Date > 1 Year")
terminal_node("Audit Started")
terminal_node("Audit Done")
terminal_node("Audit Requested")
terminal_node("FDR Audit Available")
terminal_node("Third Party Notice Sent")
terminal_node("Customer Selected Audit Provider")
terminal_node("Audit Provider was FDR Accepted")
```

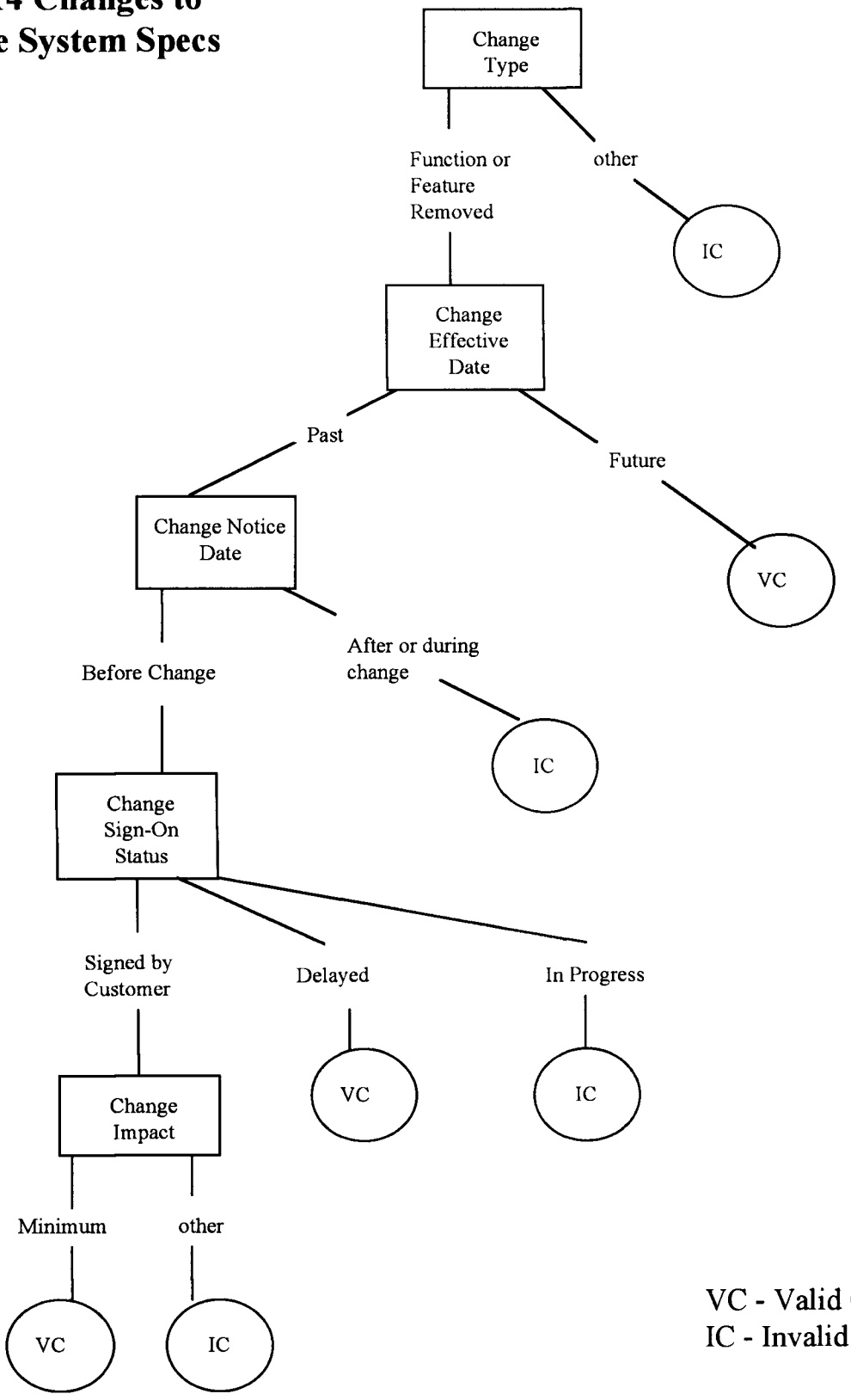
4.3.4 Cláusulas de Procedimiento

2.14 Changes to the System Specifications. *FDR convenants to provide to Customer a notice in the form of a Customer Bulletin that describes each change to the System Specifications required in connection with an Enhancement made pursuant to Section 2.13 or any other change to the System Specifications at a reasonable time before such change becomes effective; provided, however, that (i) FDR shall asure that, at all times during the Term, Customer has a set of manuals that when modified by later dated Customer Bulletins accurately describe the functions performed by the FDR System (ii) FDR is not permitted to remove any feature or function (including without limitation any security feature or function) from the System Specifications, whether or not such feature or function is then used by Permitted Users, unless consented to by Customer, which consent will not be unreasonably withheld or delayed, and (iii) FDR shall implement changes in the System Specifications in a manner that is reasonably likely to minimize the inconvenience or disruption to Permitted Users associated with such changes.*

El ejemplo de cláusulas de procedimiento corresponde a los "Cambios a las especificaciones del sistema". El procedimiento representa la forma correcta de implementar un cambio al sistema por parte de FDR; por lo tanto, la hipótesis o conclusión es si el procedimiento resulta en un cambio válido o no aceptado.

Valid Change	VC	El procedimiento para implementar el cambio es correcto
Invalid Change	IC	El procedimiento para implementar el cambio no es correcto

2.14 Changes to the System Specs



VC - Valid Change
IC - Invalid Change

Subject:

Conjunto de reglas para verificar la validez de un cambio conforme a lo estipulado en la cláusula 2.14 del Contrato

Author:
Arturo Lee

Starting text:
Contestar las preguntas con referencia a un cambio en el sistema, efectuado por FDR

Ending text:
Para las condiciones del cambio establecidas, se concluye que:

Uses all applicable rules in data derivations.

RULES:

RULE NUMBER: 1

IF:
Change Type is Feature or Function Removed
and [CHANGE EFFECTIVE DATE] < [TODAY]
and [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]
and Change Sign_on Status is Signed by Customer
and Change Impact is Minimum
THEN:
Change is Valid - Probability=1

RULE NUMBER: 2

IF:
Change Type is Feature or Function Removed
and [CHANGE EFFECTIVE DATE] < [TODAY]
and [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]
and Change Sign_on Status is Signed by Customer
and Change Impact is Other
THEN:
Change is Invalid - Probability=1

RULE NUMBER: 3

IF:
Change Type is Feature or Function Removed
and [CHANGE EFFECTIVE DATE] < [TODAY]
and [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]
and Change Sign_on Status is Delayed
THEN:
Change is Valid - Probability=1

RULE NUMBER: 4

IF:
Change Type is Feature or Function Removed
and [CHANGE EFFECTIVE DATE] < [TODAY]
and [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]
and Change Sign_on Status is In Progress
THEN:
Change is Invalid - Probability=1

RULE NUMBER: 5

IF:
Change Type is Feature or Function Removed
and [CHANGE EFFECTIVE DATE] > [TODAY]
THEN:
[CHANGE NOTICE DATE] IS GIVEN THE VALUE [TODAY]
and Change Sign_on Status is In Progress
and Change is Valid - Probability=1

RULE NUMBER: 6

IF:
Change Type is Feature or Function Removed
and [CHANGE EFFECTIVE DATE] < [TODAY]
and [CHANGE NOTICE DATE] >= [CHANGE EFFECTIVE DATE]
THEN:
Change is Invalid - Probability=1

QUALIFIERS:

1 Change Type is

Feature or Function Removed
Other

Used in rule(s): 1 2 3 4 5 6

2 Change Sign_on Status is

Signed by Customer
Delayed
In Progress

Used in rule(s): 1 2 3 4 (5)

3 Change Impact is

Minimum
Other

Used in rule(s): 1 2

CHOICES:

1 Change is Valid

Used in rule(s): (1) (3) (5)

2 Change is Invalid

Used in rule(s): (2) (4) (6)

FORMULAS:

1 [CHANGE EFFECTIVE DATE] < [TODAY]

Used in rule(s):

2 [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]

Used in rule(s):

3 [CHANGE EFFECTIVE DATE] < [TODAY]

Used in rule(s):

4 [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]

Used in rule(s):

5 [CHANGE EFFECTIVE DATE] < [TODAY]

Used in rule(s): 4

6 [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]

Used in rule(s): 4

7 [CHANGE EFFECTIVE DATE] > [TODAY]

Used in rule(s): 5

8 [TODAY]

Used in rule(s): (5)

9 [CHANGE EFFECTIVE DATE] < [TODAY]

Used in rule(s):

10 [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]

Used in rule(s): 1

11 [CHANGE EFFECTIVE DATE] < [TODAY]

Used in rule(s): 2

12 [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]

Used in rule(s): 2

13 [CHANGE EFFECTIVE DATE] < [TODAY]

Used in rule(s): 3

14 [CHANGE NOTICE DATE] < [CHANGE EFFECTIVE DATE]

Used in rule(s): 3

15 [CHANGE EFFECTIVE DATE] < [TODAY]

Used in rule(s):

16 [CHANGE EFFECTIVE DATE] < [TODAY]

Used in rule(s): 1

17 [CHANGE EFFECTIVE DATE] < [TODAY]

Used in rule(s): 6

18 [CHANGE NOTICE DATE] >= [CHANGE EFFECTIVE DATE]

Used in rule(s): 6

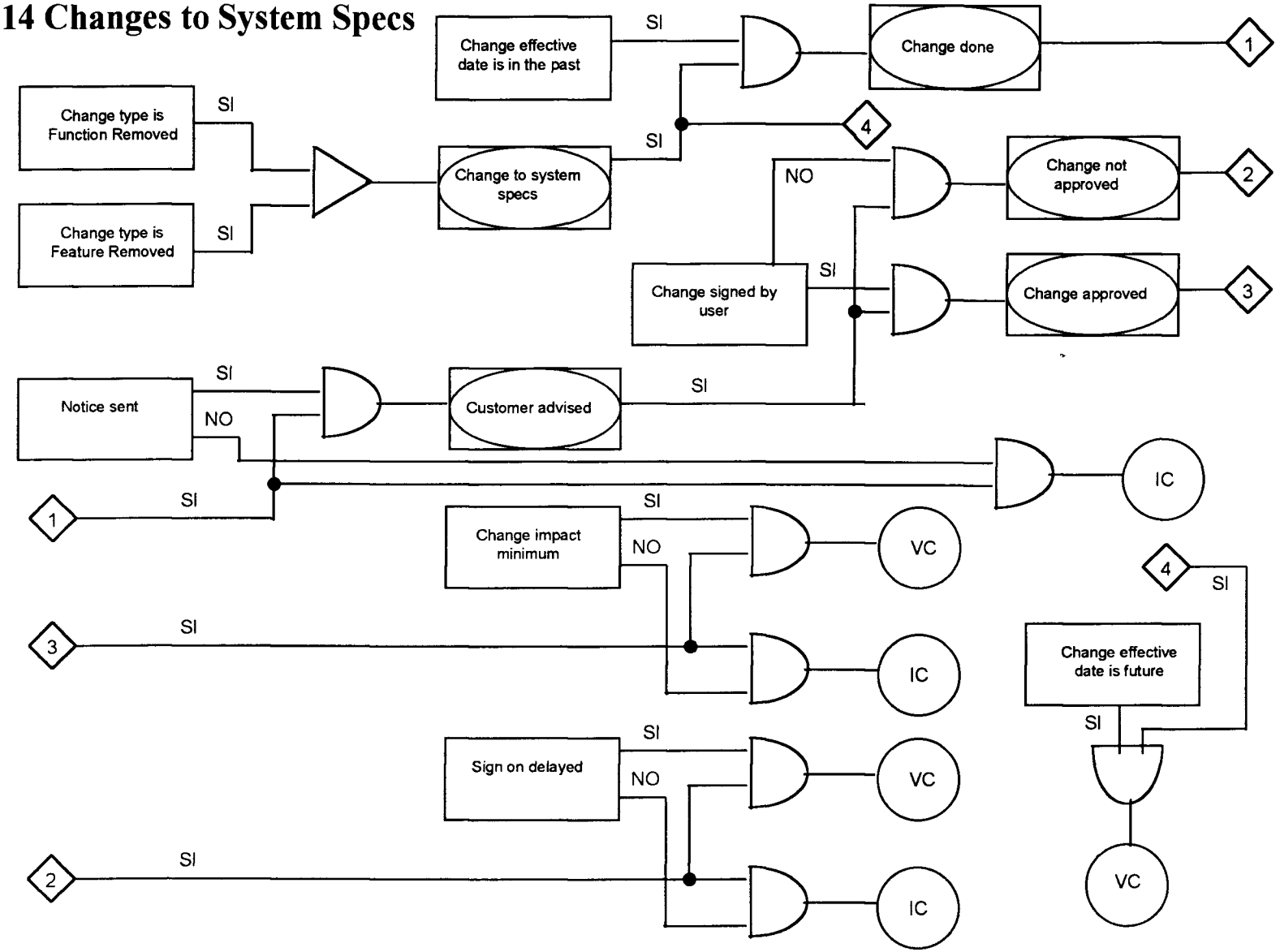
VARIABLES:

1 CHANGE EFFECTIVE DAT
Date the Change is/was effective
Numeric variable

2 TODAY
Today date
Numeric variable

3 CHANGE NOTICE DATE
Date of Change Notice
Numeric variable

2.14 Changes to System Specs



Implementación de una cláusula de "procedimiento" en modelo IMP

```
imp("o","n","Change to System Specs","pos","Function Removed","pos","Feature Removed",1)
imp("a","n","Change Done","pos","Change to System Specs","pos","Change effective date is past",1)
imp("a","n","Valid Change","pos","Change to System Specs","pos","Change effective date is future",1)
imp("a","n","Customer Adviced","pos","Change Done","pos","Notice Sent",1)
imp("a","n","Invalid Change","pos","Change Done","neg","Notice Sent",1)
imp("a","n","Change Approved","pos","Change Signed On by User","pos","Customer Adviced",1)
imp("a","n","Change not Approved","neg","Change Signed On by User","pos","Customer Adviced",1)
imp("a","n","Valid Change","pos","Change not Approved","pos","Sign On Delayed",1)
imp("a","n","Valid Change","pos","Change Approved","pos","Change Impact is minimum",1)
imp("a","n","Invalid Change","pos","Change Approved","neg","Change Impact is minimum",1)
hypothesis_node("Valid Change")
hypothesis_node("Invalid Change")
terminal_node("Function Removed")
terminal_node("Feature Removed")
terminal_node("Change effective date is past")
terminal_node("Change effective date is future")
terminal_node("Notice Sent")
terminal_node("Change Signed On by User")
terminal_node("Sign On Delayed")
terminal_node("Change Impact is minimum")
```

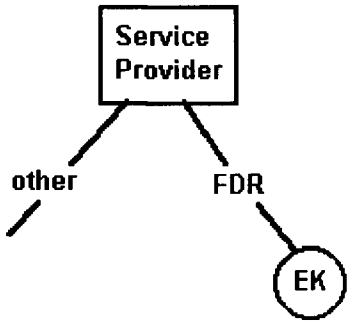
4.3.5 Comparativo de Representaciones

Debido al carácter binario de las instrucciones del modelo IMP, su representación requiere de instancias extras "de paso" para poder implementar *AND's* de más de 2 entradas. Esta limitante es propia del *shell*, más que de las redes de inferencia, por lo que, en cuanto a los *shells* utilizados, EXSYS es mucho más fácil de utilizar y es más claro en su sintaxis. Pero, de nuevo dejaremos a un lado los *shells* y atenderemos a la representación gráfica.

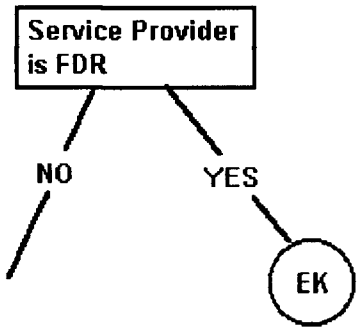
Por lo que respecta a los diagramas, es mejor utilizar árboles de decisión, en lugar de redes semánticas, para graficar sistemas basados en reglas. Los árboles de decisión permiten preguntar sobre la presencia o asignación de ciertos valores sobre variables simbólicas, lo que facilita el trabajar con atributos y, eventualmente, formar objetos.

Las redes de inferencia, por otro lado, requieren estructurarse en base a instancias VERDADERO o FALSO (SI o NO). Si bien es cierto que toda premisa devuelve, a fin de cuentas, un valor verdadero o falso para activar o no la regla, la naturaleza de las preguntas es ligeramente distinta en ambos diagramas.

Ejemplo: Fig. 4.1 Instancia en árboles de decisión



Ejemplo: Fig. 4.2 Instancia en redes de inferencia



La flexibilidad de los árboles de decisión es evidente ya que permiten transformaciones a otras representaciones de manera más sencilla. En el Anexo A se muestra la codificación de reglas en base a árboles de decisión con una orientación más a objetos, definiendo estructuras de datos pertenecientes a un grupo específico (clase).

Esto permite una gama más amplia de implementaciones en el computador, como lenguajes orientados a objetos, bases de datos relacionales, etc.

CAPÍTULO 5. CONCLUSIONES

El prototipo de SE implementado responde a preguntas sencillas sobre las cláusulas ejemplo codificadas de manera escueta pero correcta. Con el patrón de base de conocimientos sugerido puede construirse la totalidad del sistema tomando en consideración que éste puede ser voluminoso (por comparación al tamaño de las cuatro cláusulas ejemplo). Sin embargo, las oportunidades ofrecidas pueden llevar a un replanteamiento de estrategias de negocio más acorde con la normatividad del contrato, o en otro orden de importancia, a una saludable relación entre las partes conforme a los acuerdos firmados, que bien harían justificable el esfuerzo invertido en tal empresa.

En recientes fechas se ha dado una menor atención a los SE por parte de la comunidad de AI e inclusive se comenta que no se les debería dar tal nombre, sino que se deberían referir a ellos como "Sistemas de deducción basados en reglas", ya que los , erróneamente llamados, SE exhiben un comportamiento al resolver problemas, que se parece más al de los novatos humanos que al de los expertos. Esto, por una parte puede ayudar a desmitificar a los SE al quitarles la investidura de sistemas complejos e inteligentes, difíciles de entender, desarrollar y usar. En Bancomer no existe hoy en día un SE en producción para alguna aplicación ya sea importante o no. Se les considera, como en muchas empresas, juguetes experimentales de poca confiabilidad. Los temores no son infundados, después de todo, ya que se han hecho esfuerzos por desarrollar algunas aplicaciones de SE que no han resultado en buen término. Algunos proyectos demasiado ambiciosos para dar resultados palpables de manera rápida. Otros con un alcance muy limitado para ser útiles. A fin de cuentas no se ha cristalizado ningún trabajo tendiente a utilizar esta tecnología.

A manera de guía, y como corolario a este trabajo, me permito exponer algunos factores importantes para llevar a cabo un proyecto de SE. Los siguientes son los puntos importantes para desarrollar un sistema de este tipo, desde el enfoque de su ciclo de desarrollo e implementación.

Ciclo de Desarrollo de un Sistema Experto

1. Conceptualización
   - 1.1 Selección del problema
   - 1.2 Selección de Herramientas
   - 1.3 Equipo de desarrollo
2. Creación de un prototipo
3. Implantación
4. Adquisición de conocimientos
5. Desarrollo Incremental
6. Validación
7. Si aún no está completo repetir desde 3.
8. Entrega
9. Mantenimiento

También es necesario medir el alcance del proyecto a realizar. Algunos autores sugieren que el aventurarse en esta tecnología debe hacerse tomando en consideración los siguientes puntos: el primer SE debe ser fácil (chico, autónomo y accesible), seguro (que no tenga riesgos implícitos grandes), visible y rentable.

El enfoque de tipificar las cláusulas por su intención y la derivación de hipótesis a partir de éstas, pudiera parecer como generalizable a todo documento legal, o más aún, deseable que se pueda generalizar. Tal vez esto sea posible teniendo una gran cantidad de casos analizados de estatutos, leyes, códigos, de los cuales pudiéramos deducir si esto es factible. En tal caso, estaríamos en condiciones de formular bases de conocimientos de Derecho, acopladas a sistemas de acceso sencillo para cualquier usuario. Esto sería un gran apoyo(que no reemplazo) para todos los profesionales dedicados a las normas y procesos legales. Una base inteligente de consulta que haría, así quisiéramos pensar, más justas e imparciales, las relaciones y acuerdos entre personas físicas o morales.

Dejo pues la inquietud para todos aquellos desarrolladores e investigadores de *AI* el recopilar ejemplos de sistemas de este tipo que pudieran dar la pauta para establecer una metodología de desarrollo de Sistemas Expertos (o como algunos prefieren llamarlos, Sistemas de deducción basados en reglas) sobre aspectos legales.

ANEXO A.

Codificación de reglas en formato estructurado

Reglas Cláusula 1 Insolvency Event

R1_1

IF *PARTY.STATUS = DISSOLVED*
OR PARTY.STATUS = ADMITS_INABILITY
OR PARTY.STATUS = FAILS_TO _PAY
THEN INSOLVENCY_EVENT.STATUS = SET

R1_2

IF PARTY.STATUS <> DISSOLVED
AND PARTY.STATUS <> ADMITS_INABILITY
AND PARTY.STATUS <> FAILS_TO _PAY
AND PARTY.AGREEMENT_WITH_DEBTORS = SET
THEN INSOLVENCY_EVENT.STATUS = SET

R1_3

IF PARTY.STATUS <> DISSOLVED
AND PARTY.STATUS <> ADMITS_INABILITY
AND PARTY.STATUS <> FAILS_TO _PAY
AND PARTY.AGREEMENT_WITH_DEBTORS <> SET
AND (PARTY.ACTION = PETITION_BANKRUPTCY
OR PARTY.ACTION = SUSPENSION_OF_PAYMENTS
OR PARTY.ACTION = ASKED_RELIEF_OF_DEBTORS
OR PARTY.ACTION = CONSENT_CUSTODIAN_FOR_ITS_BUSINESS
)
THEN INSOLVENCY_EVENT.STATUS = SET

R1_4

IF PARTY.STATUS <> DISSOLVED
AND PARTY.STATUS <> ADMITS_INABILITY
AND PARTY.STATUS <> FAILS_TO _PAY
AND PARTY.AGREEMENT_WITH_DEBTORS <> SET
AND (PARTY.ACTION <> PETITION_BANKRUPTCY
AND PARTY.ACTION <> SUSPENSION_OF_PAYMENTS
AND PARTY.ACTION <> ASKED_RELIEF_OF_DEBTORS
AND PARTY.ACTION <> CONSENT_CUSTODIAN_FOR_ITS_BUSINESS
THEN INSOLVENCY_EVENT.STATUS = SET

Reglas Cláusula 3 Exclusivity

R3_1

IF SERVICE.PROVIDER = FDR
THEN SERVICE.EXCLUSIVITY = KEEPED

R3_2

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = NO
AND SERVICE.DISP_CAUSE = FORCE_MAJEURE
THEN SERVICE.EXCLUSIVITY = KEEPED

R3_3

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = NO
AND SERVICE.DISP_CAUSE <> FORCE_MAJEURE
AND (SERVICE.STATUS = REMOVED
OR SERVICE.STATUS = ADDED)
THEN SERVICE.EXCLUSIVITY = KEEPED

R3_4

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = NO
AND SERVICE.DISP_CAUSE = FORCE_MAJEURE
AND SERVICE.STATUS <> REMOVED
AND SERVICE.STATUS <> ADDED
THEN SERVICE.EXCLUSIVITY = BROKEN

R3_5

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = YES
AND (SERVICE.TYPE = TESTING
OR SERVICE.TYPE = BACKUP)
THEN SERVICE.EXCLUSIVITY = KEEPED

R3_6

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = YES
AND SERVICE TYPE <> TESTING
AND SERVICE.TYPE <> BACKUP
AND (SERVICE.STATUS = REMOVED
OR SERVICE.STATUS = ADDED)
THEN SERVICE.EXCLUSIVITY = KEEPED

R3_7

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = YES
AND SERVICE TYPE <> TESTING
AND SERVICE.TYPE <> BACKUP
AND SERVICE.STATUS = IN_CONTRACT
AND SERVICE.STORAGE_DATA > 6 (MONTHS)
AND SERVICE.SUMMARY_DATA > 2 (YEARS)
THEN SERVICE.EXCLUSIVITY = KEEPED

R3_8

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = YES
AND SERVICE TYPE <> TESTING
AND SERVICE.TYPE <> BACKUP
AND SERVICE.STATUS = IN_CONTRACT
AND SERVICE.STORAGE_DATA > 6 (MONTHS)
AND SERVICE.SUMMARY_DATA < 2 (YEARS)
THEN SERVICE.EXCLUSIVITY = BROKEN

R3_9

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = YES
AND SERVICE TYPE <> TESTING
AND SERVICE.TYPE <> BACKUP
AND SERVICE.STATUS = IN_CONTRACT
AND SERVICE.STORAGE_DATA < 6 (MONTHS)
THEN SERVICE.EXCLUSIVITY = BROKEN

R3_10

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = YES
AND SERVICE.TYPE = ENHANCEMENT
THEN SERVICE.EXCLUSIVITY = BROKEN

R3_11

IF SERVICE.PROVIDER <> FDR
AND SERVICE.FDR_AVAILABLE = NO
AND SERVICE.TYPE = ENHANCEMENT
THEN SERVICE.EXCLUSIVITY = BROKEN

Reglas Cláusula 2.7a EDP Audit Provisions

R2.7a_1

IF AUDIT.LAST_DATE < 1 (YEAR)
THEN CUSTOMER.ACTION = DONT_REQUEST_AUDIT

R2.7a_2

IF AUDIT.LAST_DATE > 1 (YEAR)
AND AUDIT.STATUS = IN_PROGRESS
THEN CUSTOMER.ACTION = WAIT_AUDIT_RESULTS

R2.7a_3

IF AUDIT.LAST_DATE > 1 (YEAR)
AND AUDIT.STATUS = DONE
THEN CUSTOMER.ACTION = REQUEST_AUDIT_COPY

R2.7a_4

IF AUDIT.LAST_DATE > 1 (YEAR)
AND AUDIT.STATUS = NOT_STARTED
THEN CUSTOMER.ACTION = REQUEST_AUDIT

R2.7a_5

IF AUDIT.LAST_DATE > 1 (YEAR)
AND AUDIT.STATUS = REQUESTED
AND.AUDIT.FDR = AVAILABLE
THEN CUSTOMER.ACTION = WAIT_AUDIT_RESULTS

R2.7a_6

IF AUDIT.LAST_DATE > 1 (YEAR)
AND AUDIT.STATUS = REQUESTED
AND.AUDIT.FDR = UNAVAILABLE
AND AUDIT.NOTICE = NOT_SENT
THEN CUSTOMER.ACTION = SEND_NOTIFICATION_OF_THIRD_PARTY
_AUDIT_PROVIDER

R2.7a_7

IF AUDIT.LAST_DATE > 1 (YEAR)
AND AUDIT.STATUS = REQUESTED
AND.AUDIT.FDR = UNAVAILABLE
AND AUDIT.NOTICE = SENT
AND AUDIT.PROVIDER_SELECTOR <> CUSTOMER
THEN CUSTOMER.ACTION = SELECT_THIRD_PARTY_AUDIT_PROVIDER

R2.7a_8

IF AUDIT.LAST_DATE > 1 (YEAR)
AND AUDIT.STATUS = REQUESTED
AND.AUDIT.FDR = UNAVAILABLE
AND AUDIT.NOTICE = SENT
AND AUDIT.PROVIDER_SELECTOR = CUSTOMER
AND AUDIT.PROVIDER <> FDR_ACCEPTED
THEN CUSTOMER.ACTION = SELECT_AUDIT_PROVIDER

R2.7a_9

IF AUDIT.LAST_DATE > 1 (YEAR)
AND AUDIT.STATUS = REQUESTED
AND.AUDIT.FDR = UNAVAILABLE
AND AUDIT.NOTICE = SENT
AND AUDIT.PROVIDER_SELECTOR = CUSTOMER
AND AUDIT.PROVIDER = FDR_ACCEPTED
THEN CUSTOMER.ACTION = WAIT_AUDIT_RESULTS

Reglas Cláusula 2.14 *Changes to System Specs*

R2.14_1

IF (CHANGE.TYPE = FEATURE_REMOVED
OR CHANGE.TYPE = FUNCTION_REMOVED)
AND CHANGE.EFFECTIVE_DATE = PAST
AND CHANGE.NOTICE_DATE < CHANGE.EFFECTIVE_DATE
AND CHANGE.SIG_ON_STATUS = SIGNED_BY_CUSTOMER
AND CHANGE.IMPACT = MINIMUM
THEN CHANGE.VALIDITY = TRUE

R2.14_2

IF (CHANGE.TYPE = FEATURE_REMOVED
OR CHANGE.TYPE = FUNCTION_REMOVED)
AND CHANGE.EFFECTIVE_DATE = PAST
AND CHANGE.NOTICE_DATE < CHANGE.EFFECTIVE_DATE
AND CHANGE.SIG_ON_STATUS = SIGNED_BY_CUSTOMER
AND CHANGE.IMPACT <> MINIMUM
THEN CHANGE.VALIDITY = FALSE

R2.14_3

IF (CHANGE.TYPE = FEATURE_REMOVED
OR CHANGE.TYPE = FUNCTION_REMOVED)
AND CHANGE.EFFECTIVE_DATE = PAST
AND CHANGE.NOTICE_DATE < CHANGE.EFFECTIVE_DATE
AND CHANGE.SIG_ON_STATUS = DELAYED
THEN CHANGE.VALIDITY = TRUE

R2.14_4

IF (CHANGE.TYPE = FEATURE_REMOVED
OR CHANGE.TYPE = FUNCTION_REMOVED)
AND CHANGE.EFFECTIVE_DATE = PAST
AND CHANGE.NOTICE_DATE < CHANGE.EFFECTIVE_DATE
AND CHANGE.SIG_ON_STATUS = IN_PROGRESS
THEN CHANGE.VALIDITY = FALSE

R2.14_5

IF (CHANGE.TYPE = FEATURE_REMOVED
OR CHANGE.TYPE = FUNCTION_REMOVED)
AND CHANGE.EFFECTIVE_DATE = FUTURE
AND CHANGE.NOTICE_DATE = NOW
AND CHANGE.SIG_ON_STATUS = IN_PROGRESS
THEN CHANGE.VALIDITY = TRUE

GLOSARIO DE TÉRMINOS

A

AI (Artificial Intelligence): Inteligencia Artificial es el estudio de las computaciones que permiten percibir, razonar y actuar.

B

Backward Chaining (Encadenamiento hacia atrás): En este procedimiento, generalmente se empieza con una conclusión en la que estamos interesados. Una que no es un hecho explícito, es decir, que no está en el almacén cuando iniciamos el sistema. Lo que queremos encontrar es ¿está este hecho implicado por los otros hechos y reglas que conocemos?, ¿hay algún patrón de razonamiento que podamos descubrir que establecerá este hecho como verdadero? La forma en que un motor de inferencias puede investigar esto es identificando todas las reglas que tienen el hecho en cuestión como conclusión. Entonces busca las premisas en todas las reglas. Quizá estos hechos estén en el almacén de hechos verdaderos. En este caso, hemos determinado que el hecho bajo investigación es verdadero. Pero ¿qué pasa si ninguna de las reglas pueden ser usadas directamente para determinar la conclusión bajo investigación debido a que las premisas en estas reglas también necesitan establecerse como verdaderas? Entonces debe trabajar hacia atrás y tratar de determinar la validez de todas las premisas en las reglas que pueden ser usadas para establecer la conclusión final.

D

DSS (Decision Support Systems): Sistemas que ayudan a analizar los efectos de futuras decisiones, políticas o normas de conducta; y casi siempre involucran un modelo predictivo explícito. Los *DSS* tienen una orientación más hacia modelos que hacia datos.
EXSYS: *Shell* de SE que permite insertar conocimiento en forma de reglas de producción. Contiene mecanismos de solicitud de explicación a una pregunta (*WHY*) y explicación de una respuesta (*HOW*).

F

FDR (First Data Resources): Procesador de tarjetas de crédito encargado del proceso de Bancomer.

Forward Chaining (Encadenamiento hacia adelante): *Forward Chaining* significa usar las reglas para deducir nuevos hechos, hechos que fueron implícitos antes, pero que pueden hacerse explícitos a través de la operación de las reglas. El patrón típico de la operación de Forward Chaining es el siguiente: El motor de inferencias cicla por todas las reglas y examina cada una a la vez para ver si la información en el lado izquierdo es verdadera. Si lo es, el motor de inferencias agrega el hecho (o conclusión) del lado derecho de la regla a su almacén de hechos verdaderos.

Entonces continúa con la siguiente regla y trata de hacer lo mismo. Cuando acaba con todas las reglas, comienza otra vez.

H

Heurística: Las reglas heurísticas, o heurística para abreviar, son reglas que son desarrolladas a través de intuición, experiencia y juicio. Típicamente no representan nuestro conocimiento del diseño de, o las interrelaciones entre, un sistema. En su lugar ellas representan guías a través de las cuales un sistema puede ser operado. La heurística no necesariamente resulta en el mejor, u óptimo, resultado

I

IE (Inference Engine): Motor de Inferencias: La máquina de inferencias es empleada sobre la base de conocimientos durante una consulta con un sistema experto. Ejecuta dos tareas primordiales. Primero, examina el status de la base de conocimientos y la memoria de trabajo para determinar qué hechos son conocidos a un tiempo dado, y adiciona cualquier nuevo hecho que esté disponible. Segundo, provee el control de la sesión al determinar el orden en el cual las inferencias son hechas. Una designación alternativa para la máquina de inferencias, y quizá más apropiada es la de Procesador de Conocimiento.
El motor de inferencias sirve para mezclar los hechos con reglas para desarrollar, o inferir, nuevos hechos.

IMP: *Shell* de SE basado en *Prolog*.

K

KB (Knowledge Base): Base de Conocimiento: Componente de un SE encargado de determinar, por medio de una serie de reglas heurísticas, la conclusión a la que llegará un experto. Contiene una descripción, o modelo, de lo que sabemos para llegar a la solución de un problema dado. El motor de inferencias contiene una descripción de lo que hacemos para desarrollar la solución. Mientras que la Base de Conocimientos cambia de dominio a dominio, la máquina de inferencias es la misma.

O

OAV (Tripletas OAV - Objeto, Valor, Atributo): Forma de representación del conocimiento en el que cada tripleta hace referencia a alguna entidad específica u objeto conteniendo un conjunto de atributos que sirven para caracterizar al objeto.

S

SE (Sistema Experto): Es un programa de computadora que exhibe, dentro de un dominio específico, un grado de *expertise* al resolver un problema, comparable a un experto humano

REFERENCIAS BIBLIOGRÁFICAS

[Benfer 89] Benfer, A. Robert et. al. Adquisición de Conocimiento en los Andes Peruanos. En *AI Expert Magazine* - Nov. 89

[Bringmann, 92] Bringmann, M. W. y E. Frederick. *A Semantic Network Representation of Personal Construct Systems*. En: IEEE *Transactions on Systems, Man & Cybernetics* Vol. 22 Issue 5. pp. 1161-1168

[Copi, 92] Copi, Irving M. y Cohen, Carl. *Introduction to Logic*

[Cuena, 85] Cuena, José. y Fernández, Gregorio. Inteligencia Artificial: Sistemas Expertos. Alianza Editorial. 1985

[Davis, 90] Davis, Ernest. *Representations of Common Sense Knowledge*. Morgan Kaufmann Pub. 1990

[Ignizio, 91] Ignizio, James P. *Introduction to Expert Systems - The Development and Implementation of Rule-Based Expert Systems*. McGraw Hill. 1991

[Marcellus, 89] Marcellus, Daniel H. *Expert System Programming in TURBO PROLOG*. Prentice Hall. 1989

[Nebendahl, 91] Nebendahl, Dieter. Sistemas Expertos. Boixareau Editores. 1991

[Popescu, 92] Popescu, Ilié. *A Relational Model for Knowledge Representation in Expert Systems*. En: *Journal of Systems Software*. Vol. 18. Iss:2. 1992 pp. 147-155

[Rucker, 87] Rucker, Rudy. *Mind Tools - The 5 Levels of Mathematical Reality*. Houghton Mifflin. 1987

[Todd, 92] Todd, B.S. et. al. *Formal Specification of a Rule-Based Expert System*. *The Programming Research Group, Oxford University*. pp. 333-340

[Winston, 92]Winston, H. Patrick. Inteligencia Artificial. Addison-Wesley Iberoamericana. 1994