# UNIDAD III
# REPRESENTACIÒN DE PROBLEMAS EN INTELIGENCIA ARTIFICIAL

## 3.1 Métodos de Inteligencia Artificial para resolver problemas

### MÉTODOS DE RESOLUCIÓN DE PROBLEMAS

La inteligencia artificial es una parte de la informática que concierne también al diseño de sistemas inteligentes para el computador, esto es, sistemas que presentan muchas características asociadas con el comportamiento humano inteligente.

A través de diferentes estudios formales, se cree que la percepción que se encuentra en la naturaleza de la mente, por lo que puede ser adquirida estudiando la lógica de los programas que ejecuta.

Tanto si estas técnicas promueves un mejor conocimiento de la mente como si no, es bastante evidente que los desarrollos conducen a una nueva tecnología inteligente que puede tener efectos sorprendentes en nuestra sociedad.

Sistemas experimentales de inteligencia han sido ya generados de una manera interesada y entusiasta en la industria y están siendo desarrollados comercialmente. Estos sistemas experimentales incluyen programas que:

1. Resuelven algunos problemas complicados en disciplinas como química, biología, geología, ingeniería y medicina que habitualmente precisan de expertos humanos con diferentes niveles de especialización.
2. Dispositivos que manipulan robots para resolver algunas tareas usuales, repetitivas y relacionadas con motores que reaccionan ante el estímulo de sensores.
3. Responden cuestionen expresadas en dialectos simples, derivados de lenguas como: inglés, francés, japonés o cualquier otro lenguaje natural.

Estos puntos indican que, normalmente, los programas de inteligencia artificial juegan un papel muy importante en el entorno informático dentro de la vida humana (un entorno que ha cambiado, en nuestro tiempo, de ser ajeno a nosotros a formar parte de nuestra convivencia normal y que, si las expectativas acerca de los costes relacionados con la informática y su potencia son correctas, es seguro que pasara de ser inusual a esencial).

## RESOLUCION DE PROBLEMAS

Técnicas que tan pronto investigan varios movimientos futuros como dividen problemas difíciles en subproblemas sencillos, se incluyen en las técnicas fundamentales de inteligencia artificial de búsqueda y reducción de problemas.

Otros programas resolutivos de problemas que integran formulas matemáticas, simbólicamente, han logrado niveles muy altos de rendimiento, y son usados por científicos e ingenieros de todo el mundo.

Otra cuestión básica abierta abarca la conceptualización original de un problema, llamado en la inteligencia artificial la elección de la representación de un problema.

Los problemas de inteligencia artificial hasta ahora pueden ser expuestos como lo que se cree que se necesita para que el problema sea resuelto.

## TRES PODEROSOS MÉTODOS DE RESOLUCIÓN DE PROBLEMAS

Generación y Prueba, Análisis de Medios y Metas y Reducción del Problema. Dos nuevas representaciones que se pueden considerar como casos especiales de la representación de red semántica.

Una es la representación de espacio de estado que se da a conocer al hablar del análisis de medios y metas, y la otra es el árbol de metas que se menciona en la reducción del problema.

## MÉTODO DE GENERACIÓN Y PRUEBA

Quienes resuelven problemas y han adoptado el paradigma de la generación y prueba usan dos módulos básicos. Uno de ellos, el generador, enumera las soluciones posibles. El otro, el probador, evalúa cada solución propuesta ya sea aceptándola o rechazándola.

El generador puede producir todas las soluciones posibles antes de que el probador inicie sus funciones, la generación y la prueba se intercale. La acción puede determinar cuando se encuentra una solución aceptable, continuar hasta que se encuentre un numero satisfactorio de soluciones, o bien hasta que se hallen todas las soluciones posibles.

Para realizar la generación de prueba:

- Hasta que se encuentre una solución satisfactoria o no se puede generar más soluciones posibles.
  - ✓ Genere una solución posible
  - ✓ Pruebe la solución posible

- Si se halla un solución aceptable, menciónela de otro modo notifique el fracaso.

*Los sistemas de generación y prueba a menudo realizan identificaciones.*

El paradigma de generación y prueba se utiliza con más frecuencia para resolver problemas de identificación. En los problemas de identificación se dice que un generador produce una hipótesis.

Para usar el paradigma de generación y prueba con el objeto de identificar, digamos un árbol, puede tomar un libro sobre árboles, recorrer pagina por página y detenerse cuando encuentre un dibujo del árbol que se parezca al que quiere identificar. Recorrer las páginas del libro es el proceso de generación; parear los dibujos con el árbol es el procedimiento de prueba.

*Los buenos generadores son completos, no redundantes e informados, estos contienen tres propiedades:*

- Son completos: en algún momento producen todas las soluciones posibles.

- No son redundantes: nunca comprometen la eficiencia al proponer la misma solución dos veces.

- Están informados: usan información que limita las posibilidades restringiendo las soluciones que proponen de acuerdo con ella.

La capacidad de información es importante, ya que de otro modo suele haber un gran número de soluciones que intentar. Considere el ejemplo de la identificación de árboles. Si estamos en invierno y el árbol que desea identificar no tiene hojas no se molestaría en revisar la sección de coníferas del libro.

De manera similar, si un ladrón sabe, de algún modo, que todos los números que hay en la combinación de una caja fuerte son primos que van de 0 a 99, entonces puede limitarse a 15625 números, con lo cual podría abrir la caja en menos de 2 días, en promedio, en lugar de 16 semanas.

## 3.2 Representación de espacio de estado

**REPRESENTACIÓN DE ESPACIO-ESTADO**

Un estado es un conjunto de condiciones o valores que describen un sistema en una punta específico durante el proceso. Una representación Espacio-Estado es aquella en la que los operadores usan un nuevo estado, y solo uno, en la base de datos, cada vez que se emplean.

Las representaciones Espacio-Estado, que utilizan varios métodos de búsqueda a través de la base de datos para alcanzar soluciones, han sido usadas por juegos de ajedrez, sistemas para encontrar rutas y problemas que incluyan muchos operadores y muchos posibles estados.

La búsqueda de un camino de solución a través del Espacio-Estado se puede ilustrar gráficamente mediante nodos y enlaces de conexión.

Cada nodo representa un estado del sistema y los enlaces representan la acción de un operador para cambiar el sistema de un estado a otro. Los nodos pueden tener punteros que señalen al nodo origen de forma que cuando se alcance la solución podamos conocer su camino.

En algunos métodos de búsqueda los punteros también se usan para marcar los caminos "falsos".

Los gráficos son dibujos generalmente como árboles o redes. Un árbol es un gráfico que empieza por la cima con un nodo raíz, de manera que los nodos de un nivel tienen un solo origen (un solo padre).

Esta estructura presenta solamente un camino desde el nodo raíz a otro nodo cualquiera. Los enlaces o conexiones entre nodos de un árbol se llaman ramas (ramificaciones). Los nodos en la parte inferior del árbol se denominan nodos terminales (hojas). Los nodos en los gráficos de árbol se denominan algunas veces padres e hijos (descendientes).

Hay potencialmente un árbol de un espacio-estado que representa cada posible estado del sistema y otro árbol mas limitado del camino que puede ser construido si el programa busca la respuesta eficientemente.

Puede ser importante encontrar un modo eficiente de búsqueda porque el espacio potencial de búsqueda puede ser infinito o, al menos, tan grande que haga el proceso no viable.

Por lo que la Representación de espacio-estado:

- Define Un Espacio De Estados (Espacio Con Todas Las Posibles Soluciones Potenciales Implícita / Explícitamente Enumerado
- Especifica Los Estados Iniciales O Situaciones De Donde Empezar
- Especifica Los Estados Finales (Metas) O Aquellos Reconocidos Como Soluciones
- Especifica Las Reglas Que Definen Las Acciones U Operaciones Disponibles Para Moverse o ir de un estado a otro dentro del espacio del problema

En este contexto el proceso de solución de problemas trata de encontrar una secuencia de operaciones que transformen al estado inicial en uno final

En la práctica se necesita seguir una estrategia de búsqueda

## 3.3 Métodos de búsqueda en un espacio de estado

Hay varias técnicas para la búsqueda de una posible solución:

- Primero en profundidad
- Primero en amplitud
- Escalando la colina (Hill Climbing).
- Menor coste

### *EVALUACIÓN DE UNA BÚSQUEDA*

La evaluación de una eficiencia de una técnica de una búsqueda, puede ser muy complicada. Esta evaluación se lleva gran parte de la investigación en la IA. Sin embargo, para el propósito de esta discusión hay dos medidas elementales que son importantes.

1. La rapidez con que se encuentra una solución.
2. La calidad de la solución

Hay varios tipos de problemas para los cuales lo principal es encontrar una solución con el mínimo esfuerzo. Otras situaciones lo más importante es que la situación sea lo más aproximada a una solución optima.

Tanto la longitud del camino para la solución el numero real de nodos que atraviesa, determina la velocidad de la búsqueda.

Es importante entender la diferencia entre encontrar una solución óptima y una solución buena.

La diferencia radica en el hecho de encontrar una solución optima a menudo nos exige una búsqueda exhaustiva por que puede que sea el único camino para determinar si hemos o no la mejor solución.

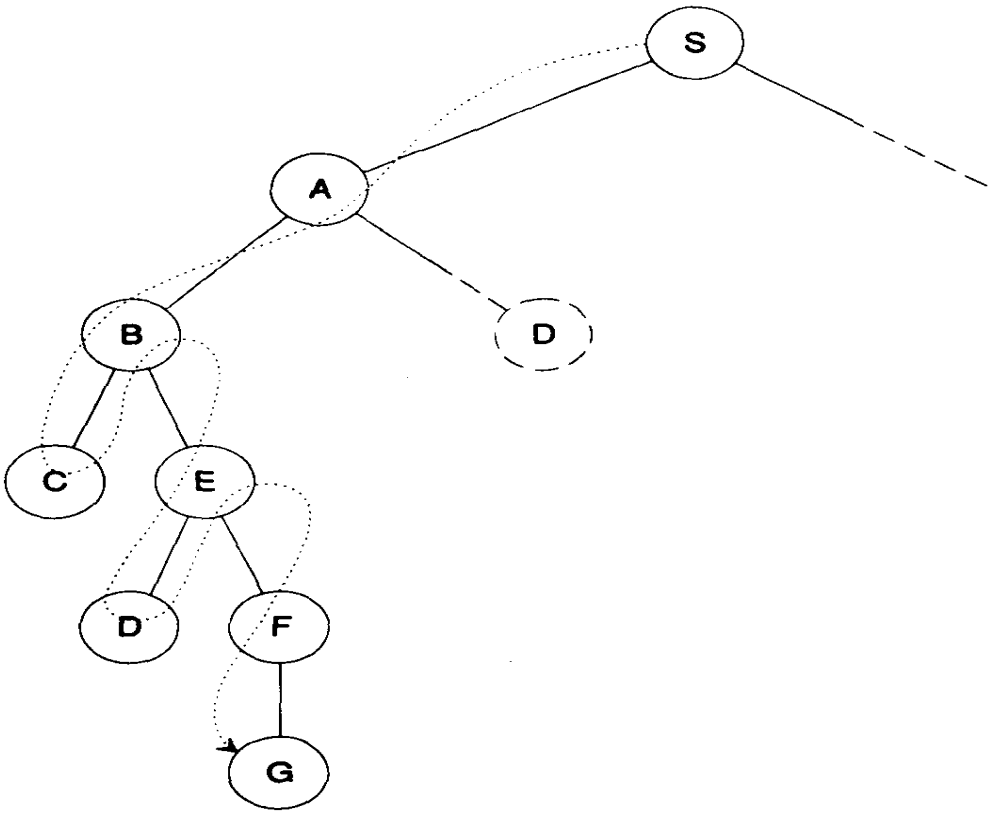
Puede llevar acabo el retro seguimiento usando retinas recursivas y una pila de vuelta atrás. Virtualmente todas las situaciones de retro seguimiento operan como una pila, la primera en entrar es la última en salir.

# TÉCNICA DE BÚSQUEDA DEL PRIMERO EN PROFUNDIDAD

Una búsqueda del primero en profundidad explora cualquier cambio posible hasta sus conclusiones o meta. Si alcanza un nodo terminal, entonces sube un nivel, va hacia la derecha y luego hacia la izquierda hasta encontrar la meta, o bien un nodo terminal. Repite este proceso hasta que encuentre la meta o examine el último nodo en el espacio de búsqueda.

Para llevar a cabo una búsqueda en profundidad:

1. Forme una cola de un elemento consistente en una trayectoria de longitud cero que contenga solo al nodo raíz.
2. Hasta que la primera trayectoria de la fila termine en el nodo meta o se vacíe la cola
   - Elimine la primera trayectoria de la cola; cree nuevas trayectorias extendiendo el primer paso a todos los vecinos del nodo terminal
   - Rechace todas las trayectorias nuevas con ciclos
   - Agregue las nuevas trayectorias, si las hay, al frente de la cola.
3. Si el nodo meta se alcanza, mencione que hubo éxito; de otro modo, notifique el fracaso.




**Ejemplo de Búsqueda en Profundidad**

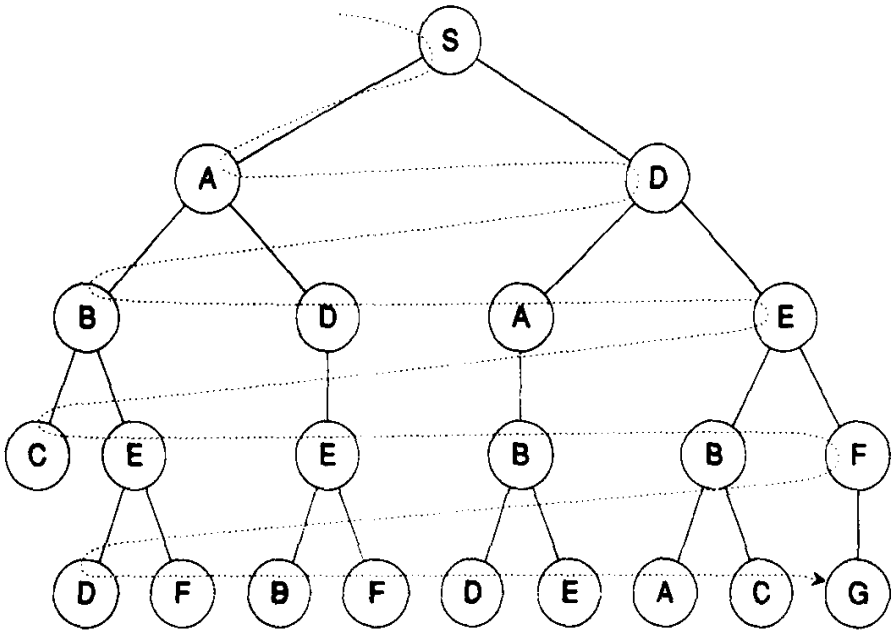
Se escoge una alternativa y se le sigue en cada nodo hasta llegar a la meta o a un nodo en el que ya no es posible seguir hacia abajo, la búsqueda se reinicia en el antecesor más cercano que posea hijos sin explorar.

### *TÉCNICA DE BÚSQUEDA DEL PRIMERO EN AMPLITUD*

La técnica de búsqueda del primero en amplitud es la opuesta a la de profundidad. La técnica de búsqueda del primero en amplitud comprueba cada nodo sobre el mismo nivel antes de proceder analizar un nivel mas profundo.

Para efectuar una búsqueda en amplitud:

1. Forme una cola de un solo elemento consistente en una trayectoria de longitud cero que contenga solo al nodo raíz.
2. Hasta que la primera trayectoria de la cola concluya en el nodo meta o se vacíe la cola.
   - Elimine la primera trayectoria de la cola; cree nuevas trayectorias extendiendo el primer paso a todos los vecinos del nodo terminal.
   - Rechace las trayectorias nuevas con ciclos.
   - Agregue las nuevas trayectorias, si las hay, al final de la cola.
3. Si se halla el nodo meta, mencione que hubo éxito; si no, notifique el fracaso.




**Ejemplo de Búsqueda en amplitud. El movimiento hacia abajo se efectúa nivel por nivel, hasta que se alcanza la meta.**

### *ADICIÓN DE INFORMACIÓN HEURÍSTICA*

Llegando a este punto halla supuesto que las búsquedas en profundidad y en anchura son rutinas (ciegas) busca su solución solamente del movimiento de una meta a otra sin uso de una “suposición inteligente “.

La única manera de alcázar tal tipo de búsqueda es la adición de capacidades heurísticas.

Las heurísticas son reglas simples que habilitan que la posibilidad de que una búsqueda proceda en una dirección correcta.

## *TÉCNICA DE BÚSQUEDA DE ASCENSO DE LA COLINA*

El algoritmo de ascenso de la colina escoge como próximo paso el nodo que aparezca en el lugar más cercano a la meta.

El algoritmo deriva de su nombre de la analogía de un excursionista perdido en la oscuridad en medio de una montaña, incluso en la oscuridad, el excursionista sabe que cualquier paso hacia arriba le sitúa en la dirección correcta.

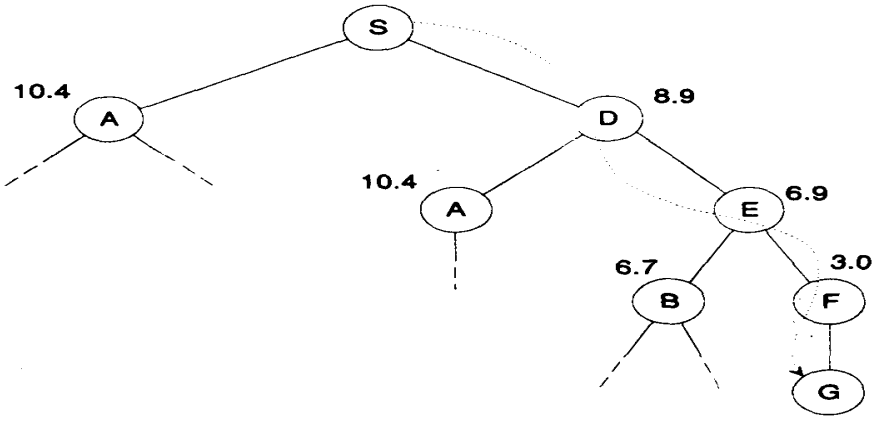
Para efectuar una búsqueda de ascenso de colina:

1. Forme una cola de un elemento consistente en una trayectoria de longitud cero que contenga solo al nodo raíz.

2. Hasta que la primera trayectoria de la cola termine en el nodo meta o la cola se vacíe.

   - Elimine la primera trayectoria de la cola; cree nuevas trayectorias extendiendo el primer paso a todos los vecinos del nodo terminal.
   - Rechace todas las trayectorias nuevas con ciclos.
   - Ordene las trayectorias nuevas, si las hay, según las distancias estimadas entre sus nodos terminales y la meta.
   - Agregue las nuevas trayectorias, si las hay, al frente de la fila.

3. Si el nodo meta se encuentra, mencione que hubo éxito; de otro modo, notifique el fracaso.




**Ejemplo de Ascenso de Colina**

Aunque sencillo, el ascenso de colina implica varios problemas. Estos problemas afloran mas cuando el ascenso de colina se usa para optimizar parámetros, como en los siguientes ejemplos:

- Al entrar en un cuarto, se da cuenta que la temperatura no es confortable; entonces, decide ajustar el termostato.
- La imagen de su televisor se ha deteriorado con el tiempo; entonces, decide ajustar los controles de color, tinte y brillantez para obtener **una** mejor imagen.
- Usted se encuentra escalando una montaña cuando se presenta una densa niebla. No dispone de un mapa o un sendero que seguir, pero tiene una brújula, un altímetro y la determinación de llegar a la cima.

Cada uno de estos problemas se atiene a una abstracción en la que existen *parámetros ajustables* y una *cantidad medida* que le indica la calidad o desempeño asociado con cualquier disposición particular de los parámetros ajustables.

De manera mas general, para efectuar un ascenso de colina orientado a parámetros, usted realiza un ajuste de un paso, hacia arriba y hacia abajo, a cada valor de parámetro; luego se mueve hacia la mejor de las alternativas resultantes de acuerdo con la medida apropiada de calidad o desempeño; y, por ultimo, repite el proceso hasta que encuentra una combinación de valores de parámetros que produce una mejor calidad o desempeño que todas las alternativas cercanas.

Pero se observa que puede tener serios problemas con el ascenso de colina orientado a los parámetros:

- **El problema de la falda de colina** ocurre siempre que hay picos secundarios, como en el ejemplo superior de la figura 4.7. Los picos secundarios atraen al procedimiento de ascenso de colina como si se tratara de imanes. Se encuentra un punto óptimo, pero se trata de un máximo local, no de un máximo global, y el usuario queda con una falsa impresión de seguridad o de haber terminado.
- **El problema de la meseta** se presenta cuando hay un área casi plana que separa los picos. En casos extremos, los picos pueden verse como postes de teléfono en un campo de fútbol. La operación de mejoramiento local fracasa por completo, ya que para todas las posiciones —excepto un número muy pequeño— todas las pruebas de paso normal dejan intacta la medición de calidad.
- **El problema del reborde** es más sutil y, en consecuencia, más frustrante. Suponga que se encuentra parado en lo que parece un contorno tan afilado como una navaja, que generalmente va de noreste a suroeste. Un mapa del contorno muestra que cada paso normal le lleva hacia abajo, aun cuando no se encuentra en ninguna especie de máximo local o global. Puede ser de ayuda aumentar el número de direcciones que se utilizan en los pasos de prueba.

Entre estos problemas, el problema de la falda de colina es particularmente molesto, sobre todo al aumentar el número de dimensiones de parámetro. Cuando usted alcanza un punto desde el cual todos los pasos conducen hacia abajo, se puede retirar a un punto escogido con anterioridad y hacer otra cosa, como lo prescribe el ascenso de colina, pero puede haber millones de trayectorias que lleven al mismo máximo local. Si este es el caso, usted se encontrara realmente "atascado" si se atiene al ascenso de colina ordinario.

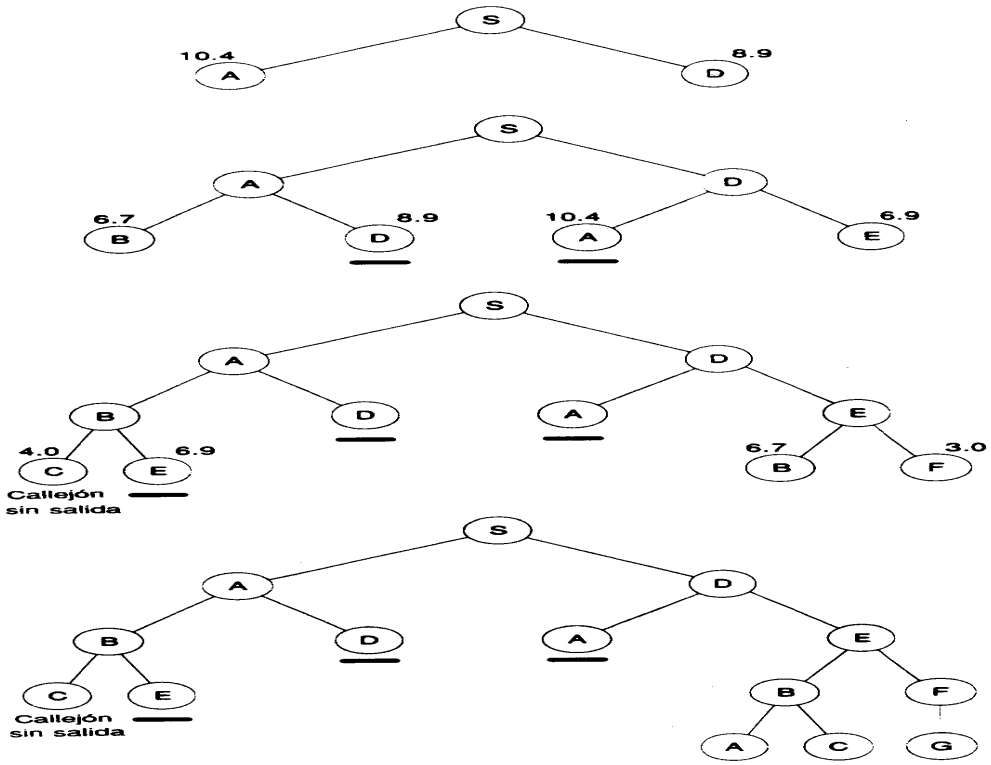
Por consiguiente, tal vez desee realizar un poco de búsqueda no determinista cuando detecte un máximo local. La razón para usar esta estrategia es que un numero aleatorio de pasos, de tamaño aleatorio, en direcciones al azar, puede formar un escudo contra la atracción magnética del máximo local, lo suficientemente grande para que usted pueda escapar.

## *TÉCNICA DE BÚSQUEDA DEL MENOR COSTE*

Opuesta a la técnica de la escalada de la colina es la de menor coste. Ejemplo si se encontrara a la mitad de una calle en lo alto de una gran montaña llevando puesto unos patines tendrá la absoluta certeza de que es mas fácil bajar que subir, por tanto, en el menor coste toma el camino de menor esfuerzo.

## *TECNICA DE BUSQUEDA EN HAZ*

La búsqueda en haz es parecida a la búsqueda en amplitud en cuanto a que avanza nivel por nivel. Sin embargo, a diferencia de esta, la búsqueda en haz se mueve hacia abajo solo a través de los mejores *w* nodos de cada nivel; los otros nodos se ignoran. En consecuencia, el numero de nodos explorados se mantiene manejable, aun cuando haya gran cantidad de ramificaciones y la búsqueda sea profunda, Siempre que se usa la búsqueda en haz, solo existen w nodos en consideración a cualquier profundidad, y no el inmenso numero exponencial de nodos con los que tiene que tratar siempre que utiliza la búsqueda en amplitud. En la figura 9 se ilustra como la búsqueda de haz luminoso abordaría el problema del cruce de mapa.




**Ejemplo de búsqueda en Haz**

## *ELECCIÓN DE UNA TÉCNICA DE BÚSQUEDA*

Las técnicas heurísticas tienden a funcionar mejor en las búsquedas a ciegas. Sin embargo, no es siempre posible utilizar una técnica heurística porque puede que no halla suficiente información disponible que le permite considerar la posibilidad de que el próximo nodo este en el camino hacia la meta, por tanto, puede dividir las reglas para la elección de un método de búsqueda en dos categorías:

- La primera para aquellos problemas que pueden utilizar una búsqueda heurística.
- La segunda para aquellos problemas que no pueden hacerlo.

Si no pueden aplicar una heurística a un problema, debería de elegir una búsqueda del primero en profundidad porque es generalmente la mejor.

La única excepción de esta regla esta en el caso de que posea una información que le demuestre que una búsqueda del primero en anchura funcionara mejor.

La elección entre la escala de la colina y el menor coste depende de los obstáculos que intente minimizar o exagerar.

## *ENCONTRANDO MULTIPLES SOLUCIONES*

Algunas veces, debe encontrar diversas soluciones a un mismo problema. Sin embargo esta situación no es similar a la de encontrar todas las soluciones, lo que ocurre en una búsqueda exhaustiva.

Por ejemplo, imagine que esta tratando de diseñar su casa (casa de ensueño); usted prueba varios planos para encontrar uno que le guste; lo que usted no hace es proyectar todas las casas. Por tanto las soluciones múltiples pueden ofrecerle distintas opciones que le ayuden a encontrar la mejor solución para su problema.

A pesar de que hay muchos métodos de generación de múltiples soluciones, solo dos son las más importantes: *eliminación de caminos y eliminación de nodos.*

## *MÉTODO DE ELIMINACIÓN DE CAMINOS*

El método de eliminación de caminos para la generación de múltiples soluciones elimina de la base de datos todos los nodos que llevan a la solución actual y luego intenta encontrar otra solución.

En esencia, el método de eliminación de caminos poda las ramas del árbol.

## *MÉTODO DE ELIMINACIÓN DE NODOS*

Este método supone otra forma de forzar la generación de soluciones múltiples. Simplemente, elimine el ultimo nodo del camino de la solución actual y luego lo intenta de nuevo.
Este método elimina una entrada, usando cadenas de longitud cero.

## 3.4 Representación reducida del problema

## MÉTODO DE REDUCCIÓN DEL PROBLEMA

El método de reducción del problema por lo general se reconocen las metas y se convierten en submetas apropiadas. Cuando se usa de este modo, la reducción del problema se reconoce a menudo y de manera equivalente como reducción de metas.

Los cubos en movimiento ilustran la reducción del problema el procedimiento mover resuelve problemas de manipulación de cubos y contesta preguntas acerca de su propio comportamiento. Mover trabaja con cubos obedeciendo mandatos como el siguiente:

Coloca < nombre del cubo > sobre < otro nombre de cubo >

Para obedecer, mover planea una sucesión de movimientos para un robot de una sola mano que toma un solo cubo a la vez. Mover consiste en procedimientos que reducen los problemas dados a otros más simples, aventurándose así en lo que se conoce como la reducción del problema.

- Coloca hace que se coloque un cubo encima de otro.
- Consigue espacio, encuentra un lugar en la cima de un cubo destino para el cubo en movimiento.
- Toma, coge cubos. Si la mano robot esta haciendo un cubo cuando se llama a toma debe hacer que el robot se deshaga de ese cubo.
- Despeja cima, limpia la cima. Opera desasiéndose de todo lo que haya en la cima del objeto que se va a tomar.
- Deshazte de, aparta obstáculos poniéndolos sobre la mesa.
- Suelta, hace que el robot suelte lo que esta haciendo.
- Mueve, traslada objetos una vez que han sido cogidos mediante el movimiento de la mano robot.

*La idea clave de la reducción del problema es explorar un árbol de metas.*

Un árbol de metas es una red semántica con enlaces especiales, llamado ramas, cada una de las cuales conecta a dos nodos.

- Ciertos enlaces se conocen como ramas. Cada rama conecta a dos nodos; el nodo cabeza se designa como nodo padre y el nodo cola como nodo hijo.
- Hay un nodo que no tiene padre y se le conoce como nodo raíz. Otros nodos tienen exactamente un padre.
- Algunos nodos no tienen hijos y se les denominan nodos hojas.
- Cuando dos nodos están conectados por una cadena de dos o más ramas, se dice que uno es antecesor y el otro es descendiente.

Contiene constructores que:

- Conectan a un nodo padre a un hijo mediante un enlace rama.
- Producen una lista de los hijos de un nodo dado.
- Producen el padre de un nodo dado.

Contiene lectores que:

- Producen una lista de los hijos de un nodo dado
- Producen el padre de un nodo dado.

Un árbol de metas es un árbol semántico en los que los nodos representan metas y las ramas indican la forma en la que puede lograr metas, mediante la solución de una o más submetas.

Los hijos de cada nodo corresponden a submetas inmediatas. El nodo superior que no tiene padre es la meta raíz.

*Los árboles de metas pueden hacer transparente la interacción entre procedimientos.*

Hace transparente los complicados argumentos de mover. La acción de despejar la cima del cubo A se muestra como una submeta inmediata de tomar el cubo A.

Las metas que satisfacen solo cunden todas sus submetas inmediatas quedan satisfechas se conoce como metas Y. Los nodos correspondientes son los nodos Y, y se les señalas colocando arcos en las ramas.

La mayoría de los árboles de metas contienen también metas O; estas metas se satisfacen cuando cualesquiera de sus metas inmediatas quedan satisfechas. Los nodos correspondientes, que permanecen sin señalar, se conoce como nodos O.

Finalmente algunas metas satisface directamente, sin hacer referencia a ninguna submeta. Estas metas se conocen como metas hoja, y los nodos correspondientes se denominan nodos hoja.
Debido a que los árboles de metas siempre implican nodos Y ó nodos O, o ambos, a menudo se les conoce como árboles Y-O.

Para determinar si una meta se ha logrado, usted necesita un procedimiento de prueba. El procedimiento clave REDUCE, canaliza la acción hacia los procedimientos REDUCE Y y REDUCE O.

El Árbol de Metas Permite Responder Preguntas de Carácter Introspectivo.

El programa Mover es capaz de construir árboles Y-O ya que los especialistas mantienen una estrecha correspondencia con metas identificables.

De hecho los árboles Y-O de Mover son tan ilustrativos que pueden utilizarse para responder preguntas sobre como y porque se han tomado las acciones, otorgando Mover cierto talento para realizar una introspección de su propio comportamiento.

*La reducción del problema es Ubicua en programación.*

Mover consiste en un conjunto de procedimientos especializados. Cada vez que un procedimiento especializado llama a otro, efectúa un paso de reducción del problema. Siempre un procedimiento llame a un subprocedimiento, se da un paso de reducción del problema. Por tanto, la reducción del problema es el método de resolución de problemas que a todos los programas muestran en gran cantidad, excepto los más cortos

*Los métodos de resolución de problemas trabajan juntos*

Pocos son los problemas reales que se pueden resolver mediante un solo método de resolución de problemas. De ahí que con frecuencia se vean métodos de resolución de problemas trabajando juntos.

Por ejemplo, suponga que desea ir de su casa, cerca de Boston, a la de un amigo, cerca de Seattle. Usted puede usar el análisis de medios y metas para decidir que tipo de transporte es el que mas se prefiere porque reduce la distancia entre el lugar en que se encuentra y aquel donde desea estar.

Debido a que la distancia entre Boston y Seattle es grande, el análisis de medios y metas indudablemente le sugeriría tomar el avión, pero esto solo resuelve parte del problema: todavía tiene que ver como llegar al aeropuerto de Boston desde su casa y como transportarse del aeropuerto de Seattle a la casa de su amigo.

Por tanto, la meta inicial, se convierte en tres submetas, cada una de las cuales se pueden tratar, quizá, mediante el análisis de medios y metas.

*Las cajas de herramientas matemáticas aplican la reducción del problema para resolver problemas de cálculo.*

La reducción del problema es un método de resolución de problemas tan obvio y tan poderoso, se encuentra en los primeros que se llevaron a la práctica. Una aplicación

de la reducción del problema, el programa SAINT, fue de los primeros en utilizar la inteligencia artificial en la resolución de problemas matemáticos.

Básicamente, SAINT fue un programa que integraba expresiones simbólicas de una variable.

Los programas parecidos a SAINT efectúan sustitución de variables y otros manejos algebraicos para transformar integrados difíciles en uno ó más integrados de mayor sencillez, así una reducción del problema, con la esperanza de llegar a producir integrados tan simples que la respuesta se pueda hallar directamente en las tablas de integrantes.

## 3.5 Métodos de búsqueda en representaciones reducidas del problema

## RESOLUCIÓN DE PROBLEMAS: BÚSQUEDA DE SOLUCIÓNES

La resolución de problemas es fundamental para la mayoría de las aplicaciones de la IA. La capacidad de resolver problemas suele usarse como una medida de la inteligencia tanto para el hombre como para la computadora. Hay principalmente dos clases de problemas:

- Una primera clase puede ser resuelta usando algún tipo de procedimiento determinista cuyo éxito esta garantizado. A este procedimiento se le llama de computación aplicada en las matemáticas, a través de métodos de frecuencia que se pueden resolver de manera fácil a un algoritmo que pueda ser ejecutado por una computadora.
- A pesar de que pocos problemas reales se prestan a soluciones computables deben ser situados en la segunda categoría que consiste en problemas que resuelven con la búsqueda de una solución. Ese es el método que resolución de problemas de la que se preocupa la IA.

Los obstáculos que se deben evitar cuando se intentan aplicar las técnicas de la IA a problemas reales es la magnitud absoluta y la complejidad de la mayoría de las situaciones, en las investigaciones de la IA se debían desarrollar buenos métodos de búsqueda y esto era un fin primordial por su necesidad.

### *PRESENTACIÓN Y TERMINOLOGÍA*

Presenta la solución en forma de grafos, una forma sencilla de observar como funcionan las diferentes técnicas de búsqueda. (Además la posibilidad de representar problemas como grafos permite a los investigadores de la IA aplicar distintos teoremas de la teoría de grafos).

***METODOS CIEGOS***

Suponga que desea hallar una trayectoria de una ciudad a otra mediante un mapa de carreteras como el de la figura 4. Su trayectoria deberá comenzar en la ciudad S, su punto inicial, y terminar en la ciudad G, su meta. Para hallar una trayectoria apropiada en el mapa, necesitara considerar dos costos distintos:

- En primer lugar, el costo de calculo para *encontrar* la trayectoria.
- En segundo, el costo del viaje cuando se *sigue* la trayectoria.

*La búsqueda de red es en realidad una búsqueda de árbol*

La manera mas obvia de hallar una solución es considerar todas las trayectorias posibles. Por supuesto, usted deberá descartar las trayectorias que pasan mas de una vez por una ciudad en particular a fin de no quedar atrapado en un ciclo, como seria el caso de S-A-D-S-A-D-S-A-D-...

Cada nodo representa una trayectoria. Cada nodo hijo representa una trayectoria que es una extensión de un paso de la trayectoria representada por su padre. Usted convierte redes en árboles de búsqueda trazando todas las trayectorias posibles hasta que ya no pueda extender ninguna de ellas sin crear un ciclo.

Un árbol de búsqueda es una representación, o sea, un árbol semantico en el que:

- Los nodos representan trayectorias

- Las ramas conectan trayectorias a extensiones de trayectoria de un solo paso.

Contiene escritores que:

- Conectan una trayectoria a una descripción de trayectoria.

Contiene lectores que:

- Producen una descripción de trayectoria.

Una vez eliminadas las trayectorias con ciclos, usted puede ordenar todas las trayectorias posibles a partir del nodo inicial en un árbol de búsqueda, un tipo especial de árbol semantico en el que cada nodo representa una trayectoria:

***EXPLOSIONES COMBINATORIAS***

La mayoría de los problemas en los que quieran usar una computadora para hallar la solución, la situación es diferente. Generalmente, usara una computadora para resolver problemas en los que le número de nodos en el espacio de búsqueda sea muy grande y como va creciendo el espacio de búsqueda de igual manera se incrementan el número de diferentes caminos posibles hasta la meta.

El problema estriba en que cada nodo añadido al espacio de búsqueda añade más de un camino; por lo que el número de caminos hasta la meta se incrementara con cada nuevo nodo.

Los investigadores en la IA normalmente se refieren a una explosión combinatoria, concepto de explosión combinatoria con el de la resolución de problemas, cada nodo adicional que se une al espacio de búsqueda incrementa el número de soluciones posibles en un número mucho mayor.

Mientras que la técnica exhaustiva, o "fuerza Bruta" teóricamente siempre funciona normalmente no es una práctica por que consume o demasiado tiempo o demasiados recursos de computación e incluso ambos. Otras técnicas de búsqueda han ido desarrollándose.

**MÉTODOS DE BÚSQUEDA RESPALDADOS CON INFORMACIÓN :**

Contar con información sobre un espacio de estados evita a los algoritmos emprender búsquedas a ciegas, permitiendo así encontrar soluciones con más eficiencia. Presentaremos los siguientes métodos:

***Búsqueda preferente por lo mejor***

Se utiliza esta búsqueda cuando los nodos se ordenan de manera tal que se expande primero aquel con mejor evaluación. Su objetivo es encontrar soluciones de bajo costo, por lo general estos algoritmos utilizan alguna medida estimada del costo de la solución y tratan de reducir esta medida al mínimo.

Para enfocar la búsqueda, debe figurar algún tipo de cálculo del costo de ruta que va de un estado al más cercano a la meta. Se trata de dos tipos de aproximaciones básicas: el primero trata de expandir el nodo más cercano a la meta, y el segundo el correspondiente a la ruta de la solución menos costosa. Explicaremos ambos métodos:

*1. Búsqueda Avara:* consiste en reducir al mínimo el costo estimado para obtener la meta. Es decir, expande primero el nodo cuyo estado se considere ser el más cercano al estado de la meta. Para calcular tales estimados de costos utiliza una función denominada función heurística, simbolizada por:

h (n) = costo estimado de la ruta más barata que une el estado del nodo n con un estado meta.

h puede ser cualquier función, pero debe cumplir con el requisito de ser h (n) = 0, cuando n sea una meta.

Este método presenta las siguientes características:

- ✓ No es óptimo: tiende a procurar soluciones rápidas, pero no siempre garantiza que la solución encontrada sea la mejor.
- ✓ Es incompleto: puede recorrer una ruta infinita y nunca regresar a probar otras posibilidades.

  - ✓ Complejidad: en el peor de los casos es de O (bm) en donde, es la profundidad máxima del espacio de búsqueda.

*2. Búsqueda A*:* surge de la combinación del método avaro y la búsqueda por costo uniforme, se basa en una función:

f(n) = g(n) + h (n)

donde:

g (n) = costo de la ruta que va del nodo de partida al nodo n

h (n) = costo estimado de la ruta más barata que va de n a la meta

Por lo cual: f (n) = costo estimado de la solución más barata, pasando por n.

Es decir, expande primero el nodo cuyo valor de f sea el más bajo.

Características del método:

- ✓ Es óptimo y completo: si se le da a la función h la sencilla restricción que nunca sobrestima el costo que implica alcanzar la meta.

A las funciones que cumplen esta restricción se las conoce como heurísticas admisibles u optimistas, pues consideran que el costo para resolver un problema siempre es inferior a lo que en realidad es.

***Búsqueda limitada por la capacidad de memoria***

1. *Búsqueda A** por profundización iterativa (A* PI): en éste caso la búsqueda preferente por profundidad se modifica para utilizar un límite del costo de f en lugar de un límite de profundidad.De ésta forma, en cada iteración se expanden todos los nodos que están

dentro del contorno del costo f actual, y se echa un vistazo al contorno para determinar en donde se encuentra el siguiente contorno.

Una vez concluida la búsqueda dentro de un contorno, se procede a efectuar una nueva iteración utilizando un nuevo costo f para el contorno siguiente.

Solo necesita de un espacio proporcional con la ruta más larga que explore.

Si ß es el mínimo costo de operador y f* es el costo de la solución óptima, en el peor de los casos en A*PI se necesita de bf*/ß nodos de almacenamientos.

3. *Búsqueda A*SRM*

(A* acotada por memoria simplificada): este método emplea toda la capacidad de memoria disponible para efectuar la búsqueda. El empleo de más memoria permite mejorar la eficiencia de la búsqueda. Se caracteriza por:

- ✓ Hacer uso de toda la memoria que pueda disponer.
- ✓ Evitar estados repetitivos, en la medida que la memoria se lo facilite.
- ✓ Ser completo, si la memoria disponible tiene capacidad suficiente para guardar la ruta de la solución más cercana.
- ✓ Ser óptima, si dispone de memoria suficiente como para guardar la ruta de solución óptima más cercana. De lo contrario produce la mejor solución que sea posible obtener con la memoria disponible.
- ✓ Si dispone de memoria suficiente para todo el árbol de búsqueda, resultará óptimamente eficiente.

## BUSQUEDAS SIN RESPALDO DE INFORMACIÓN

### *Busqueda Preferente Por Amplitud*

En la búsqueda por profundidad siempre se expande uno de los nodos que se encuentre en lo más profundo del árbol. Sólo si la búsqueda conduce a un callejón sin salida (un nodo sin meta que no tiene expansión), se revierte la búsqueda y se expanden los nodos de niveles menos profundos.

Es decir, revisa todas las trayectorias de una longitud dada antes de avanzar a una trayectoria más larga.

El volumen de memoria necesario es bastante modesto. Sólo es necesario guardar la ruta que vaya del nodo raíz al nodo hoja, junto con los nodos restantes no expandidos, por cada nodo de la ruta.

Si un espacio de estados tiene un factor de ramificación b y profundidad máxima m, la cantidad de memoria necesaria en una búsqueda preferente por profundidad serán solo de b*m nodos, en contraste con la cantidad de bd necesaria en una búsqueda preferente por amplitud, si la meta más inmediata se encuentra a una profundidad d.

Funciona aún con árboles infinitamente profundos. Constituye un desperdicio cuando todas las trayectorias conducen a la meta aproximadamente a la misma profundidad.

No es aconsejable si el factor de ramificación es grande, debido a la expanción exponencial.

Se puede opatar por esta búsqueda, si nos preocupa el hecho de que pueda haber trayectorias largas, y hasta infinitas, incapaces de llegar a callejones sin salida ovoverse completas.

***Búsqueda Preferente por profundidad:***

Este método representa una forma sencilla de hallar una trayectoria, consiste en tomar uno de los hijos en cada nodo que se visita y avanzar a partir de ese hijo. Otras alternativas del mismo nivel se ignoran por completo, en tanto haya posibilidades de alcanzar la meta mediante la selección original.

La desventaja de la búsqueda preferente por profundidad es la posibilidad de que se quede estancada al avanzar por una ruta equivocada. En muchos problemas, los árboles de búsqueda son muy profundos, o hasta infinitos, por lo que en una búsqueda preferente por profundidad nunca será posible recuperarse de alguna desafortunada opción en uno de los nodos que están cerca de la parte superior del árbol. La búsqueda proseguirá

siempre en sentido descendente, sin hacia atrás, aún en el caso de que exista una solución próxima.

Por lo tanto, o se queda atorada en un bucle infinito y nunca es posible regresar al encuentro de una solución, o a la larga encontrará una ruta de solución más larga que la solución óptima.

Esto quiere decir que la búsqueda preferente por profundidad ni es la más completa ni la más óptima. Por ello, cuando hay árboles de búsqueda con prolongadas o infinitas profundidades maximas hay que evitar el empleo de la búsqueda preferente por profundidad.

Si la cantidad de soluciones de un problema es excesiva, la búsqueda preferente por profundidad será mucha más rápida que la búsqueda preferente por amplitud, puesto que tiene más probabilidades de poder encontrar una solución luego de explorar tan sólo una pequeña porción de todo el espacio. En el peor de los casos la búsqueda preferente por profundidad se reduce a una búsqueda preferente por amplitud.

***Búsqueda Limitada Por Profundidad***

Se eliminan las dificultades que conlleva la búsqueda preferente por profundidad al imponer un límite a la profundidad máxima de una ruta. Para implantar tal límite se utiliza un algoritmo especial de búsqueda limitada por profundidad, o utilizando el algoritmo general de búsqueda con operadores que se informan constantemente de la profundidad.

Los operadores garantizan que, de existir, se encontrará la solución; lo que no se garantiza es que la primera solución encontrada sea necesariamente la más breve: la búsqueda limitada por profundidad es completa, pero no óptima. Incluso al escogerse un límite de profundidad excesivamente pequeño, la búsqueda limitada por profundidad ni siquiera será completa.

***Búsqueda Por Profundización Iterativa***

Lo difícil de la búsqueda limitada por profundidad es la elección de un límite adecuado. Existe un número, conocido como el diámetro del espacio de estados (máximo número de pasos para llegar a la meta), que constituye un mejor límite de profundidad, lo que, a su vez, favorece una búsqueda limitada por profundidad más eficiente.

Es una estrategia que esquiva el tema de la elección del mejor limite de la profundidad probando todos los límites de profundidad posibles: la primera profundidad 0, luego la 1, la 2 etc.

En efecto en la profundización iterativa se combinan las ventajas de las búsquedas preferente por profundidad y preferente por amplitud.

Es óptima y completa, como la búsqueda por amplitud, al tiempo que la memoria que necesita es sólo la de la búsqueda por profundidad.

El orden de los estados es semejante a la preferente por amplitud, excepto que algunos de los estados se expanden varias veces.

Podría dar la impresión de que es un desperdicio, debido a la expansión de tantos estados tantas veces; sin embargo, en la mayoría de los problemas, el exceso de esta expansión múltiple resulta ser en realidad bastante pequeña.

Una búsqueda por profundización iterativa que comience por la profundidad 1 y continúe hasta llegar a la profundidad se expande solo 11% más nodos que una búsqueda preferente por amplitud o por profundidad hasta la profundidad d.

Por lo general, la profundización iterativa es el método idóneo para aquellos casos donde el espacio de búsqueda es grande y se ignora la profundidad de la solución.

***Búsqueda Bidireccional***

Es una búsqueda simultánea que avanza a partir del estado inicial y que retrocede a partir de la meta y que se detiene cuando ambas búsquedas se encuetran en algún punto intermedio.

Hay que resolver varias cuestiones primero:

¿Qué significa buscar hacia atrás a partir de la meta? : se definen los predecesores de un nodo n como todos aquellos nodos cuyo sucesor es n. La búsqueda hacia atrás implica la sucesiva generación de predecesores a partir del nodo meta.

Si todos los operadores son reversibles, los conjuntos de predecesor y sucesor son idénticos, en algunos problemas sin embargo, el cálculo de los predecesores puede resultar muy dificil.

¿qué se hace cuando son varios los posibles estados meta? : en caso de contar con una lista explicita de los estados meta, podemos aplicar una función de predecesor al conjunto de estado exactamente como se aplicó la función del sucesor en la búsqueda de estado múltiple.

Es necesario contar con una manera eficiente de verificar cada uno de los nodos nuevos para ver si ya están en el árbol de búsqueda de la otra mitad de la búsqueda.

Hay que definir que tipo de búsqueda se realizará en cada una de las mitades.

Para efectuar el procedimiento para probar la intersección de las dos fronteras se emplea una embrollada tabla. Con el fin de que ambas búsquedas lleguen a encontrarse en algún momento, los nodos de por lo menos uno de ellos deberá quedar retenido en la memoria.

## 3.6 Representación en forma de procedimiento o subrutinas.

Son aquellos que se realizan a través de esquemas lógicos probados de un orden de acción y tienen las características de

- Énfasis en representar y soportar las relaciones inferenciales del dominio
- Objetivo: Aplicar métodos potentes de inferencia sobre conjuntos relativamente reducidos de hechos
- Permiten razonar y obtener inferencias con falta de conocimiento
- Adecuados para representar conocimiento heurístico (experiencia)
- Ejemplos: métodos búsqueda en espacio de estados, sistemas de reglas de producción
- En la práctica: mezcla de ambas aproximaciones (reglas de producción, lógica formal)

## 3.7 Redes Semánticas

**MÉTODO DE ANÁLISIS DE MEDIOS Y METAS**

El estado de un sistema es una descripción que basta para determinar el futuro. En un espacio de estados, cada nodo representa un estado y cada enlace una posible transición de un paso de un estado a otro.

Un espacio de estados es una representación, o sea una red semántica en la que:

1. Los nodos representan estados.
2. Los enlaces representan transiciones entre estados.

**RED SEMÁNTICA.**

Modelo de memoria humana para capturar la semántica de las palabras y lograr uso del significado parecido a los humanos. Un tipo de red en la cual los nodos representan objetos, conceptos o situaciones y los arcos representan relaciones entre ellos.

Realmente es una estructura de datos sofisticada y mucho depende del programa que la mantiene y la usa. Se llama red semántica porque se usaron originalmente para representar el sentido en expresiones de lenguaje natural.

Los nodos: conceptos de palabras

Los arcos: ligan conceptos para establecer la definición

Cada palabra o nodo conceptual se consideraba la cabeza de un ``plano'' que tiene su definición (ejemplo, si banco tiene 2 significados, entones existen 2 planos para él).

Una red semántica es una representación en la cual:

- Desde el punto de vista del léxico, existen nodos, enlaces y etiquetas de enlace para aplicaciones específicas.
- Desde el punto de vista estructural, cada enlace conecta un nodo cola con uno cabeza.
- Desde el punto de vista semantico, los nodos y enlaces representan entidades de aplicación específica.

Contiene constructores que:

- Construyen un nodo.
- Construyen un enlace, dados una etiqueta de enlace y dos nodos que se conectan.

Contiene lectores que:

- Producen una lista de todos los enlaces que salen de un nodo dado
- Producen una lista de todos los enlaces que llegan a un nodo dado
- Dado un enlace, producen un nodo cola

**Semántica operativa.**

Debe haber un conjunto de programas que operan sobre las descripciones de la representación. Digamos que el significado esta definido por lo que hacen los programas.

**Semántica descriptiva.**

Debe haber explicaciones de lo que significan las descripciones en términos que podamos entender intuitivamente.

Un espacio de estados es miembro de la familia de redes semánticas de representaciones. Los estados corresponden al lugar en que usted esta o debería estar en el proceso de resolver un problema. El estado actual sea el lugar en que usted se encuentre; el estado meta, el lugar al que se desea llegar; y el problema consiste en hallar una sucesión de transiciones que conduzca del estado inicial al estado meta.

## SISTEMAS DE MARCO Y REDES SEMÁNTICAS

En 1896, siete años después de que Peirce creara lo que actualmente es la notación normaliza de la lógica de primer orden, Charles Peirce propuso una notación gráfica denominada gráficas existentes, a lo que se refirió como “la lógica del futuro”. Se inicio así el prolongado debate entre los partidario de la “lógica” y los de las “redes semánticas”.

Desafortunadamente, este debate trajo consigo el perder de viste la unidad básica de este campo. Actualmente se acepta que toda red semántica o sistema de cuadro también puede definirse como oraciones de una lógica y la mayoría está de acuerdo en que esta sea la lógica de primer orden.

Lo importante en todo lenguaje de representación reside en la comprensión de la semántica y en la teoría de la demostración; los detalles de la sintaxis son menos importantes.

El que en lenguaje se utilicen cadenas o nodos y vínculos, y el que se le llame red semántica o lógica, es irrelevante para su significado o para su implantación.

El formato de un lenguaje puede tener un efecto significativo en cuanto a claridad para un lector humano.

La comprensión de algunas cosas se facilitan mediante la notación gráfica; algunas otras son más claras cuando aparecen como cadenas de caracteres.

Afortunadamente, no es necesario decidirse en definitiva por una o por otra; el profesional en IA diestro es capaz de traducir en ambos sentidos las notaciones y elige aquella que mejor se adapta a sus necesidades del momento; lo que no puede obtener intuiciones de otras notaciones. Algunos sistemas, como el sistema CYC cuenta con los dos tipos de interfaces.

Además del atractivo de esos bonitos diagramas nodo-vinculo, las redes semánticas han tenido aceptación por la misma razón que Prolog se usa más que los demostradores de teoremas de lógica de primer orden: porque el modelo de ejecución de la mayoría de los formalismos de la red semántica es muy sencillo.

Aunque un programador llegue a construir una red muy grande, aún así puede hacerse una buena idea de qué consultas pueden ser eficientes, puesto que es fácil de visualizar es tan sencillo que impide formular consultas complicadas.

Posiblemente esta sea la razón de que muchos de los investigadores pioneros en la ontología del sentido común sintieron más confianza de trabajar con sus teorías mediante el método de la red semántica.

## SINTAXIS Y SEMÁNTICA DE LAS REDES SEMANTICAS

En las redes semánticas la atención esta opuesta en las categoría de los objetos y las relaciones que existen. En ellas es muy natural establecer vínculos como:

Gatos $\overset{Subconjunto}{\longrightarrow}$ Mamíferos

Para afirmar que son mamíferos. Desde luego se trata de una forma senilla de expresar la oración lógica Gatos$\subset$ Mamíferos, pero cuando las redes semánticas se utilizaron por primera vez en IA (alrededor de 1961), este tipo de expresión no gozaba de aceptación generalizada; se pensaba que para utilizar la lógica era necesario escribir:

$\forall$x Gato(x)$\Rightarrow$ Mamífero(x)

y que producía una respetuosa intimidación. Se consideraba también que $\forall$x no daba lugar a excepciones, en tanto que $\overset{Subconjunto}{\longrightarrow}$ era más que una expresión benevolente.

Ahora se reconoce que la semántica es más importante que la notación.

Es irónico pero las redes semánticas a veces adolecen de una semánticas clara. Muchas veces el usuario tiene que inducir la semántica de un lenguaje a partir del comportamiento del lenguaje que lo implanto.

Por ello, los usuarios tienen que considerar las redes semánticas a nivel de implementación, no a nivel lógico o a nivel de conocimiento.

Para definir la semántica de lenguaje de red semántica sencilla es necesario proponer los equivalentes en lógica de primer orden de las aseveraciones que se hagan en el lenguaje de red.

Conviene aclarar que un demostrador de teoremas o un lenguaje de programación lógica es capaz de utilizar las traducciones lógicas de los vínculos para efectuar una herencia mediante inferencia lógica común.

En un sistema de red semántica se emplean algoritmos de propósito especial para recorrer los vínculos, y por ello puede ser más rápido que la inferencia lógica general.

## IMPLANTACIÓN DE REDES SEMÁNTICAS

Una vez definido el significado de una res, se podrá proceder a implantar una red. Desde luego que es posible implantarla mediante un demostrador de teoremas o con un lenguaje de programación lógica, lo que en algunos casos es la mejor opción.

Sin embargo, en el caso de las redes semánticas es sencilla, es posible utilizar una implantación más directa. El nodo de una red se representa mediante una estructura de datos, en la que hay campos para las conexiones taxonómicas básicas: dd que categoría de que miembro, que elementos contiene, cuáles son los subconjuntos y supraconjuntos inmediatos.

## EXPRESIVIDAD DE LAS REDES SEMÁNTICAS

Las redes que se han visto están excesivamente limitadas en cuanto a su expresividad. Por ejemplo, no es posible representar una negación, una disyunción, ni una cuantificación. Este tipo de construcciones son fundamentales en muchos dominios.

En algunas de las redes semánticas se amplia ala notación de manera que puedan trabajar con lógica de primer orden. Es lo que se hizo en las gráficas existenciales de Peirce, redes semánticas fragmentadas(HENDRIX,1975), y en SNEPS(Shapiro, 1979).

En un método de uso más frecuente se conservan las limitaciones en cuanto a expresividad y para cubrir los vacíos se utiliza la vinculación procedimientos. Esta es una técnica en la que la función escrita en un lenguaje de programación se guarda en calidad de valor de una determinada relación y sirve para responder las invocaciones PREGUNTAR correspondientes a dicha relación( a veces también se invoca es decir.

## COMPRENSIÓN DEL LENGUAJE

La comprensión de señales verbales consiste en establecer una correlación entre toda la señal acústica y una interpretación del significado de una señal vocal. El sistema de comprensión de las señales vocales deberá responder a las siguientes preguntas: ¿cuáles son los sonidos de que articulo el hablante?, ¿Qué palabras trata de expresar el hablante por medio de tales sonidos verbales', ¿Qué es lo que el hablante desea expresar con esas palabras?.

Para dar respuesta a la primer pregunta, primero debemos ponernos de acuerdo en lo que es un sonido vocal.

Es un hecho que en todos los idiomas humanos se utiliza un repertorio que consta de 40 a 50 sonidos y que se conocen con el nombre de fonemas. De manera burda que puede considerarse que un fonema es el sonido que corresponde a una vocal o a una constante, aunque en realidad no es tn sencillo: en inglés la combinación de letras como "th" y "ng" produce un solo fonema, y hay letras cuyos respectivos fonemas varían, dependiendo de los contextos en que se presentes (por ejemplo la "a" de rat y rate).

Una vez que sabemos cuáles son todos los sonidos posible, necesitamos caracterizarlos de acuerdo con los rasgos que sea posible recuperar de una señal acústica, como puede ser la frecuencia o la amplitud de las ondas sonoras.

La segunda pregunta resulta mucho más sencilla, desde un punto de vista conceptual, podría considerarse que la consulta de las palabras en un dicci0nario en las que estas se ordenaran de acuerdo con su pronunciación. Obtendremos así una secuencia de tres fonemas [k],[$\wp$] , y [t]; en el diccionario encontráremos que lo anterior corresponde a la pronunciación de la palabra cat (gato), hay dos factores que dificultan lo anterior.

El primero es la existencia de Homófonos palabras distintas que tienen el mismo sonido, como es el caso de las palabras en ingles "two"(dos), y too(también).

El segundo es la segmentación, el problema que consiste en saber en donde termina una palabra y en donde inicia la siguiente. Las personas que hayan intentado aprender una lengua extranjera seguramente están concientes de este problema; al principio dala impresión de que todas las palabras se pronunciaran juntas.

Poco a poco, se aprende a ir tomando palabras de entre la mezcolanza de sonidos. En este caso, las primeras impresiones son las correctas; el análisis espectográfico permite constatar que durante una articulación verbal fluida, las palabras en realidad si están integradas y no hay un silencio entre una y otra. Somos capaces de aprender a identificar los limites entre una palabra y otra a esar de la falta de silencio.

En realidad y sabemos la respuesta a la tercer pregunta: empleo de algoritmos de análisis , en algunos sistemas para la compresión de sonidos vocales se obtiene la cadena de palabras que sea más probable y se les envía directamente aun analizador.

Hay otros sistemas cuya estructura es más compleja, en la que se toman en cuenta diversas posibles interpretando de una palabra, de manera que se pueda lograr la comprensión no obstante que no se hayan logrado identificar adecuadamente algunas palabras.

## 3.8 Cuadro de Minsky

**Minsky**

Uno de los pioneros y creadores de estas teorías es Marvin Minsky. Rafael C. Minsky, Ingeniero Civil (UBA 1967) con título de posgrado en Ingeniería Sanitaria. Desde 1969 hasta 1986 se desempeñó en la Dirección Nacional de Saneamiento Ambiental del Ministerio de Salud, que condujo.

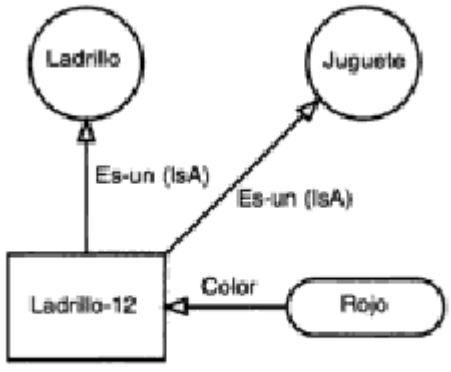
En 1986 ingresó al II PROFAG, egresando en 1988 como Administrador Gubernamental. En tal carácter cumplió funciones en distintos organismos públicos. Desde julio de 2003 a mayo de 2007 fue Subgerente de Compras y Contrataciones del INSSJP.

Dicta las cátedras de Análisis y Diagnóstico Organizacional y Planeamiento y Gestión del Territorio para la Licenciatura en Administración Pública de la U.N. de Tres de Febrero.

Para este autor existen algunas diferencias, como por ejemplo el en su percepción, distingue tres tipos de conocimiento, agregando un tercer tipo a la clásica clasificación epistemológica dejandonos asi:

1. Conocimiento declarativo
2. Conocimiento procedural (*procedural*)
3. Conocimiento de depuración (*debugging*)

El conocimiento de depuración es el que permite saber qué hacer cuando no se conoce algo, lo que se conoce no funciona, o hay una equivocación. Este conocimiento puede llegar a entenderse como el resultado de “aprender a aprender” [Minsky, 1975].




Un marco representado como red semántica.

# UNIDAD IV
# USO Y MODIFICACION DEL CONOCIMENTO

## 4.1 Programa que resuelva problemas generales

LOGICA DEL PROGRAMA

El programa se ha diseñado para diagnosticar un problema con una cocina eléctrica o de gas stove y recomienda la mejor forma de arreglarlos.
Los problemas y formas de arreglarlos son los siguientes.

Primero el programa determina si la cocina es eléctrica o de gas y si el problema es con los quemadores o con el horno.

**PROBLEMA**

No llega electricidad a la cocina

Compruebe la caja de fusibles, reemplazando cualquiera que este fundido. Si esto no funciona bien, llame a un profesional.

*COCINA ELECTRICA*

Los quemadores no funcionan

Intercambie los lugares con un elemento en buen estado. Si el elemento aun no funciona, sustitúyalo. Si el elemento funciona, quítelo y com pruebe los terminales para ver si están doblados, teñidos de azulado o recubiertos de negro.

Todo ello puede significar que la unidad bloque terminal esta dañada o está mal. Compruebe marcas de quemado que se hacen en la unidad y, si las encuentra, reemplácela.

Si no, compruebe el quemador y compruébelo si funciona. Si no, es que hay un cable defectuoso y debe rastrearlo desde el enchufe hasta el interruptor para encontrar el problema.

El horno no funciona

Compruebe la resistencia del horno para comprobar si esta mal, reemplacela en ese caso. Si no, compruebe el interruptor y reemplacelo si esta mal. Si no, compruebe el cable.

COCINA DE GAS
Los quemadores no funcionan

Compruebe los del quemador y reemplácelos si están rotos. Si no, compruebe el regulador y reemplácelo si esta roto. Si no, compruebe la unidad de electrodo y reemplácela si esta mal. Si no, llame a un profesional.

El horno no funciona

Compruebe si el encendedor está incandescente. Si no es así, reemplácelo. En caso contrario, compruebe y limpie el orificio y reemplácelo. Compruebe que no hay fugas durante la instalación. Si nada de eso funciona, llame a un profesional.

---

**CODIGO DEL PROGRAMA**

---

regla 1:
Inicializa el programa

---

```
(defrule iniciar
 (declare (salience 9980))
 ?x <- (initial-fact)
=>
 (retract ?x)
 (assert (necesita profesional no))
 (assert (empezar))
)
```

---

regla 2:
Usada para finalizar el programa y llamar a
la regla inicial despues de resetear el programa

---

```
(defrule fin1
 (declare (salience 9200))
 ?w <- (parar)
```

```
 =>
 (retract ?w)
 (reset)
 (halt)
)
```

---

regla 3:
Da el mensaje de apertura y pregunta que tipo de cocina tiene

---

```
(defrule inicio
 (declare (salience -9000))
 ?x <- (empezar)
=>
 (printout t crlf crlf crlf
          "El programa esta diseñado para ayudarlo a reparar una cocina electrica" crlf
          "o de gas.  El programa le indicara paso a paso los procedimientos" crlf
          " que ha de realizar para reparar la cocina." crlf crlf
          "Tiene usted una" crlf
     "  1) Cocina electrica" crlf
          "  2) Cocina de gas" crlf
     "  3) o salir del programa" crlf
          "Elija 1 - 3 -> ")
 (retract ?x)
 (assert (tipo cocina =(read)))
)
```

---

regla 4:
Comprueba que se ha realizado una seleccion correcta del menu
de apertura; en caso contrario, muestra mensaje

---

```
(defrule comprueba-seleccion
 (declare (salience 9998))
 ?x <- (tipo cocina ?ch&:(not (numberp ?ch)))
 (or (test (<= ?ch 0)) (test (> ?ch 3)))
=>
 (printout t crlf crlf
          "*** Su seleccion debe ser 1,2, o 3 ***" crlf)
 (retract ?x)
 (assert (empezar))
)
```

---

regla 5:

Pregunta si hay problemas con los quemadores

---

```
(defrule prob-quemadores
 (declare (salience -500))
 (tipo cocina ?)
=>
 (printout t crlf crlf "¿Hay problemas con los quemadores?. Conteste si o no ")
 (assert (problema quemadores =(read)))
)
```

---

regla 6:

Pregunta si hay problemas con el horno

---

```
(defrule prob-horno
 (declare (salience -1000))
 (tipo cocina ?)
=>
 (printout t crlf crlf "¿Hay problemas con el horno?. Conteste si o no ")
 (assert (problema horno =(read)))
)
```

---

rule7

Pregunta si hay algun problema de electricidad en la cocina

---

```
(defrule comprobar-energia
 (declare (salience 1500))
 (tipo cocina 1)
=>
 (printout t crlf crlf "¿Llega a la cocina electricidad?. Conteste si o no ")
 (assert (energia =(read)))
)
```

---

rule8

tells the user to check the breaker for a blown fuse

---

```
(defrule comprobar-cajafusibles
 (declare (salience 100))
 ?x <- (energia no) =>
 (printout t crlf crlf
          "Vaya a su caja de fusibles y compruebe el fusible de la" crlf
          "cocina para ver si esta fundido. Si es asi, cambielo" crlf
          "e intente encenderla de nuevo." crlf crlf
          "¿Esta todavia la cocina sin electricidad?. Conteste si o no ")
 (retract ?x)
 (assert (todavia sin energia =(read)))
)
```

---

regla 9:

comprueba si despues de comprobar fusibles esta sin energia

---

```
(defrule prob-todavia
 ?x <- (todavia sin energia si)
=>
 (retract ?x)
 (assert (necesita profesional si))
 (assert (parar))
)
```

---

regla 10:

dice al usuario que deje a un profesional el problema
para evitar peligros

---

```
(defrule pida-ayuda
 (declare (salience 9999))
 (necesita profesional si)
=>
 (printout t crlf crlf
          "Este problema puede ser serio y deberia dejarse a un " crlf
          "profesional. Seguir intentando reparar la cocina podria ser" crlf
          "peligroso." crlf crlf)
 (assert (parar))
)
```

---

regla 11:

pregunta al usuario que tipo de elemento quemador tiene

---

```
 (defrule tipo-elemento
```

```
 (tipo cocina 1)
 (problema quemadores si) =>
 (printout t crlf crlf "De que tipo es el quemador de la cocina? " crlf crlf
                       " 1) Insertado" crlf
                       " 2) Atornillado" crlf crlf
                       "Elija opcion -> ")
 (assert (tipo quemador =(read)))
)
```

---

regla 12:

comprueba si un quemador del mismo tamaño del que da problemas esta funcionando

---

```
(defrule otro-igual-funcionando
 (tipo cocina 1)
 (problema quemadores si)
=>
 (printout t crlf crlf
          "¿Hay al menos un quemador del mismo tamaño que este todavia funcionando?.
Conteste si o no ")
 (assert (igual funcionando =(read)))
)
```

---

rule 13

regla para intercambiar un elemento que funciona con otro que no para comprobacion

---

```
(defrule intercambiar-elemento
 ?x <- (igual funcionando si)
=>
 (printout t crlf crlf
          "Quite el elemento que no funciona correctamente e intercambielo" crlf
          "con otro quemador del mismo tamaño." crlf crlf
          "***  asegurese de apagar la cocina durante la operacion ***" crlf crlf
          "Ponga ahora al maximo los controles del elemento. El elemento que " crlf
          "no trabajaba correctamente lo hace ahora?. Conteste si o no ")
 (retract ?x)
 (assert (elemento roto =(read)))
)
```

---

rule14

Determina que el elemento esta roto

```
(defrule elemento-roto
 ?x <- (elemento roto si)
 =>
 (printout t crlf crlf
          "El elemento del quemador esta roto. Debe reemplazar el elemento roto" crlf
          "Despues, compruebe el quemador y, si hay algun problema, vuelva a " crlf
          "ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)
```

regla 15:

Dice al usuario que use un ohmetro para comprobar el elemento

```
(defrule intercambiar-elemento2
 ?x <- (igual funcionando no)
=>
 (printout t crlf crlf
          "Como no hay otro elemento igual funcionando para intercambiar" crlf
          "con el que no funciona, tendra que comprobar el elemento con un" crlf
          "ohmetro. Desconecte la cocina y quite el quemador. Ponga una sonda" crlf
          "del ohmetro sobre cada uno de los terminales del elemento." crlf
          "¿Registr el ohmetro resistencia 0 para el elemento?. Conteste si o no. ")
 (retract ?x)
 (assert (elemento roto =(read)))
)
```

;regla 16:

Comprueba si los terminales estan doblados

```
(defrule comprobar-elemento1
 (declare (salience -20))
 (problema quemadores si)
 (tipo quemador 1)
=>
 (printout t crlf crlf
          "Compruebe los extremos de los terminales." crlf
          "¿Se ven doblados?. Conteste si o no. ")
 (assert (terminales doblados =(read)))
)
```

regla 17

regla para cuando haya un terminal doblado

---

```
(defrule terminales-doblados
 ?x <- (terminales doblados si) =>
 (printout t crlf crlf
        "Este muestra que el elemento ha sido instalado incorrectamente." crlf
        "Esto puede provocar fallos en el bloque terminal." crlf)
 (retract ?x)
 (assert (comprobar bloque terminal)))
```

---

regla 18

Comprueba si hay una capa negra sobre los terminales

---

```
(defrule comprobar-elemento2
 (declare (salience -10))
 (problema quemadores y)
 (tipo quemador 1) =>
 (printout t crlf crlf
        "Compruebe los extremos de los terminales." crlf
        "¿Hay algún rastro negro sobre los terminales?. Conteste si o no ")
 (assert (capa negra =(read)))
)
```

---

regla 19

regla por si se ha encontrado una capa negra en los terminales

---

```
(defrule capa-negra
 ?x <- (capa negra si)
 =>
 (printout t crlf crlf
        "Esto es debido a un mal contacto con los terminales" crlf
        "del bloque terminal." crlf)
 (assert (comprobar bloque terminal))
 (retract ?x)
)
```

---

regla 20:

regla para comprobar si hay una capa azul sobre los terminales

```
(defrule capa-azul
 (declare (salience -10))
 (problema quemadores si)
 (tipo quemador ?)
=>
 (printout t crlf crlf
          "Compruebe en los extremos de los terminales algun rastro de color azul" crlf
          "¿Existe o no?. Conteste si o no ")
 (assert (capa azul =(read)))
)
```

---

regla 21:
regla para cuando se encuentre una capa azul sobre terminales

```
(defrule existe-capa-azul
 ?x <- (capa azul si)
=>
 (printout t crlf crlf
          "Esto es debido a que la grasa ha salpicado los terminales" crlf
          "y se han calentado a una temperatura muy elevada. Esto puede causar" crlf
          "fallos en el bloque terminal" crlf)
 (retract ?x)
 (assert (comprobar bloque terminal))
)


;*************************************************************************
;regla 22:
;          Dice al usuario que compruebe si la unidad bloque terminal tiene
;          marcas de quemaduras
;
;*************************************************************************

(defrule comprobar-bloque-terminal
 ?x <- (comprobar bloque terminal)
=>
 (printout t crlf crlf
          "Ilumine con una luz dentro del bloque temrminal." crlf
          "¿Se ven los contactos quemados por dentro?. Conteste si o no ")
 (retract ?x)
 (assert (contactos quemados =(read)))
)
```

```
;*****************************************************************
;regla 23
;           regla por si hay marcas de quemaduras en el bloque terminal
;
;*****************************************************************

(defrule contactos-quemados
 ?x <- (contactos quemados si)
 =>
 (printout t crlf crlf
           "Sustituya el bloque terminal para el quemador. Despues" crlf
           "lije el final del terminal con un papel de lija fino o con lana de acero." crlf
           "Reinstale la unidad y compruebe el quemador. Si todavia hay problemas" crlf
           "vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)

;*****************************************************************************
;regla 24:
;           comprueba si hay algun desgaste sobre la superficie del elemento
;
;*****************************************************************************

(defrule comprobar-desgaste-elemento
 (declare (salience -10))
 (tipo cocina 1)
 (problema quemadores si) =>
 (printout t crlf crlf
           "Los elementos electricos usan alambres de una aleacion hecha" crlf
           "de niquel y cromo cubierta por una funda metalica con una capa" crlf
           "negra. ¿Se ve desgastada dicha capa en cualquier punto?." crlf
           "Conteste si o no ")
 (assert (elemento desgastado =(read)))
)

;***********************************************
;regla 25:
;           Regla por si el elemento esta desgastado
;
;***********************************************

(defrule elemento-desgastado
 ?x <- (elemento desgastado si)
 =>
```

```
 (printout t crlf crlf
          "El alambre de niquel-cromo ha tocado la funda y se ha quemado." crlf
          "Sustituya el elemento. Despues vuelva a comprobar el quemador." crlf
          "Si todavia hay problemas, vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar)))

;*****************************************************
;rule26:
;          Regla para comprobar si hay alambres doblados
;
;*****************************************************

(defrule comprobar-alambres
 (declare (salience -100))
 (elemento roto no)
=>
 (printout t crlf crlf
          "Ahora debe usted levantar la parte superior de la cocina" crlf
          "y comprobar los alambres que conducen al elemento que no funciona." crlf
      "¿Se ven doblados los alambres?. Conteste si o no ")
 (assert (alambre doblado =(read)))
)

;*******************************************
;rule27:
;          Regla por si hay un alambre doblado
;
;*******************************************

(defrule alambre-doblado
 ?x <- (alambre doblado si)
 =>
 (printout t crlf crlf
          "Primero asegurese de que la cocina esta apagada. Despues elimine" crlf
      "el aislamiento del area ""rizada"" y doble el extremo suelto." crlf
          "Suelde el cable y cubra la union con un revestimento ceramico." crlf
          "*** PRECAUCION - no use un recubrimiento plastico, ya que esta" crlf
          "              temperatura tan alta provocaria que se fundiera" crlf
          "Despues, enchufe la cocina y vuelva a comprobar el elemento." crlf
          "Si todavia hay problemas, vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)

;*********************************************
;rule27:
```

```
;           regla por si no hay alambre doblado
;
;*********************************************

(defrule no-alambre-doblado
 ?x <- (alambre doblado no)
=>
 (printout t crlf crlf
           "Desenchife la cocina. Despues, quite los tornillos del panel de control" crlf
           "e inclinelo. Puede que sea necesario quitar los tornillos de la  " crlf
           "parte posterior del panel.  Etiquete los cables que van al interruptor" crlf
           "del elemento que no funciona y los que van al que si funciona. Intercambie " crlf
           "los cables y enchufe la cocina. Ponga el elemento al maximo. ¿Funciona" crlf
           "ahora el elemento malo y el bueno no?. Conteste si o no ")
 (retract ?x)
 (assert (interruptor roto =(read)))
)

;*********************************************************
;regla 28
;           Regla para el caso de que haya un interruptor roto
;
;*********************************************************

(defrule interruptor-roto
 ?x <- (interruptor roto si)
=>
 (printout t crlf crlf
           "Sustituya el interruptor del elemento que no funciona y enganche" crlf
           "los alambres en sus lugares originales. Despues vuelva a comprobar" crlf
           "el elemento. Si ay algun otro problema vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)

;************************************************************
;regla 29
;           Regla para el caso de que el interruptor no este roto
;
;************************************************************

(defrule interruptor-bueno
 ?x <- (interruptor roto no)
=>
 (printout t crlf crlf
           "Tiene un alambre roto en algun lugar entre el interruptor y el " crlf
           "elemento que no funciona. Debe buscar la zona defectuosa y sustituirla." crlf
```

```
        "Despues vuelva a comprobar la cocina. Si todavia hay algun problema" crlf
        "vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)


;*****************************************************************
;regla 30
;       Da el primer paso si hay problemas con un horno electrico
;
;*****************************************************************

(defrule comprobar-horno
 (tipo cocina 1)
 (problema horno si)
=>
 (printout t crlf crlf
        "Encienda el horno a 400 grados y determine que elementos" crlf
        "no funcionan. Apague el horno y deje que se enfrie Desenchufe." crlf
        "la cocina y quite la placa que tiene la resistencia que no" crlf
        "funciona. Tire de la resistencia hacia usted. La resistencia" crlf
        "estara sujeta por cables de alimentacion con conectores o tornillos." crlf
        "Quite los cables y saque la resistencia. Compruebe la resistencia " crlf
        "con un ohmetro poniendo una sonda en cada terminal." crlf
        "Registra el medidor una resistencia 0?. Conteste si o no ")
 (assert (elemento horno roto =(read)))
)

;*************************************************
;regla 31
;       Regla por si hay un elemento del horno roto
;
;*************************************************

(defrule elemento-horno-roto
 ?x <- (elemento horno roto si)
=>
 (printout t crlf crlf
        "El elemento del horno esta roto y debe ser sustituido. Sustituya el " crlf
        "elemento y vuelva a intentar que el horno funcione. Si sigue habiendo
problemas" crlf
        "vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)
```

```
;*************************************************
;rule32
;           Regla si el elemento del horno esta bien
;
;*************************************************

(defrule elemento-horno-bien
 (declare (salience 100))
 (elemento horno roto no)
=>
 (printout t crlf crlf
           "Busque un diagrama del cableado del interruptor de seleccion." crlf
           "Este suele estar pegado en la parte interior trasera del panel" crlf
           "o en el manual del propietario. Si no hay ningun diagrama del cableado" crlf
           "utilice el que viene en el manual de este programa. Utilizando el " crlf
           "grafico compruebe el interruptor con un ohmetro. Desenchufe el horno." crlf
           "Abra el panel de control para acceder a los contactos del interruptor." crlf
      "Etiquete los cables y desconectelos. Conecte el ohmetro a los terminales"crlf
           "L1 y BK. ¿Hay resistencia 0?. Conteste si o no")
 (assert (interruptor falla =(read)))
)

;*****************************************************
;regla 33
;           Regla para comprobar el interruptor del horno
;
;*****************************************************

(defrule comprobar-interruptor1
 (elemento horno roto no)
 (interruptor falla ~si)
=>
 (printout t crlf crlf
           "Conecte ahora a los terminales PL y N. ¿Hay resistencia 0?. Conteste si o no"
crlf)
 (assert (interruptor falla =(read)))
)

;*****************************************************
;regla 34
;           Regla para comprobar el interruptor del horno
;
;*****************************************************

(defrule comprobar-interruptor2
 (declare (salience -10))
 (elemento horno roto no)
```

```
 (interruptor falla ~si)
=>
 (printout t crlf crlf
          "Conecte ahora a los terminales PL y BR. ¿Hay resistencia 0?. Conteste si o
no." crlf)
 (assert (interruptor falla =(read)))
)


;*****************************************************************
;regla 35
;          Regla para comprobar el interruptor del gratinador del horno
;
;*****************************************************************

(defrule comprobar-interruptor3
 (declare (salience -100))
 (elemento horno roto no)
 (interruptor falla ~si)
=>
 (printout t crlf crlf
          "Para comprobar el gratinador:" crlf
          "Conecte ahora a los terminales L1 y BR.¿Hay resistencia 0?. Conteste si o no."
crlf)
 (assert (interruptor falla =(read)))
)

;*****************************************************************
;regla 36
;          Regla para comprobar el interruptor del gratinador del horno
;
;*****************************************************************

(defrule comprobar-interruptor4
 (declare (salience -150))
 (elemento horno roto no)
 (interruptor falla ~si)
=>
 (printout t crlf crlf
          "Conecte ahora a los terminales L1 y PL.¿Hay resistencia 0?. Conteste si o no."
crlf)
 (assert (interruptor falla  =(read)))
)
```

```
;*********************************************************
;regla 37
;           Regla por si hay un interruptor del horno averiado
;
;*********************************************************

(defrule switch-failed
 (declare (salience 10))
 ?x <- (interruptor falla si)
=>
 (printout t crlf crlf
           "Debe usted reemplazar el interruptor. Despues, situe los cables" crlf
           "en su posicion original y vuelva a comprobar el horno. Si todavia" crlf
           "hay problemas, vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)

;************************************************
;regla 38
;           regla para ver si el reloj esta activado
;
;************************************************

(defrule comprobar-reloj
 (declare (salience 200))
 (elemento horno roto no)
=>
 (printout t crlf crlf
           "Compruebe el temporizador del horno. Esta puesto en AUTOMATIC?. Conteste
si o no ")
 (assert (reloj activo =(read)))
)


;*******************************************
;regla 39
;           regla por si el reloj esta activado
;
;*******************************************

(defrule reloj-activado
 (declare (salience 200))
 ?x <- (reloj activo si)
=>
 (printout t crlf crlf
           "El horno no funcionara hasta que se alcance la hora fijada en el reloj." crlf
```

```
        "Ponga el interruptor en modo manual y compruebe de nuevo. Si todavia hay" crlf
        "problemas, vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)

;*******************************************************
;regla 40
;       regla para comprobar el selector de temperatura
;
;*******************************************************

(defrule control-temperatura
 (declare (salience -100))
 (tipo cocina 1)
 (problema horno si)
=>
 (printout t crlf crlf
        "Ahora vamos a comprobar el selector de temperatura del horno." crlf
        "Mire si hay un diagrama del cableado pegado en el horno" crlf
        "o en el manual del propietario relativo al interruptor de control"crlf
        "de temperatura. Si no hay utilice el que hay en el manual de este."crlf
        "libro. Desenchufe el horno, quite los tornillos y abra el panel de control." crlf
        "Etiquete los cables. Ponga el selector a 300 grados. Utilizando un " crlf
        "ohmetro, situe los terminales en los contactos 1 y 2 y accione el interruptor" crlf
        "para gratinar. Es 0 la resistencia? Conteste si o no ")
 (assert (selector temperatura bueno =(read)))
)


;*******************************************************
;regla 41
;       regla para comprobar el selector de temperatura
;
;*******************************************************

(defrule control-temperatura2
 (selector temperatura bueno ~no)
=>
 (printout t crlf crlf
        "Mantenga los conectores en los terminales 1 y 3 y ponga el control" crlf
        "en hornear. Es 0 la resistencia? Conteste si o no ")
 (assert (selector temperatura bueno =(read)))
)

;*******************************************************
```

```
;regla 42
;           regla para comprobar el selector de temperatura
;
;*****************************************************

(defrule tenp-control3
 ?x <- (selector temperatura no)
=>
 (printout t crlf crlf
    "Debe usted reemplazar el interruptor. Despues, situe los cables en su" crlf
    "posicion original, enchufe la cocina y compruebe el horno. Si todavia" crlf
    "tiene problemas, vuelva a ejecutar este programa.")
 (retract ?x)
 (assert (parar))
)


;*****************************************************
;*                                                   *
;* Las reglas para las cocinas de gas comienzan aqui ****
;*                                                   *
;*****************************************************

;*********************************************
;regla 43
;           Primera regla si hay una cocina de gas
;
;*********************************************

(defrule cocina-de-gas
 (declare (salience 100))
 (tipo cocina 2)
 (problema quemadores si)
=>
 (printout t crlf crlf
           "Que encendedores tiene? " crlf
           "  1) alimentados son gas" crlf
           "  2) tipo bujía" crlf crlf
           "Choose 1 or 2 ")
 (assert (tipo encendedor =(read)))
)

;*********************************************************
;regla 44
;           Regla por si el encendedor esta alimentado por gas
;
;*********************************************************
```

```
(defrule alimentado-gas
 (declare (salience 100))
 ?x <- (tipo encendedor 1)
=>
 (printout t crlf crlf
           "Desconecte el gas mientras manipula cualquier parte de la cocina" crlf)
 (retract ?x)
)


;************************************************
;regla 45
;           Regla por si el encendedor es electrico
;
;************************************************

(defrule tipo-encendedor
 (declare (salience 100))
 ?x <- (tipo encendedor 2)
=>
 (printout t crlf crlf
           "Desenchufe la cocina mientras manipula cualquier elemento" crlf)
 (retract ?x)
)

;***************************************
;regla 46
;           Comprueba los quemadores de gas
;
;***************************************

(defrule quemadores-gas
 (tipo cocina 2)
 (problema quemadores si)
=>
 (printout t crlf crlf
           "Cada pareja de quemadores utiliza un encendedor comun." crlf
           "Esta un quemador apagado mientras que el otro que usa el mismo" crlf
           "encendedor funciona?. Conteste si o no ")
 (assert (llave gas averiada =(read)))
)


;*********************************************
;regla 47
;           Regla por si hay un encendedor averiado
```

```
;
;*********************************************

(defrule encendedor-malo
 ?x <- (llave gas averiada si)
 ?y <- (problema quemadores ?)
=>
 (printout t crlf crlf
          "Sustituya el interruptor del quemador que no funciona ya que, " crlf
          "probablemente, el agua ha penetrado en el causando un cortocircuito ." crlf
          "Reemplacelo y compruebe el quemador. Si todavia tiene problemas, " crlf
          "vuelva a ejecutar este programa.")
 (retract ?x ?y)
 (assert (parar))
)

;*****************************************
;regla 48
;          Regla para comprobar el encendedor
;
;*****************************************

(defrule comprobar-encendedor
 (declare (salience -100))
 (tipo cocina 2)
 ?x <- (problema quemadores si)
=>
 (printout t crlf crlf
          "Si los dos quemadores que comparten el mismo encendedor no funcionan," crlf
          "desconecte el horno. En el modulo, desconecte e invierta los cables de" crlf
          "los dos electrodos. Vuelva a conectar el horno y compruebe todos los" crlf
          "quemadores. Si el problema estaba en los quemadores de un lado y ahora" crlf
          "esta en los del otro, o viceversa, reemplace el electrodo del lado que" crlf
          "no funciona. Antes de extraer el ensamblaje electrodo/cable fuera del" crlf
          "hornillo, corte el antiguo electrodo, una con una cinta aislante los" crlf
          "cables viejos fuera del tornillo mientras esta poniendo los nuevos." crlf
          "Si todavia tiene problemas, vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)


;*************************************************
;regla 49
;          Regla para comprobar el encendedor del horno
;
;*************************************************
```

```
(defrule horno-gas
 (tipo cocina 2)
 (problema horno si)
=>
 (printout t crlf crlf
           "Encienda el horno y mire dentro. Brilla el extremo del encendedor?." crlf
           "Conteste si o no ")
 (assert (encendedor horno bueno =(read)))
)


;*********************************************************
;regla 50
;          Regla por si hay un encendedor del horno averiado
;
;*********************************************************

(defrule encendedor-horno-averiado
 ?x <- (encendedor horno bueno no)
=>
 (printout t crlf crlf
           "Reemplace la bobina del encendedor del horno." crlf crlf
           "** ATENCION - no toque la bobina durante la instalacion" crlf
           "            ya que el aceite de sus dedos la estropearia ***" crlf crlf
           "Despues de ello, si todavia persiste el problema, vuelva a ejecutar este
programa. " crlf)
 (retract ?x)
 (assert (parar))
)

;*********************************************************
;regla 51
;          Regla por si el encendedor del horno esta bien
;
;*********************************************************

(defrule encendedor-horno-bien
 (encendedor horno bueno si)
=>
 (printout t crlf crlf
           "Tiene usted un encendedor alimentado por gas?. Conteste si o no ")
 (assert (encendedor alimentado gas =(read)))
)


;*********************************************************
```

```
;regla 52
;           Regla por si hay un encendedor alimentado por gas
;
;*****************************************************

(defrule encendedor-alimentado-gas
 ?x <- (encendedor alimentado gas si)
=>
 (printout t crlf crlf
           "Las salpicaduras pueden haber obstruido el orificio." crlf
           "Desenrosque el quemador y limpielo. Extienda jabón detergente" crlf
           "y agua alrededor de la salida del quemador, y conecte de nuevo el gas." crlf
           "Si se empiezan a formar burbujas en la base reajuste el quemador" crlf
           "y vuelva a comprobar. Si las burbujas persisten, apague el gas, quite" crlf
           "la tapa del quemador y aplique un producto para unir tuberias en" crlf
           "toda la trama. Reinstale la tapa y compruebe si hay perdidas." crlf
           "Se siguen formando burbujas? Conteste si o no ")
 (retract ?x )
 (assert (burbujas =(read)))
)

;*************************************************************************
;regla 53
;           Regla para comprobar perdidas de gas durante la limpieza del orificio
;
;*************************************************************************

(defrule burbujas
 ?x <- (burbujas si)
 ?y <- (necesita profesional ?)
=>
 (retract ?x ?y)
 (assert (necesita profesional si))
)


;***********************************
;rule54:
;           regla por si no hay burbujas
;
;***********************************

(defrule no-burbujas
 ?x <- (burbujas no)
=>
 (printout t crlf crlf
   "Vuelva a comprobar la cocina."crlf
```

```
   "Si hay todavia problemas vuelva a ejecutar este programa." crlf)
 (retract ?x)
 (assert (parar))
)


;*****************************************************************
;rule55:
;           Regla por si el problema es mas apropiado para un profesional
;
;*****************************************************************

(defrule no-gas-fed-pilot
 (encendedor alimentado gas no)
 ?x <- (necesita profesional ?)
=>
 (printout t crlf crlf
           "El problema deberia ser tratado por un profesional" crlf
           "ya que es peligroso trabajar con las partes restantes " crlf
           "relacionadas con gas.")
 (retract ?x)
 (assert (necesita profesional si))
)


;*********************************************************
;rule56:
;           Regla cuando el usuario quiere salir del programa
;
;*********************************************************

(defrule salir
 (declare (salience 9700))
 (tipo cocina 3)
=>
 (printout t crlf crlf crlf crlf
           "Saliendo del programa " crlf crlf)
 (assert (parar))
 (assert (salir))
)


;*******************************************************
;rule57:
;           Regla para cuando el usuario abandone el sistema
;
;*******************************************************
```

```
(defrule finalizado
 (declare (salience 9990))
 ?w <- (salir)
=>
 (retract ?w)
 (halt)
)
```

## 4.2 Programa que resuelva problemas generales por medio de Premisas

***GENERAL PROBLEM SOLVER (GPS)***

Newell, Shaw and Simon

Primer programa en separar procedimientos generales de solución de conocimiento específico del dominio.

El espacio es codificado en metas, objetos y operadores.

Método primitivo para resolver problemas de manera jerárquica

Uso de Means-Ends Analysis (MEA): la idea es reducir la diferencia entre el estado actual y la meta.

El GPS usa las diferencias de dos formas:

1. sirve como una medida de evaluación del progreso global hacia la meta
2. pueden sugerir qué acción a realizar.

Los operadores se describen como tripletes (generalmente por medio de una tabla):

- precondiciones
- función de transformación definiendo los efectos

3. descripción de las diferencias reducidas por el operador

***Trabajo inicial (GPS)***

Establecemos para iniciar una idea importante*:* de esta forma comenzamos por la descomposición de problemas grandes y/o complicados, para transformarlos en subproblemas más simples, para los cuales su solución es más fácil de encontrar (ya sea eventualmente o directamente), y luego al concluir con éstas, combinar dichas soluciones (que son parciales) para obtener una solución del problema inicial que es el grandey/o coplicado (aditividad).

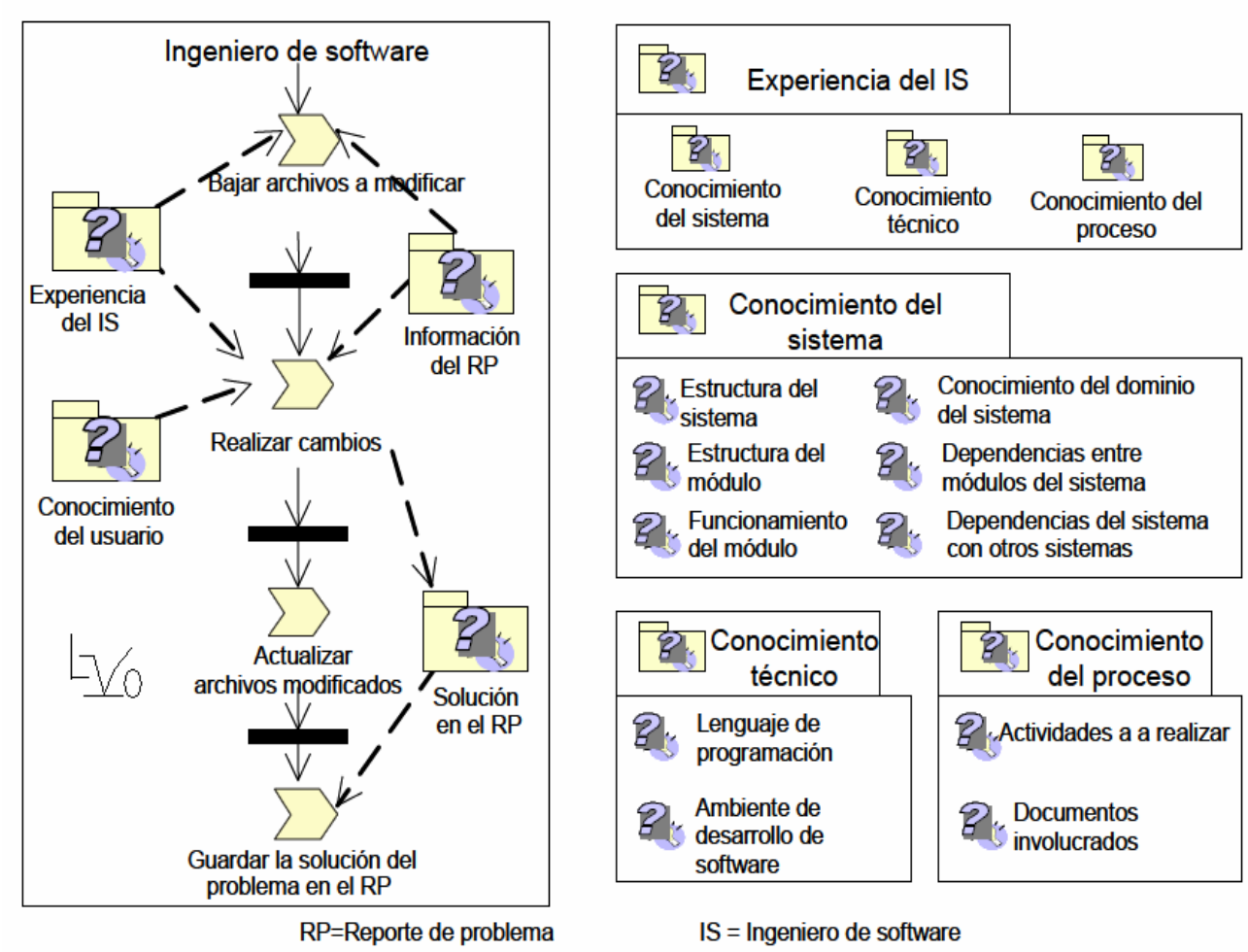
No siempre se pueden hacer descomposiciones totalmente independientes. Los métodos de planeación fueron desarrollados en gran medida para manejar la descomposición de problemas, interacción de subproblemas y representación de estados, aumentando flexibilidad y manteniendo la eficiencia.

Dentro de lo que se puede manejar en forma simple, vamos a presentar ahora un caso de estudio, basado en un caso que aplica en México.

Con el fin de aplicar estrategias de AC en una organización, uno de los primeros pasos es identificar las necesidades reales de conocimiento de las personas involucradas en los procesos de la misma.

Esto puede llevarse a cabo por medio de una "auditoria de conocimiento" que nos ayude a identificar qué conocimiento es requerido por los participantes del proceso, qué fuentes de conocimiento existen, y qué conocimiento puede ser obtenido de dichas fuentes.

Para facilitar el análisis y manejo de esta información, una técnica muy útil es la definición de taxonomías de conocimientos y fuentes, las cuales nos permitirán clasificar estos elementos de una manera que se facilite la estructuración y generación de la base de conocimientos de la organización.




Ejemplo de un diagrama de actividad que muestra el conocimiento involucrado en las actividades.

En el caso de estudio presentado en esta sección, como podemos observar en la imagen superior, se ejemplifica cómo el uso de los elementos de modelado descritos, permiten identificar el conocimiento y sus fuentes.

A su vez también se ilustra cómo desde el modelado de los procesos, es posible comenzar a definir una clasificación de los elementos que lo conforman, con el fin de que ésta sirva para definir las taxonomías de lo que se consideran los conocimientos y fuentes que posteriormente puedan ser usadas para estructurar y generar la base de conocimientos del proceso.

Para este ejemplo se ha tomado información obtenida en un grupo dedicado al mantenimiento del software de un centro de investigación en México.

## 4.3 Sistemas de creencias.

Los antecedentes culturales que han servido de base. Algunos de los puntos más importantes son:

Se adopta el criterio de que la inteligencia tiene que ver principalmente con las acciones racionales. Desde un punto de vista ideal, un agente inteligente es aquel que emprende la mejor acción posible en una situación dada. Se estudiará el problema de la construcción de agentes que sean inteligentes en este sentido.

Los filósofos (desde el año 400 A.C.) permitieron el poder pensar en IA, al concebir a la mente, con maneras diversas, como una máquina que funciona a partir del conocimiento codificado en un lenguaje interno y al considerar que el pensamiento servía para determinar cuál era la acción correcta que había que emprender.

Las matemáticas proveyeron las herramientas para manipular las aseveraciones de certeza lógica, así como las inciertas, de tipo probabilista. Así mismo, prepararon el terreno para el manejo del razonamiento con algoritmos.

Los psicológicos reforzaron la idea de que los humanos y otros animales podían ser considerados como máquinas para el procesamiento de información. Los lingüistas demostraron que el uso de un lenguaje ajusta dentro de este modelo.

La ingeniería de cómputo ofreció el dispositivo que permite hacer realidad las aplicaciones de IA. Los programas de IA por lo general son extensos y no funcionarían sin los grandes avances en velocidad y memoria aportados por la industria de cómputo.

La historia de la IA ha sido testigo de ciclos de éxito, injustificado optimismo y la consecuente desaparición de entusiasmo y apoyos financieros. También ha habido ciclos caracterizados por la introducción de nuevos y creativos enfoques y de un sistemático perfeccionamiento de los mejores.

Los avances recientes logrados en la comprensión de las bases teóricas de la inteligencia han ido aparejados con mejoras hechas en el potencial de los sistemas reales.

Como en el presente se han ocupado de la manera adecuada de razonar, aunque en cada caso es diferente lo que se entiende por ello. En el caso de la lógica de primer orden, el razonamiento adecuado significa la obtención de conclusiones a partir de premisas; si la base de conocimientos original representa fidedignamente al mundo, entonces las inferencias también lo representarán fielmente. En el caso de la probabilidad, manejamos creencias, no el estado del mundo, y en este caso "razonamiento adecuado" significa contar con creencias que permiten a un agente actuar de manera racional.

se ha visto una gran variedad de métodos para abordar el problema de la implantación de los sistemas de razonamiento lógico. En este capítulo se observó algo que parece ser un

consenso: el razonamiento eficiente mediante la probabilidad es tan reciente que solamente hay un método -las redes de creencia-, del que existen sólo ligeras variantes. Lo más importante es lo siguiente:

La información sobre la independencia condicional es una forma vital y sólida de estructurar información sobre un dominio incierto.

Las redes de creencia constituyen una manera natural de representar la información sobre la independencia condicional. Los vínculos entre los nodos representan los aspectos cualitativos del dominio; las tablas de probabilidad condicional representan los aspectos cuantitativos.

Una red de creencia es una representación completa de la distribución de probabilidad conjunta correspondiente a un dominio, pero su tamaño es exponencialmente menor.

La inferencia en las redes de creencia implica el cálculo de distribución de probabilidad de un conjunto de variables de consulta, a partir de un conjunto de variables de evidencia.

Las redes de creencia pueden razonar de manera causal, por diagnóstico, de modo combinado o de forma intercausal.

Ningún otro mecanismo de razonamiento bajo condiciones de incertidumbre puede manejar todos los modos anteriores.

La complejidad de la inferencia en una red de creencia dependerá de la estructura de la red. En el caso de los poliárboles (redes con una sola conexión), el tiempo de cálculo es función lineal del tamaño de la red.

Existen varias técnicas de inferencia que son utilizadas en las redes de creencia general, y en todas ellas la complejidad es exponencial en el peor de los casos. En los dominios reales, la estructura local tiende a que las cosas sean más factibles, aunque hay que poner atención para construir una red manejable en donde haya más de un centenar de nodos.

También es posible utilizar técnicas de aproximación, incluida la simulación estocástica, para obtener una estimación de las verdaderas probabilidades haciendo menos cálculos.

Se propusieron diversos sistemas alternos para razonar en condiciones de incertidumbre. En todos los sistemas funcionales de verdad hay serios problemas relacionados con el razonamiento mezclado o intercausal.

## 4.4 Deducción automática

La lógica moderna se desarrollo a partir de la confluencia de Matemáticas, Ingeniería y Lingüística.

Podríamos situar el comienzo de la aportación de la Ingeniería a la Lógica en 1938, cuando Claude E. Shannon (más tarde famoso por su Teoría de la Información) observó que las funciones realizadas por circuitos combinatorios, inicialmente construidos con reles, se podían representar con la notación simbólica del álgebra de Boole. La contribución de la Lingüística llega a finales de los 50. Noam Chomsky, con su teoría de las gramáticas formales, establece las bases de la lingüística matemática e inicia el camino hacia la formalización en la descripción de los lenguajes naturales.

Si lo que se pretende es mecanizar tareas en las que interviene destacadamente la capacidad deductiva, que podemos calificar como en las que se requiere:

- Tener conocimiento sobre el dominio del discurso.
- Razonar con tal conocimiento.
- Conocer como dirigir o guiar tal razonamiento.

Entonces es preciso definir con claridad y precisión, así como analizar desde el punto de vista matemático, los procesos deductivos que el hombre ejercita de modo natural. **Tal es el objetivo de la Lógica**.

Deducción Automática

**La Deducción Automática, es un área de investigación matemática de extraordinario potencial práctico. No en balde distintos autores de conocido prestigio afirman que la Lógica es a la Computación como el Cálculo Infinitesimal es a la Física.**

Hasta bien entrado el siglo XX, casi todos los esfuerzos para automatizar las matemáticas estuvieron dirigidos hacia la mecanización de la aritmética, en lugar de la mecanización del proceso deductivo mismo.

Una vez que los primeros computadoras aparecieron en la década de 1950, un gran número de investigadores intentó automatizar la demostración de teoremas en lógica de predicados.

Desde una perspectiva histórica, podemos identificar tres tendencias principales dentro de la deducción automática:

- Simular el proceso de razonamiento humano.
- Centrarse en la obtención de resultados de manera automática sin usar necesariamente métodos humanos de razonamiento.
- Utilizar enfoques interactivos, en los que el proceso de deducción mecanizada está guiado, en mayor o menor medida, por el usuario.

**Los primeros demostradores**

El crédito de haber sido el primer programa de deducción automática se suele conceder a la Logic Theory Machine, desarrollada por A. Newell, H. Simon, e implementada por J.C. Shaw en 1955, para demostrar los teoremas de la lógica proposicional de los Principia Matemática.

La Logic Theory Machine encontró demostraciones para 38 de los 52 teoremas presentes en el capítulo 2 de los Principia y, en particular, para uno de ellos encontró una demostración mucho más simple que la propuesta por Russell y Whitehead.

El éxito de tales enfoques estuvo limitado tanto por los procedimientos utilizados como por la capacidad de las máquinas disponibles, mucho menos potentes que el más barato computador personal del mercado en la actualidad.

Por otra parte, M. Davis y H. Putnam entre otros, hicieron algunos avances en la automatización de la demostración de teoremas en la lógica de predicados, aunque no presentaron ninguna implementación.

**Métodos basados en resolución.**

A principios de los 1970 Robert Kowalski introdujo la idea, sobre el trabajo de Alan Colmerauer et al., de mecanizar la deducción para solo una cierta clase de formulas llamadas cláusulas de Hornz.

Ya era conocido que la deducción con cláusulas de Horn permite calcular todas las funciones recursivas; sin embargo ni siquiera trabajando en este pequeño fragmento de la lógica de predicados se conseguía una eficiencia suficiente.

Esta se consiguió con la introducción del lenguaje de programación Prolog, cuyo procedimiento de demostración para cláusulas de Horn es conocido como resolución SLD; aunque su método de deducción es lógicamente incompleto, es suficientemente potente como para computar todas las funciones recursivas.

A pesar de todos los avances realizados, la eficiencia de las implementaciones existentes dejaban bastante que desear.

Por otra parte, algunos resultados acerca de la complejidad algorítmica indicaban que incluso la comprobación de la validez de una fórmula de la lógica proposicional es un problema NP-completo [5]. Estos hechos hicieron que gran parte de los investigadores en deducción automática se desvincularan del área, con la excepción de la gente de ANL, o bien se planteasen la utilización de estrategias no basadas en la regla de resolución.

**Métodos no basados en resolución.**

Uno de los impulsores de la deducción automática no basada en resolución de los años 70 fue W.W. Bledsoe . Sus ideas estaban inspiradas en la del proceso deductivo, que no debía encorsetarse en los formalismos de los métodos lógicos de demostración existentes. En particular, no se sintió obligado a buscar métodos completos de demostración, le bastaba que fueran buenos métodos en dominios específicos y que las demostraciones que proporcionarán fueran naturales.

Desde 1992 se celebran conferencias de carácter anual dedicadas específicamente al estudio teórico y a las aplicaciones del método de tablas, que se han mostrado como un formalismo conveniente para la deducción automática tanto de lógicas no clásicas como en la propia lógica clásica. El avance producido en el estudio de los métodos de tablas ha llevado a la conveniencia de publicar tratados monográficos en los que se presenta de manera sistematizada el desarrollo y los distintos aspectos de este enfoque .

Hemos de citar en este punto un nuevo enfoque para la deducción automática, llamado TAS. Es una alternativa a los métodos de resolución y de tablas semánticas.

El punto clave para obtener algoritmos de satisfacibilidad eficientes es la capacidad de controlar el proceso de ramificación, y el enfoque de la metodología TAS (por Transformación de Árboles Sintácticos), introducido en la tesis doctoral de F. Sanz en 1992, esta basado en la reducción de la fórmula tanto como sea posible (obviamente con una complejidad razonable tanto en tiempo como en espacio) antes de aplicar cualquier proceso de ramificación.

Para determinar la satisfacibilidád de una fórmula dada, en primer lugar se intenta reducir el tamaño de la misma mediante la aplicación sucesiva de transformaciones que conservan la satisfacibilidad (no necesariamente el significado), entonces se selecciona una variable respecto de la que ramificar, y se aplica recursivamente el proceso en cada una de las tareas generadas.

Por último, es conveniente resaltar que TAS intenta obtener características sintácticas distintivas de los teoremas, es decir, aquello que los hace válidos; considera que lo más útil es disponer de condiciones suficientes, eficientemente implementables, que aseguren la falsedad o veracidad de una fórmula. Estas condiciones se suelen formular como teoremas de reducción en términos de implicantes e implicados unitarios de las fórmulas.

Esta misma idea puede extenderse a lógicas no clásicas en las que no existe una forma normal conjuntiva, asimismo es suficientemente general porque se puede aplicar a diferentes tipos de lógicas y también es flexible, porque las modificaciones necesarias para cambiar de lógica se pueden realizar de manera modular sin necesidad de rediseñar todo el demostrador. De hecho, aparte de la versión proposicional clásica , existen métodos TAS para Lógicas Temporales y Lógicas Multi-Valuadas.

**Métodos interactivos de deducción automática.**

Otra línea de investigación en deducción automática ha consistido en dirigir el interés hacia sistemas semi-automáticos interactivos que permitan la ayuda externa del usuario en la búsqueda de las demostraciones.

J.R. Guard et al. desarrollaron a mediados de los 70 una serie de sistemas llamados SAM (Semi-Automated Mathemitics) en los que se persigue una combinación de rutinas lógicas automáticas con la intervención humana en tareas de control y de guía.

El desarrollo de sistemas interactivos de deducción automática fue, asimismo, guiado por otra meta de interés para las Matemáticas: la validación automática de pruebas.

Podemos citar en este punto el desarrollo del sistema Automath, junto con la comprobación en 1977, por parte de L.S. van Benthem Jutting, de todas las demostraciones de los Grund lagen der Análisis de Landau.

Pronto se observó que el uso de este tipo de sistemas resultaría especialmente interesante en la verificación de programas, más que en la comprobación de pruebas matemáticas.

Entre los sistemas interactivos más conocidos desarrollados hasta la fecha, podemos destacar el NQTHM de Boyer y Moore.Su principal característica es que está fuertemente basado en la inducción matemática. La dificultad principal de la automatización de pruebas por inducción es la elección adecuada del parámetro sobre el cual realizar la inducción, así como la elección de la hipótesis de inducción más conveniente. La solución propuesta consiste en la prohibición expresa del uso de los cuantificadores en el lenguaje, usándose para tal fin funciones recursivas.

Un descendiente directo de NQTHM es ACL2 (A Computacional Logic for Aplícate Common Lisp) que está formado por un lenguaje lógico junto con un demostrador automático basado en NQTHM. El sistema tiene diferentes facetas pues puede usarse, por ejemplo, como lenguaje de programación, como una lógica matemática formal, o como un demostrador semiautomático.

En relación con la deducción automática, es posible usar ACL2 para demostrares automáticos; Especificando el marco abstracto, es posible obtener versiones ejecutables y formalmente verificadas de, por ejemplo, el método de Davis y Putman y del método de tablas semánticas.

Deducción Automática . . . y Matemáticas

El desarrollo formal de la demostración automática ha necesitado de herramientas puramente matemáticas bien conocidas, en particular de matemática discreta (se hace uso abundante de clausuras inductivas libremente generadas, teorías de árboles y de grafos, etc).

Por su parte, ya hemos visto que algunos problemas abiertos en Matemáticas han sido demostrados mediante el uso de sistemas de deducción automática. Quizá al lector le venga a la memoria la demostración por Appel y Haken en 1977 del teorema de los cuatro colores. En dicha prueba se uso la potencia computacional de un computador para la reducibilidad de un conjunto de miles de configuraciones básicas inevitables. La obtención de dicho teorema no tiene nada que ver con el uso de la capacidad deductiva del sistema usado, sino la aplicación de la fuerza bruta.

Dentro de los teoremas demostrados usando sistemas de deducción automática, destacada la prueba en 1996 de una conjetura de H.Robbins hecha en los años treinta, según la cual todas las álgebras conmutativas y asociativas que satisfacen, además la ecuación de Robbins n(n(x + y)+ n(x + n(y))) = x (donde n representa el complemento) son álgebras de Boole.

Por otra parte, es destacable el demostrador para la Geometría desarrollado por desarrollado por Chou, Gao y Zhang, que ha sido usado para obtener nuevos resultados en geometrías no euclídeas.

Dentro de la verificación formal de resultados matemáticos es conveniente destacar el proyecto Mizar (http://www.mizar.org), que desde 1973 tiene como objetivo construir una matemática formalmente certificada, mediante el desarrollo de un lenguaje formal fácilmente legible por los matemáticos y, simultáneamente, suficientemente riguroso como para que pueda ser procesado y verificado por computador. En conjunción con el J. Of Formalized Mathematics, se esta construyendo una base de datos de la matemática formalizada que en la actualidad contiene más de dos mil definiciones y treinta mil teoremas.

Es conveniente comentar aunque sea brevemente los intentos de aunar la potencia deductiva y la potencia y la potencia computacional que se están realizando dentro del grupo Calculemus (http://www.calculemus.net).

Podemos hacer una extrapolación semejante para la deducción automática en un futuro próximo, puesto que las implementaciones se van adaptando rápidamente a los nuevos modelos de computadores. Este extraordinario desarrollo ha permitido que los sistemas de deducción automática hayan ampliado su horizonte de aplicaciones en áreas tales como el razonamiento sobre programas, razonamiento sobre agentes, y planificación; actualmente existen matemáticos que usan e incrementan conjuntos de lemas, demostraciones, tácticas y estrategias.

**Cada vez se espera más de la Deducción Automática, de modo que sepa reaccionar en función de las distintas peticiones al sistema, deduciendo ciertos datos automáticamente.**

**Es esperable que, finalmente, la deducción automática sea, simplemente, tan útil como es el cálculo en la actualidad y con el mismo amplio campo de aplicaciones.**

## 4.5 Planeación

***Introducción***

La planeación de la tecnología de IA ha estado alrededor por un número de años. En los años 70's y 80's, la comunidad de lenguaje natural adoptó tempranamente sistemas de planeación de IA para asistir en la selección automática y estructuras de contenido en textos multi-sentencial.

Aparte de el GPS, ninguno de los métodos de planeación descritos están interesados como modelos psicológicos, todos los ejercicios están en conseguir la planeación de máquinas, ejercicios, los cuales destaparon más y más dificultades en lo que parecía originalmente un problema simple. La IA se refiere no solamente a inteligencias naturales existentes, sino también a la estructura de todas las inteligencias posibles, existiendo o no. Pero aunque ningunos de estos planificadores "poste-GPS" como modelos cognoscitivos, algunos de ellos han tomado indudablemente la inspiración de solucionar el problema humano. Y nos han permitido clarificar y formalizar los problemas (tales como submetas que obran recíprocamente) que todos los planificadores deben solucionar, como los seres humanos o las computadoras.

Esencialmente, un ***plan*** es un conjunto de acciones que cuando está ejecutada, la voluntad (salvo incidentes) permite a algún agente alcanzar su meta o metas, dado el estado actual del mundo. Operacionalmente la planeación trata de reducir la cantidad de trabajo para tratar de encontrar o producir un resultado correcto.

Lo más simplemente posible, una meta es (como en búsqueda del estado-espacio y el GPS) apenas un nuevo estado del mundo. Pero puede haber otros tipos de meta. El planificador necesita saber las acciones del agente que afectarán el mundo; y puede tener que ocuparse de otros cambios o acontecimientos.

La planeación es un asunto importante en la inteligencia artificial, psicología cognoscitiva, neurobiología, gerencia de negocio, ciencia e ingeniería de fabricación, control basado en los satélites, y un computador principal de otros temas. En la IA, el planear implica la generación de una secuencia de acción o un programa de acción para un agente, tal como un robot o un sistema de software o un artefacto vivo, que pueden alterar sus alrededores. Por ejemplo, el propósito de un plan puede ser dirigir y permitir a un robot traer un objeto, decir A, y lugar encima de otro objeto, decir B.

***Ideas principales de un planificador:***

1. Se ``abre'' la representación de estados, metas y acciones, lo que permite hacer mejor las conexiones entre estados y acciones. Esto evita que realice acciones irrelevantes.

2. El planificador es libre de añadir las acciones al plan cuando se necesitan, en lugar de en forma incremental y secuencial. Esto permite hacer decisiones iniciales obvias y reducir el espacio de búsqueda.
3. Muchos problemas están compuestos de subproblemas independientes, por lo que se puede realizar una estrategia ``divide y vencerás'' (a menos que el costo de combinar soluciones parciales sea muy alto).

Los planeadores utilizan una representación formal, por ejemplo lógica de predicados, y una máquina de inferencia.

Los planificadores en la práctica:

- restringen el lenguaje de representación
- usan un planificador, en lugar de un resolvedor genérico (como un probador de teoremas).

***Planeación involucra áreas principales de IA:***

- ❖ Búsqueda y toma de decisión
- ❖ Representación de conocimiento
- ❖ Aprendizaje

***Tiene sus propios problemas:***

- ❖ Representación y razonamiento temporal, causal y de intenciones
- ❖ Incertidumbre en la ejecución de un plan y considerar al “mundo real"
- ❖ Percepciones y creencias del “mundo real"
- ❖ Agentes múltiples que cooperan / interfieren
- ❖ Restricciones físicas o de otro tipo sobre posibles soluciones

***Aplicaciones:***

| | |
|---|---|
| Control de robots | STRIPS (STanford Research Institute Problem Solver) |
| Generación de programas | HACKER |
| Supervisión de aprendiz de ingeniero mecánico | NOAH (Nets Of Action Hierarchies) |
| Construcción de casas | NONLIN |
| Genética molecular | MOLGEN |

| | |
|---|---|
| Viajes | OPM |
| “scheduling" para producción de turbinas | ISIS-II |
| Misión aviones de carga | SIPE, AIRPLAN |
| Logística naval | NONLIN |
| Secuenciación del Voyager | DEVISER |
| Conversaciones | KAMP |

**Implementaciones específicas**

La interacción entre los subproblemas depende mucho del dominio, las técnicas específicas del dominio no son generalizables. Las técnicas generales no garantizan terminar a menos que se le impongan restricciones

“*la planeación automática* es un programa de computo que involucra la representación del mundo, representación de acciones y sus efectos en el mundo, razonamiento acerca de los efectos de secuencias de tales acciones, razonamiento acerca de las interacciones de las acciones que ocurren concurrentemente, y control de la búsqueda para que se puedan encontrar con eficiencia razonable los planes"

***Modelos:***
*I) Búsqueda de estado-espacio:*

- Define un espacio de estados (espacio con todas las posibles soluciones potenciales) implícita / explícitamente enumerado
- Especifica los estados iniciales, o situaciones de donde empezar
- Especifica los estados finales (metas), o aquellos reconocidos como soluciones
- Especifica las reglas que definen acciones u operaciones disponibles para moverse o ir de un estado a otro dentro del espacio del problema

En este contexto, el proceso de solución de problemas trata de encontrar una secuencia de operaciones que transformen el estado inicial a uno final.

Con suficiente conocimiento, el solucionador puede saber qué hacer directamente. En la práctica, todos involucran algo de búsqueda sistemática.

Se usa el conjunto de reglas junto con una estrategia de control adecuada para moverse en el espacio de búsqueda hasta encontrar un camino o solución del estado inicial al final.

En las estrategias de búsqueda, en general se necesitan heurísticas

II) Descomposición de problemas:

- descripción de la meta del problema
- descripción del estado inicial del problema
- conjunto de descripciones de metas primitivas cuyas soluciones son inmediatas
- conjunto de operadores para transformar metas a submetas (los operadores constituyen las transformaciones en que la descripción inicial del problema puede ser transformada a un conjunto de submetas cuyas soluciones son inmediatas)

  En este contexto, la búsqueda de reducción de problemas es una gráfica AND / OR, mientras que la búsqueda en el espacio de estados es una gráfica OR. En el AND / OR los AND son submetas (las cuales se deben de cumplir todas para lograr la meta) y un nodo OR representa descomposiciones alternas en diferentes submetas.

  En un OR, los nodos OR son divisiones del camino en estados alternativos. En planeación, reducir una tarea, implica encontrar una colección de subtareas tales que la tarea se pueda lograr al hacer todas las subtareas. La colección de esas subtareas entonces representa un plan.

  El escoger conjuntos relevantes de subtareas (OR choices) es un problema por si mismo.

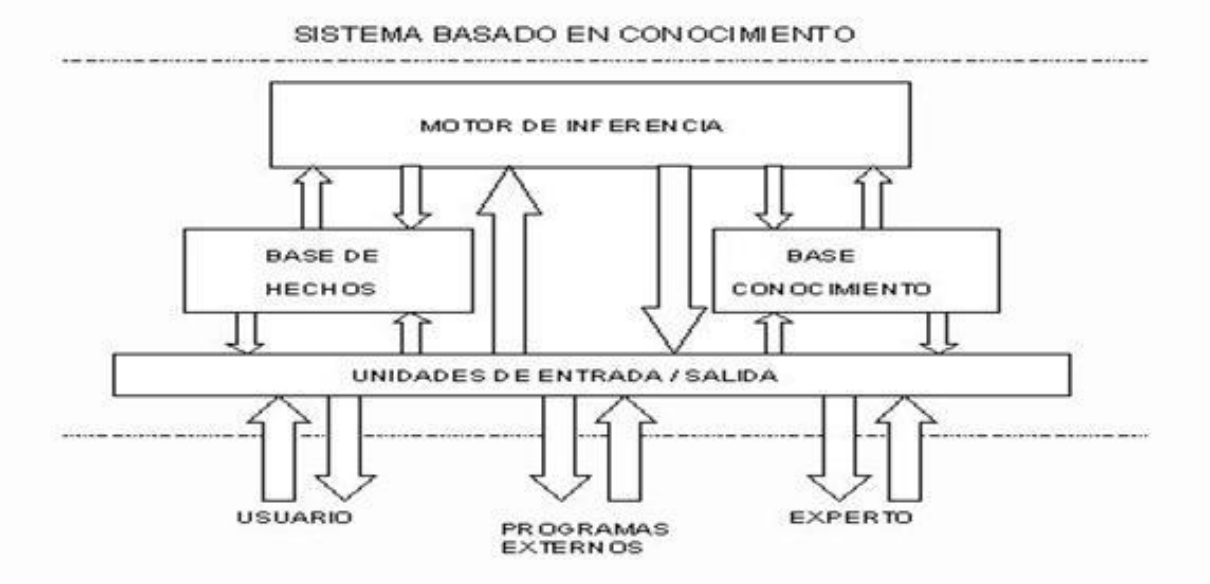
  Muchos sistemas usan apareamiento.

  El resolver un subproblema puede hacer inaplicable el operador requerido para resolver otro subproblema!!

  Planeación queda entre “saber exactamente que hacer" y “búsqueda ciega"

## 4.6 Aprendizaje e inferencia deductiva.

La arquitectura de un sistema basado en conocimiento de alguna manera refleja la estructura cognitiva y los procesos humanos. La primera parte es la memoria de largo plazo, en la que guarda los hechos (Base de Hechos) y los conocimientos (Base de Conocimientos) acerca del dominio en el que tiene experiencia.




**Estructura de un sistema basado en conocimiento**

La segunda parte es el sistema que realiza la función de razonamiento para resolver problemas (Motor de Inferencia). Finalmente, la tercera parte la conforman las unidades de entrada y salida que permiten la comunicación entre el sistema y su entorno.

La representación del conocimiento determina el desarrollo de un sistema experto e influye en las estrategias de control. Es muy versátil, hay muchas formas de hacerlo y es complicada la elección. Generalmente la elección está basada en la intuición o en las especiales circunstancias del problema.

La primera fase es la adquisición del conocimiento desde un experto y depende de la cantidad y de su adecuada representación. Se deben conocer los límites, pues sólo llegan hasta donde son capaces de resolver un problema. Esto implica, que tiene que circunscribirse a un dominio homogéneo.

El lenguaje de programación debe ser adecuado, se usan principalmente el LISP y PROLOG, que difieren de los lenguajes clásicos en que sus reglas (que también contienen hechos) entran masivamente, y que el programa, es decir, el motor de inferencia, encontrará el camino a través de reglas.

Por otra parte, la programación procedural consiste en una cuidadosa descripción de los algoritmos que marcan el camino de los diferentes procedimientos y funciones en los programas.

Estos dos modos de programación son teóricamente iguales, pero en la práctica difieren.

La programación declarativa es un tipo de programación implantada en los sistemas expertos y por ciertos lenguajes. La principal diferencia es que en la declarativa las reglas se formulan independientemente de su secuencia de aplicación. El modo en que se ejecutan las reglas corresponde al motor de inferencia y es independiente de las reglas en sí mismas. En la programación procedural, las reglas se implantan y ejecutan en un orden establecido.

Las ventajas en los sistemas expertos es la modularidad, no hay un solo camino para responder a un problema y son fáciles de leer, lo que simplifica comprobar su consistencia.

La representación del conocimiento es esencial en inteligencia artificial y es la base de la construcción de un sistema experto.

**Fases en la adquisición del conocimiento:**

- identificación del problema.
- Estructuración del conocimiento.
- La arquitectura del sistema y la maqueta.
- El prototipo y terminación del sistema.

**Tecnología de los Sistemas Basados en Conocimiento.**

Desde el punto de vista tecnológico, los Sistemas Basados en Conocimiento pueden presentar varias formas de aplicación:

- **Aislada:** un Sistema Basado en Conocimiento único se relaciona con el entorno.
- **Integrada:** varios Sistemas Basados en Conocimiento conectados a bases de conocimiento comunes. Tipo Front-End cuando todos los hechos y datos están físicamente en la base común. Tipo Back-End cuando los hechos y datos necesarios que no estén en las bases comunes, pueden obtenerse de otros SBC.
- **Embebida:** un Sistema Basado en Conocimiento está integrado con otros sistemas y no se lo distingue.

***Motor de Inferencia***

Un motor de inferencia interpreta y evalúa los hechos en la base de conocimientos para proveer una respuesta. Este debe ser independiente del conocimiento y de los hechos. Se puede caracterizar por:

El lenguaje en que ha sido escrito.

- La velocidad de trabajo: Inferencias / segundo.
- Las estrategias de búsqueda de soluciones:

No Ordenada: aleatoria, heurística.

Ordenada: Encadenamiento hacia adelante (guiado por los datos, deductivo), encadenamiento hacia atrás (guiado por los objetivos, inductivo).

- La forma en que elige el conocimiento.
- La posibilidad de incorporar metaconocimiento.
- El tipo de lógica que emplea en el razonamiento:

Booleana, trivalente, multivalente, difusa.

Monotónica o no monotónica.

Atemporal o temporal.

Lógica de orden 0, orden 0+, orden 1.

- El método que utiliza para la evaluación del conocimiento incompleto o incierto:

Determinístico.

Probabilístico.

Aproximado.

Difuso.

En un sistema experto, dado un estado particular de la base de datos, se deben reconocer las reglas aplicables y esto se llama filtrado o identificación de patrones, operación especialmente larga, ya que muchos sistemas realizan una preselección de las reglas antes de que comience la identificación propiamente tal. Es decir, se considera un subconjunto de reglas y entre éstas, las aplicables. Esto, es lo que se llama restricción.

Adicionalmente puede existir una Base de Datos con información puntual sobre problemas específicos anteriormente resueltos por el sistema, y que se usa como información adicional en el proceso de inferencia.

Una vez que las reglas han sido reconocidas deben ser elegidas, dependiendo de la estrategia de control, que es la selección.

Una vez que las regla han sido elegida ha de ejecutarse.

Al módulo central del sistema experto que maneja estas tres operaciones se le denomina motor de inferencia o intérprete de reglas. Hay diversos tipos de motores de inferencia según los métodos con que se manejan las tres operaciones precedentes.

**Identificación de patrones**

Esta operación determina cuáles son las reglas potencialmente aplicables en un estado dado de la base de datos.

Hay ciertos métodos típicos para resolver problemas mediante Inteligencia Artificial como: descomposición de problemas, inferencia deductiva, deducción lógica, etc.

No es necesario que la búsqueda se aplique a todas las reglas, el sistema de control puede decidir cuáles no son necesarias a ese nivel.

El filtrado significa tener el conocimiento para encontrar las reglas cuya precondición se satisfaga. Se debe comparar por tanto la precondición con la base de datos. Si es una regla deductiva "si premisa entonces conclusión", la parte a ser probada de la regla, se llama disparador o activador (tigger) es siempre una fórmula bien formada ya sea de lógica proposicional o de lógica de primer orden. Puede aplicarse a cualquier hecho.

El único lenguaje que está totalmente basado en el cálculo de predicados es el PROLOG.

Los procesos de identificación de patrones pueden mejorarse examinando, dentro del activador de cada regla, primero los predicados con más restricciones, es decir, los que tienen el menor número de variables libres. Otra forma de proceder es que, para cada regla o parte de su activador, se guarda en la memoria el resultado de las posibles unificaciones para cada parte del activador. Después de cada ciclo del motor de inferencia, esta memoria se actualiza y se denomina filtrado por propagación.

**Ejecución de las reglas**

Después de la fase de filtrado, cuando una regla está reconocida como aplicable, teniendo en cuenta la base de datos existente, solo resta ejecutarla. Si hay varias posibles reglas, la elección la realiza la estrategia de control.

La conclusión de la regla tiene por objeto modificar la base de conocimiento, creando, modificando o suprimiendo un hecho.

**La fase de restricción**

En un sistema dirigido por los datos, el filtrado consiste en retener todas las reglas cuyas premisas son verdaderas teniendo en cuenta los hechos (verdaderos) presentes en la base de datos. Este sistema funciona en encadenamiento hacia delante.

Si el sistema es a la vez dirigido por los datos y por los objetivos, se denomina mixto.

La restricción también puede estar especificado explícitamente por el experto para utilizar reglas dentro de las reglas, es decir, meta reglas. Indica qué grupo de reglas debe ser retenido, por prioridad, o definir un orden en los subconjuntos de las reglas.

El conocimiento puede organizarse en forma de red como en las redes semánticas utilizadas en el análisis sintáctico del lenguaje. Su posición dentro de la red dirige las restricciones utilizando heurísticas. Esta formulación es particularmente eficiente si se establece válidamente una organización jerárquica del conocimiento, en este caso existiría una taxonomía de los hechos.

Otro modo de gobernar las restricciones es organizar las reglas en paquetes o esquemas, lo que genera una estructura de árbol en reglas, lo que es una ventaja.

Todos estos modos dependen de la forma en que está representado el conocimiento.

**La fase de selección**

Como después de las fases anteriores quedan varias reglas se debe seleccionar una y luego de procesada el programa debe saber dónde volver.

Cuando hay una o más reglas, se usan dos estrategias:

- La búsqueda en anchura, que elige una, generalmente la primera y se van ejecutando por turno antes de comprobar que se ha alcanzado la condición final. Si no es así se inicia otro ciclo del motor de inferencia. Este tipo de búsqueda es exhaustiva y la usa el sistema experto MYCIN.

· Otra estrategia es la de elegir una regla entre las aplicables por medio de una heurística, procesarla y modificar la base de datos. Si no se alcanza la condición final, comienza otro ciclo del motor de inferencia. Es por lo tanto, una estrategia en profundidad.

Cada una de las estrategias puede ser irrevocables o avanzar por prueba y error.

Muchos sistemas efectúan marcha atrás, si no hay regla aplicable, pero si el sistema está bajo un control irrevocable se detendrá.

Visto en la práctica, las reglas poseen la siguiente estructura:

SI (CONJUNTO_DE_PREMISAS) ENTONCES CONSECUENCIA

O bien:

(CONJUNTO_DE_PREMISAS) CONSECUENCIA

El CONJUNTO_DE_PREMISAS contiene las cláusulas, o simplemente condiciones, que deben cumplirse para que la regla pueda ejecutarse, y la CONSECUENCIA representa el resultado de la ejecución de dicha regla.

O bien:

SI (x ES CATEDRÁTICO) ENTONCES (x ES DOCTOR)

En este caso CATEDRÁTICO y DOCTOR representan de nuevo relaciones que deben estar representadas sobre el sistema.

**¿Cómo representar estas nuevas relaciones?**

Como todo sistema informático los SE pueden operar sobre un conjunto finito de elementos, para esto es necesario definir el universo de trabajo dentro del cual probaremos nuestras reglas. Sobre esta base la definición de la relación SER CATEDRÁTICO o CATEDRÁTICO, según sea la representación usada, debe hacerse puntualmente para cada individuo del universo, por ejemplo:

CATEDRÁTICO ("Efrén Veloz")

CATEDRÁTICO ("Manuel Fernández")

CATEDRÁTICO ("Maria González")

Los componentes de la relación SER DOCTOR se deducen a partir de la relación SER CATEDRÄTICO y la regla anterior, por lo que no es necesario una representación o definición explícita.

**La inferencia o extracción de conclusiones:**

Para extraer una conclusión, o inferencia, se hace necesario, antes que nada, saber el objetivo a satisfacer, o problema a resolver. Este problema se plantea en forma de objetivo que se toma por el Motor de Inferencia el cual a través de la activación y encadenamiento de reglas trata de llegar a una solución. Al seguir con el ejemplo:

Objetivo:

"Es Efrén Veloz Doctor"?

Solución:

Al aplicar la regla: SI (x ES CATEDRÁTICO) ENTONCES (x ES DOCTOR)

Se debe chequear, como precondición de activación, el cumplimiento de la relación:

CATEDRÁTICO ("Efrén Veloz")

Se debe chequear la base de Axiomas o Afirmaciones:

CATEDRÁTICO ("Efrén Veloz") existe en la base de axiomas, por lo que la precondición es verdadera.

Se dispara la consecuencia y se llega a la conclusión que Efrén Veloz es doctor.

**Encadenamientos de reglas:**

Existen dos mecanismos fundamentales de encadenamiento de reglas:

**1. Encadenamiento hacia delante:**

Se produce cuando el objetivo propuesto al sistema hace que se ejecute una regla, y la conclusión obtenida permite que se ejecute otra, y así sucesivamente hasta llegar a una respuesta, positiva o negativa. El punto final se detecta cuando no se pueden producir más encadenamientos, por ejemplo cuando se llega a un axioma.

Por ejemplo:

(1). SI (x ES JEFE_DPTO) ENTONCES (x ES CATEDRÁTICO)

(2). CATEDRÁTICO ("Efrén Veloz")

(3). CATEDRÁTICO ("Manuel Fernandez")

(4). CATEDRÁTICO ("Maria Gonzalez")

Al evaluar el objetivo: "Efrén Veloz es Jefe_Dpto"?, se dispara la regla (1), la cual a su vez se encadena con la (2), en este momento no se pueden producir más encadenamientos pues la regla (2) es un axioma. Llegado a este punto el Motor de Inferencia retrocede y da una respuesta positiva a la pregunta.

**2. Encadenamiento hacia atrás:**

Consiste en, dado un objetivo, buscar una regla que permita establecer dicha conclusión, el proceso se repite hasta encadenar con la regla cuya conclusión satisfaga el objetivo propuesto, o se detecte que dicho problema no se puede resolver positivamente. Por ejemplo para averiguar si "Efrén Veloz es doctor" se busca un regla que tenga esta afirmación en sus consecuencias. Analizando las reglas anteriores vemos que la regla:

(5). SI (x ES CATEDRÁTICO) ENTONCES (x ES DOCTOR)

Satisface estas condiciones; siguiendo esta regla hacia atrás tenemos que buscar una nueva que permita validar si "Efrén Veloz" es catedrático, lo cual se hace con el axioma (2).

Los encadenamientos constituyen, de esta manera, una de las herramientas fundamentales del Motor de Inferencia; el Lenguaje PROLOG solamente soporta el encadenamiento hacia atrás, el cual constituye el más frecuentemente implementado.

# UNIDAD V
# APLICACIONES ESPECÍFICAS

## 5.1 Visión, algoritmos para visión.

### ALGORITMOS DE VISION

*Definición de la Visión por Computadora desde el punto de vista de la Inteligencia Artificial.*

El término "Visión por Computadora" (Vision Algorithms) dentro del campo de la Inteligencia Artificial puede considerarse como el conjunto de todas aquellas técnicas y modelos que nos permitan el procesamiento, análisis y explicación de cualquier tipo de información especial obtenida a través de imágenes digitales.

Desde sus inicios la Visión por Computadora ha inspirado sus desarrollos en el estudio del sistema visual humano el cual sugiere la existencia de diferentes tipos de tratamiento de la información visual dependiendo de metas u objetivos específicos, es decir, la información visual percibida es procesada en distintas formas con base en las características particulares de la tarea a realizar, por lo que la visión por computadora propone varias técnicas que permiten obtener una representación del mundo a partir del análisis de imágenes obtenidas desde cámaras de video.

Debido a que la información visual es una de las principales fuentes de datos del mundo real, resulta útil el proveer a una computadora digital del sentido de la vista (a partir de imágenes tomadas con cámaras digitales o analógicas), que junto con otros mecanismos como el aprendizaje hagan de esta una herramienta capaz de detectar y ubicar objetos en el mundo real, objetivo principal de la Visión por Computadora.

El análisis del movimiento en imágenes está creciendo en importancia en numerosas aplicaciones. Algunas de estas aplicaciones son: (a) la televisión en cuanto a la codificación de las imágenes de vídeo mediante compensación, (b) la robótica móvil, ( c ) el tratamiento de las imágenes de satélite, ( d) las aplicaciones, tanto civiles como militares, de seguimiento de objetivos y navegación autónoma, (e) el tratamiento de las imágenes biológicas y médicas, (t) la vigilancia de lugares y su supervisión, y,(g) interfaces y realidad virtual (Mitiche y Bouthemy, 1996).

El problema de la detección de movimiento es particularmente interesante cuando el objetivo que se persigue es el de la localización espacial de los objetos móviles en la escena. Esta detección de movimiento siempre está fuertemente ligada a la detección de cambios temporales en la imagen.

Cuando existen objetos en movimiento en una escena, siempre tendremos cambios en la intensidad de los píxeles de la imagen.

Este hecho ha dado lugar a una extensa bibliografía, en la que destacamos algunos trabajos ya clásicos con diferentes enfoques. Para un estudio más extenso del tema, se recomienda la lectura de Fernández (1993) y Fernández-Caballero (1997).

El seguimiento de elementos de una imagen a otra es un procedimiento común, sobre todo en aplicaciones de vigilancia. Algunos procesos de seguimiento han sido definidos por medio de:

(a) un modelo representativo de elementos (por ejemplo, las coordenadas de imagen de algunos puntos característicos, la longitud y la orientación de los segmentos del contorno ),
(b) un modelo cinético de la evolución de los elementos (por ejemplo, una velocidad constante, una aceleración constante, un movimiento afín),
( c ) un conjunto de relaciones entre los parámetros del modelo y las medidas de la imagen, y,
( d) un filtro temporal para la estimación de los parámetros del modelo a partir de los datos de la imagen.

*Técnicas de Detección de Movimiento.*

Las distintas técnicas de detección de movimiento pueden clasificarse de muchas maneras diferentes. Nuestro enfoque contempla una primera división en métodos 3-D y 2-D. Los métodos 3-D intentan resolver las ecuaciones de las proyecciones utilizando directamente las correspondencias entre las características entre las tramas, mientras que los métodos 2-D estiman el flujo óptico.

*Estimación del Movimiento 3-D.*

El problema de estimar el movimiento 3-D a partir de un conjunto de proyecciones 2-D de la escena es un asunto complejo. La estimación 3-D, en general, se aborda desde dos ópticas distintas: la estimación basada en la correspondencia de características (Huang y Netravali, 1994) y los métodos directos (Hom y Schunck, 1981).
Los métodos directos estiman el flujo óptico 2-D y deducen el movimiento 3-D a partir de un análisis cuidadoso del campo de flujo estimado, tras haber aplicado toda una serie de restricciones adicionales para simplificar el problema. Los métodos directos 3-D pueden considerarse, por tanto, como una extensión de los métodos 2-D de flujo óptico.

Los métodos de correspondencia de características extraen un conjunto altamente discriminante de rasgos en las imágenes y deducen el movimiento resolviendo las ecuaciones de las proyecciones después de haber determinado algunas correspondencias, haciendo un uso explícito de la restricción de rigidez.

El problema se descompone en tres pasos: el descubrimiento de las características, la determinación de las correspondencias y la deducción del movimiento y de los parámetros estructurales, dadas las posiciones de los rasgos correspondientes. Algunos ejemplos típicos incluyen puntos, bordes y esquinas (Huang y Netravali, 1994; Liu y Huang, 1988; Liu y HUallg, 1989).

El problema de la correspondencia de características puede verse como un problema de búsqueda. Las técnicas de filtrado temporal, como en el caso del filtrado de Kalman (Deriche y Faugeras, 1990; Faugeras e! al., 1987), se aplican a menudo en el seguimiento a través del tiempo de características con el fin de reducir el espacio de búsqueda asociado a cada rasgo.

Muchos de los algoritmos aplicados hasta la fecha asumen que las correspondencias se conocen de antemano.

Debemos tomar en cuenta que la correspondencia de puntos fue usada por los investigadores Ullman (1979) y Roach y Aggarwal (1980). El segundo de los enfoques, a modo ilustrativo, contemplaba la resolución de un conjunto de 20 ecuaciones con 27 incógnitas.

La utilización de correspondencias de líneas complica aún más el problema. Puede demostrarse que el movimiento 3-D no puede determinarse de la correspondencia de líneas en dos proyecciones solamente (Weng el al., 1992). El uso de correspondencias de líneas lleva generalmente a un conjunto grande de ecuaciones no lineales que sólo pueden resolverse por métodos iterativos (Holt y Netravali, 1993).

Aunque existe una formulación lineal, se produce un considerable número de soluciones espurias (Liu y Huang, 1989).

*Estimación del Movimiento 2-D.*

La mayoría de las aplicaciones reales de estimación del movimiento trabajan, de alguna forma, con un acercamiento 2-D, en el que se obtienen estimaciones del movimiento en el tiempo del movimiento 2-D de patrones de intensidad (Aggarwa y Nandhakumar, 1988; Barron etal., 1994).

En estos métodos encontramos la restricción de que el brillo de la imagen a lo largo de la trayectoria del movimiento es constante, dicho en otras palabras, que cualquier cambio en el tiempo en la intensidad de un punto es debido únicamente al movimiento.

Dos visiones diferentes del mismo problema han llevado a dos filosofías diferentes en la estimación del movimiento 2-D: la utilización directa de la ecuación del flujo óptico, y, el cálculo del mínimo de la diferencia del desplazamiento entre tramas en un conjunto de regiones locales, buscando en un conjunto de vectores de

movimiento. Basándose en estas dos ideas, se han propuesto múltiples algoritmos de estimación de movimiento 2-D:

a) Los métodos basados en el gradiente, donde se usa directamente la ecuación del flujo óptico, estimando los gradientes espacial y temporal en cada punto de la imagen, e imponiendo alguna restricción adicional para identificar el movimiento de un modo unívoco.
a. Los métodos de correspondencia de regiones, donde la diferencia del desplazamiento entre tramas (o cualquier otro criterio de error similar) se minimiza sobre un conjunto de regiones locales empleando algún mecanismo de búsqueda apropiado.
b. Los métodos en el dominio de frecuencias, en los que se explota la invarianza en el desplazamiento y las propiedades de simetría de la transformada de Fourier para estimar el movimiento.
c. Los métodos bayesianos, en los que se minimiza globalmente la diferencia del desplazamiento entre tramas por medio de técnicas de optimización estocásticas y el modelado explícito de las propiedades estructurales del campo de movimiento.

*Métodos Basados en el Gradiente.*

La estimación basada en el gradiente se ha convertido en el principal acercamiento en las aplicaciones de visión por computadora. Ello se debe principalmente a que es computacional mente eficaz ya que se produce una buena estimación del campo de movimiento.

Los métodos basados en el gradiente, para poder usarse en la estimación de movimiento, requieren de algunas restricciones. Se han propuesto muchas de estas restricciones, aunque normalmente ligadas a alguna forma de restricción de suavizado.

El método basado en el gradiente más conocido es el de Hom y Schunck (1981). Su método impuso una restricción de suavizado global en el campo del movimiento. Un ejemplo de método que impone una restricción de suavizado local en el campo del movimiento fue propuesto por Lucas y Kanade (1981).

Se han realizado igualmente algunos esfuerzos para obtener modelos de alto nivel a partir del campo de flujo óptico por medio de técnicas de regresión lineal. Este acercamiento es uno de los más comunes en los modelos de movimiento afín (Black y Anandan, 1996; Meyer y Bouthemy, 1994; Wangy Adelson, 1994).

Las desventajas comunes a todos los métodos basados en el gradiente provienen de los lógicos cambios en la iluminación (Keamey eta/.,1987; Verri y Poggio, 1989) y en la dificultad de abordar numéricamente la diferenciación de la función de intensidad de la imagen.

*Métodos Basados en la Correlación de Regiones.*

Obviamente, una forma de evitar las limitaciones de los métodos basados en el gradiente está en considerar regiones de la imagen en lugar de estimar el movimiento píxel a píxel. En general, los métodos basados en la correlación de regiones son menos sensibles al ruido al tomar más datos de la imagen en el proceso de estimación del movimiento.

Las técnicas de correlación se basan en minimizar la diferencia en el desplazamiento entre tramas en un bloque de píxeles. Representan el acercamiento a la estimación de movimiento más ampliamente usado hoy en día (Jain y Jain, 1981 ).

En su forma básica, el algoritmo de correlación de bloques, divide una imagen en una serie de regiones de igual tamaño. Para cada una de las regiones se busca, en la siguiente trama, la posible correlación en su vecindad, minimizando un criterio de error como la diferencia en el desplazamiento entre tramas, u otra medida relacionada, sobre un conjunto de vectores de movimiento.

Al trabajar con un tamaño fijo de bloque, se está limitando el campo del movimiento estimado, ya que no hay posibilidad de manejar bloques que contengan varios movimientos.

Con el fin de superar esta limitación, algunos investigadores han trabajado con correlación de bloques en entornos de multirresolución. La correlación de bloques en multirresolución puede ser clasificada, a su vez, en dos grandes grupos.

El primer grupo utiliza la correlación de bloques en una pirámide gaussiana (Burt y Adelson, 1983). La idea consiste en que los movimientos mayores pueden estimarse utilizando versiones suavizadas y submuestreadas de las imágenes (Anandan, 1989).

Este tipo de métodos de multirresolución se denomina también de multiescalado. Las estimaciones más burdas se propagan por la pirámide hacia abajo, donde son refinadas

El segundo grupo, a veces llamado de técnicas multimalla, usa correlación de bloques en una estructura de quadtree, creando un mosaico de bloques espaciales de diferentes tamaños (Dufaux y Moscheni, 1995). Este acercamiento permite que la región analizada se adapte a los datos subyacentes.

También se han presentado generalizaciones a la correlación anteriormente descrita. La idea es crear un tamiz adaptable del plano de la imagen y perturbar este tamiz para minimizar una medida de error sobre las regiones (Bradshaw y Kingsbury, 1997; Nakaya y Harashima, 1994; Seferidis y Ghanbari, 1993).

Otros investigadores han usado la transformada de Hough para la estimación del movimiento (Adiv, 1985; Bober y Kittler, 1994 ). La transformada de Hough puede

concebirse como una técnica de clasificación en el espacio definido por los parámetros que describen la forma. El acercamiento mediante la transformada de Hough puede modificarse para trabajar directamente sobre los niveles de gris de la imagen (Bober y Kittler, 1994 ).

Una desventaja compartida de los esquemas basados en la correlación de regiones es su dependencia del modelo de movimiento traslacional. Las regiones deben permanecer bastante pequeñas para el que el modelo traslacional permanezca válido.

*Métodos en el Dominio de Frecuencia.*

Los métodos en el dominio de frecuencias caen en dos grandes clases: los métodos basados en las fases y los métodos basados en la energía.

Los métodos basados en las fases se aprovechan del hecho de que un cambio en el dominio espacial produce un cambio de la fase en el dominio de frecuencias (Papoulis, 1984 ), mientras que los métodos basados en la energía trabajan con la distribución de energía en el espacio de frecuencias al considerar el movimiento en sus orientaciones espacio-temporales.

Se han usado técnicas basadas en las fases para la estimación de la disparidad, sobre todo en el contexto de la estereoscopía (Jepson y Jenkin, 1989; Wilson y Knutsson, 1989). Aunque el problema de la estereoscopía es similar al problema de la estimación del movimiento, se basa en la suposición de que las disparidades son I-D, o sea, horizontales.

Para otros investigadores como Jepson y Jenkin (1989) el concepto de la disparidad, se define en términos de la diferencia de fase en un punto dado entre sus valores filtro paso-banda de dos tramas.

Otro acercamiento a la estimación del movimiento basado en fases proviene de Fleet y Jepson (1990), donde se aplica una técnica basada en el gradiente a la componente fase de las salidas de un conjunto de filtros direccionales paso-banda sintonizados a diferentes velocidades.

Calway et al. (1992) desarrollaron en conjunto un enfoque al dominio de frecuencias basado en una técnica de correlación, realizando trabajo en equipo y comparando resultados.

Su método también se lleva a cabo dentro de un entorno de multirresolución, denominado Multiresolution Fourier Transform (MFT) (Wilson et al., 1992).

La correlación en el dominio espacial es muy costosa, pero, llevada al dominio de frecuencias, la carga computacional se torna manejable. Otro método descrito es el de correlación de fases (Kuglin y Hines, 1975), donde el producto de los espectros se normaliza antes de aplicar la transformada de Fourier inversa, obteniéndose en

el caso ideal un impulso en la posición correspondiente al vector asociado al movimiento.

Otra técnica en el dominio de frecuencias es la del filtrado espacio-temporal (Adelson y Berger, 1985; Heeger, 1987). La idea básica se basa en la observación de que el movimiento se corresponde con la orientación en el espacio- tiempo.

Los métodos en el dominio de frecuencias presentan algunas ventajas notables respecto de muchos métodos en el dominio espacial. En primer lugar, son menos sensibles a los cambios en la iluminación global, y, en segundo lugar, son bastante robustos con respecto al ruido.

*Métodos Bayesianos.*

El problema de la estimación del movimiento puede formularse como un problema de estimación global bayesiano. La idea general consiste en usar un criterio de máximo a posteriori (MAP) para maximizar la probabilidad del campo de movimiento, dada la observación de la intensidad de la imagen en la próxima trama.

La estimación del MAP puede lograrse por medio de los métodos estocásticos de relajación, tales como el temple simulado, que garantizan la convergencia en un máximo global. Los métodos bayesianos han evolucionado a partir de los métodos recursivos sobre píxeles) (Robbins y Netravali, 1983) y normalmente se formulan usando campos aleatorios de Markov (MRF) (Spitzer, 1971).

Los enfoques bayesianos a la estimación del movimiento son de gran interés al enfrentarse directamente con una de las dificultades fundamentales: cómo segmentar el campo de movimiento en regiones de movimiento coherente trabajando píxel a píxel. Sin embargo, padecen de la lentitud en la convergencia inherente a los métodos estocásticos de relajación.

*Métodos de Detección de Contornos de Algoritmos de Visión.*

Para el análisis de imágenes digitales la extracción de los bordes de dichas imágenes resulta de gran utilidad. Podemos considerar que existe un borde cuando en un conjunto de puntos de la imagen aparece un cambio significativo en la intensidad luminosa (Sanfeliu,87).

Existen básicamente dos formas de obtener los bordes de una imagen, estas son:

1. - Método basado en las primeras derivadas.

   En este método primero se miden los cambios de intensidad en todos los puntos de la imagen, haciendo uso de la primera derivada, después se seleccionan como puntos de bordes aquellos puntos en los que el cambio de intensidad rebasa algún umbral preestablecido.

Una imagen puede representarse matemáticamente por una función f cuyo valor en el punto (x, y) representa la iluminación existente en ese punto, el cambio de iluminación de un punto en la imagen esta representado gráficamente por la pendiente que allí tiene la superficie que representa a f, y normalmente resulta diferente según sea la dirección considerada, la dirección en la que el cambio es más intenso esta dada por el vector gradiente en ese punto.

*Tamaño y Localización de un Objeto.*

Uno de los aspectos primordiales de la estructura de un objeto es su tamaño. Conocer el tamaño de un objeto en términos absolutos es útil para el reconocimiento del objeto (Mutch, 1986; Mutch y Heiny, 1986).

A su vez, el aspecto predominante para el cálculo de la velocidad de un objeto seria su localización. Un objeto en traslación o dilatación (no en rotación) puede reducirse a efectos de cálculo del movimiento a la obtención de esos dos parámetros: su localización y su tamaño en todo instante de tiempo t. Véase que cuando hablamos de dilatación queremos englobar igualmente la contracción.

Nuestra propuesta es que ambas variables pueden obtenerse de un modo sencillo, conocida la silueta S del objeto en el instante t (Fernández-Caballero et al., 1999). Para ello vamos a definir, por un lado, el tamaño a partir de la longitud de dos líneas rectas ( o cuerdas) determinadas por cuatro puntos conocidos de la superficie del objeto. Los puntos a los que hacemos referencia (x1, y1), (x2 , y2), (x3, y3) y (x4, y4), tales que:

$$\forall (x,y) \in S(i,j,t), \quad x_1 < x$$

$$\forall (x,y) \in S(i,j,t), \quad x_2 > x$$

$$\forall (x,y) \in S(i,j,t), \quad y_3 <$$

$$\forall (x,y) \in S(i,j,t), \quad y_4 > y$$

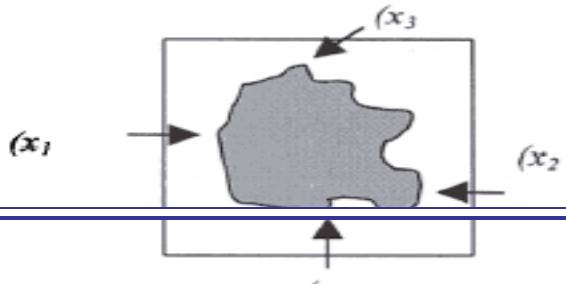
Dicho de otro modo, los cuatro puntos son:

(X1, y1}: punto más a la izquierda del objeto en la imagen

(X2, y2} : punto más a la derecha del objeto en la imagen

(X3, y3): punto más arriba del objeto en la imagen

(X4, y4): punto más abajo del objeto en la imagen



Las dos cuerdas, que denominamos segmentos de línea máximos del objeto, no van a unir los puntos (x1, y1) y (x2, y2), (X3, y3) y (x4, y4) entre sí, respectivamente, sino sus proyecciones (x1, 0) y (x2 ,0), (0, y3) y (0, y4), respectivamente, como puede apreciarse en la figura 2.

En cuanto a la localización del objeto, ésta vendrá determinada por un pu.'1to único característico del mismo (xobj, yob), a saber la intersección de los dos segmentos (x1, y3), (x2, y4) y (x2, y3), (xj, y4j). Este punto será denominado punto representativo del objeto.

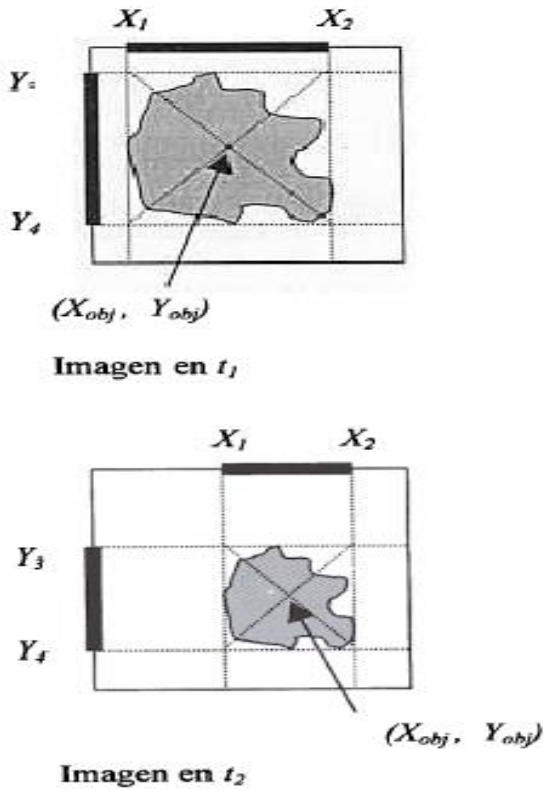
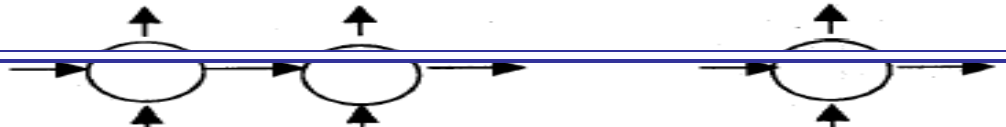



Figura 2. Obtención de los segmentos de línea máximos y del punto representativo del objeto.

## *Implementación Neuronal.*

Esta neurona tiene como característica primaria el poder ser unida en serie con otras neuronas del mismo tipo a través de las señales *INH* del siguiente modo:



Se propone una implementación hardware que obtenga los valores $X_1$, $X_2$, $Y_3$ e $Y_4$ en tiempo real. Para ello se propone implementar una estructura neuronal multifuncional (Alevizos *et al.*, Safonova *et al.*, 1996) de fácil manejo. Partimos de la estructura básica de neurona de la figura 3.

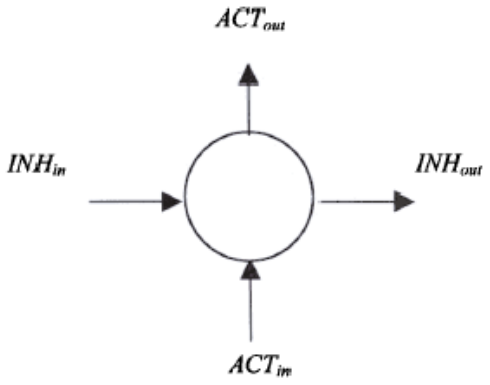
En dicha figura tenemos:

$INH_{in}$ = señal de inhibición procedente de la neurona anterior

$ACT_{in}$ = señal de activación de entrada de la neurona

$INH_{out}$ = señal de inhibición hacia la neurona Posterior

$ACT_{out}$ = señal de activación de salida de la neurona




determinada que cumpla la condición adecuada para transmitir a partir de ese momento el valor 1 (inhibir).

En el caso concreto que nos ocupa, se trata de detectar los valores $X_1$, $X_2$, $Y_3$ e $Y_4$ anteriormente descritos. Para ello se propone utilizar cuatro hileras de neuronas dispuestas según la figura 5, donde las señales $ACT$ (tanto $ACT_{in}$ como $ACT_{out}$) tienen como finalidad el pasar a través de la neurona la información a ser procesada desde un nivel inferior (obtención de la silueta del objeto) a un nivel superior (cálculo de la trayectoria del objeto).

El algoritmo se presenta aquí para el caso de las neuronas detectoras de las filas $Det_{X1}$ o $Det_{X2}$. Para los casos $Det_{Y3}$ o $Det_{Y4}$, cámbiese $i$ por $j$, fila por columna, y viceversa.

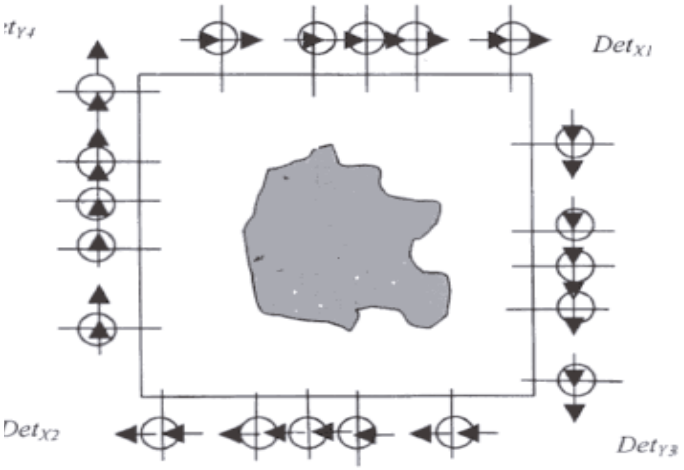



**Figura 5. Determinación de los valores $X_1$, $X_2$, $Y_3$, e $Y_4$.**

$$ACT_{in}(i,t) = \begin{cases} 1, & si \sum_{j=1}^{k} S(i,j,t) > 0 \\ 0, & en\ caso\ contrario \end{cases} \qquad (1)$$

**Esta ecuación (1) viene a decir que la neurona de la columna *i* tiene una activación en su entrada si cualquiera de los pixeles de la imagen de la columna *i* y filas *j* está incluido en la silueta del objeto.**

$$INH_{out}(i,t) = \begin{cases} 1, & si\ INH_{in}(i,t) = 1 \cup ACT_{in}(i,t) = 1 \\ 0, & en\ caso\ contrario \end{cases} \qquad (2)$$

**Se va propagando a través de la línea de neuronas la señal $INH_{out}$ con valor 0 hasta que ocurre uno de dos posibles eventos (no tienen por qué ser excluyentes entre sí): (a) a la neurona llega un valor de inhibición de 1, o, (b)**

## *Seguimiento de Movimiento.*

Una vez que se han obtenido los segmentos de línea máximos y el punto representativo de un objeto en una secuencia de imágenes, es bastante sencillo detectar un gran número de casos de movimiento. Si consideramos las siguientes posibilidades:

(a) no hay movimiento (N),

(b) traslación en el eje X o Y (T),

(c) dilatación o traslación en el eje Z (D), y,

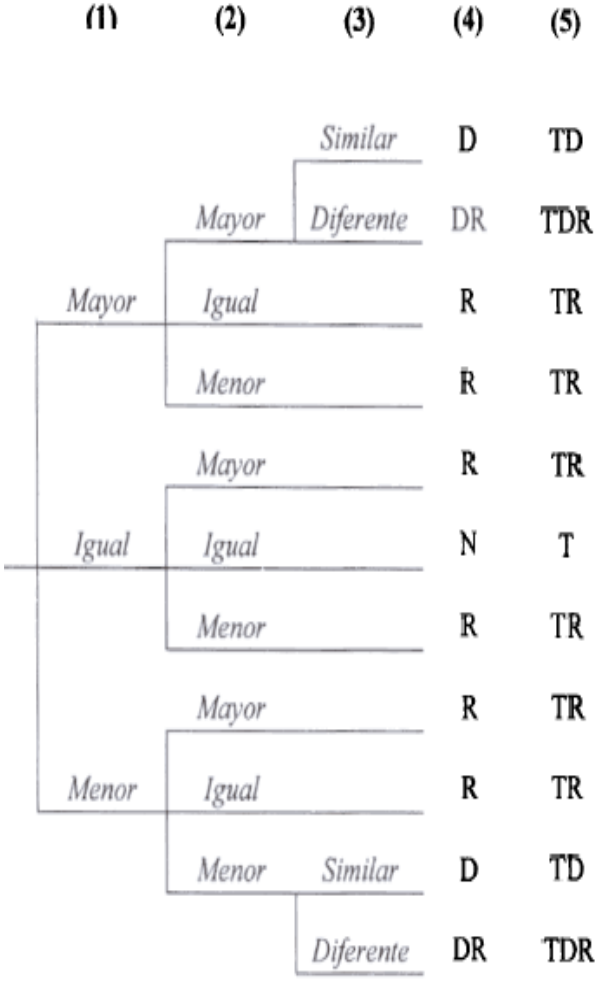
(d) rotación (R)

podemos únicamente obtener por combinación los siguientes casos de movimiento:

(A) N no se detecta movimiento

(B) T detectada una traslación pura

(C) TD detectada una traslación más una dilatación




Grafo Evaluación del movimiento.

(D) TR detectada una traslación más una rotación

(E) TDR detectada una traslación más una dilatación más una rotación

(F) D detectada una dilatación pura

(G) DR detectada una dilatación más una rotación

(H) R detectada una rotación pura

Consideramos que los estados anteriores aparecen en la mayoría de los casos tal como se muestra en el grafo 1. El grafo 1 muestra en la diferentes posibilidades dependiendo de la evolución entre dos imágenes consecutivas de los dos parámetros fundamentales estudiados hasta el momento, a saber el tamaño y la localización del objeto en movimiento.

(4) estado resultante si el punto representativo del objeto no ha cambiado sustancialmente; un cambio no sustancial se obtiene por medio del algoritmo

$$(X_{obj\,k} - X_{obj\,k-1} \leq d) \cap (Y_{obj\,k} - Y_{obj\,k} \leq d)$$

siendo $d$ el desplazamiento máximo permitido

(5) estado resultante si el punto representativo del objeto sí que ha cambiado suficientemente; se trata de la condición negada del punto anterior

Por supuesto, asumimos la posibilidad de ofrecer algunos resultados erróneos, especialmente ante algunos ejemplos que incluyan rotaciones. No obstante, si el número de imágenes es suficiente en la secuencia de imágenes, el error tenderá a ser pequeño.

## *Aplicaciones de los Algoritmos de Visión.*

Entre las aplicaciones en las que la Visión por Computadora ofrece importantes aportaciones, destacan las siguientes:

- En el ramo industrial se aplica para la inspección automatizada; permite identificar que las piezas no tengan defectos.
- En la geología permite estudiar las modificaciones de los causes de los ríos; también sirve para detectar bancos de pesca o variaciones en las corrientes marinas por cambios de clima.

- En la medicina facilita el diagnóstico automático de las enfermedades del corazón, a partir de fotogramas del movimiento del mismo y de un análisis de las deformaciones que se producen.
- En el ámbito militar se desarrollan complejas armas como misiles que se guían por sí solos por zonas previamente grabadas; en la identificación de aeropuertos, barcos, tanques o cualquier imagen tomada desde aviones o satélites.
- Entre las operaciones básicas con las que se construyen diversas aplicaciones específicas de Visión por Computadora, podemos destacar las siguientes:

    1. Identificación de objetos (por ejemplo identificar personas a partir de su fotografía).
    2. Determinación de la posición en que se halla un objeto.
    3. Establecimiento de relaciones espaciales entre varios objetos. (Para guiar a un robot en el ensamblaje de piezas).
    4. Determinación de las coordenadas de puntos importantes de un objeto.
    5. Realización de mediciones angulares.
    6. Mediciones tridimensionales (por ejemplo para ayudar a un robot a librar obstáculos)

## 5.2 Comprensión del lenguaje. (Lunar, Shrdlu, Margie, Sam, Pam, Hearsay, Harpy, Hwim, SRI/SDC)

El programa diseñado por Winograd (1972), conocido como SHRDLU es un extraordinario ejemplo de este comportamiento. SHRDLU es un programa que vive en un mundo imaginario de figuras geométricas en color, que es capaz de manipular estas figuras sobre una superficie cuando se le solicita, que representa a sí mismo este mundo, y que puede responder preguntas sobre su vida en este mundo. Un ejemplo de un diálogo con SHRDLU es el siguiente (de nuevo las respuestas de la computadora en mayúsculas):

¿Tocaste alguna pirámide antes de tocar el cubo azul?

SI, LA VERDE

¿En qué momento?

MIENTRAS APILABA EL CUBO ROJO PEQUEÑO ENCIMA DEL CUBO ROJO GRANDE Y DEL CUBO VERDE

¿Por qué la tocaste?

PORQUE NECESITABA MOVERLA PARA COLOCAR EL CUBO ROJO PEQUEÑO EN SU LUGAR

¿Para qué?

PARA PONER EL CUBO ROJO PEQUEÑO DONDE ESTA AHORA

¿Por qué?

PORQUE TU ME LO PEDISTE...

Las respuestas del programa son una muestra de la claridad con la que representa y pera en su mundo imaginario, incluyendo al interlocutor que da las órdenes, basándose únicamente en esta idea de estructuras de datos. Ninguna otra explicación ha llegado tan cerca de modelar el proceso de representación interna.

Otro problema en el que la IA ha sido de gran utilidad teórica se refiere a la discusión sobre la existencia y la naturaleza de las imágenes mentales. He discutido este problema en detalle en otro artículo (Elguea, 1983) de manera que referiré al lector interesado a este artículo y sólo mencionaré que el problema de las Imágenes mentales se encuentra íntimamente relacionado con el de la representación y que el trabajo en IA ha inclinado a la teoría psicológica en favor de la eliminación de las imágenes como concepto teórico y la aceptación de las proposiciones como el vehículo del pensamiento y la representación.

A principios de los sesentas, la IA comienza una fase distinta de su desarrollo. En 1962, McCarthy y Raphael inician sus trabajos sobre el diseño y la construcción de un robot móvil que llamarían "Shakey".

La diferencia fundamental entre este robot y los programas en computadora utilizados hasta ahora por la IA, es que "Shakey" tendría que enfrentar el reto de interactuar con el mundo real en términos de espacio, tiempo, movimiento, etc. En otras palabras, el robot tendría que tener alguna forma de "conocimiento" del mundo que lo rodeaba.

Este reto inició una fuerte preocupación en la IA por el estudio de la epistemología y los procesos cognoscitivos. La discusión se centró alrededor de los problemas de la representación mental o interna del conocimiento, la percepción y los problemas del significado. La idea básica de Raphael era la de reunir, en una sola, distintas máquinas con capacidad de aprender por experiencia, de reconocer patrones visuales, de modelar, de manipular símbolos, etc., y esperar que el todo fuera mayor que la suma de las partes.

El resultado del experimento no fue el éxito que Raphael esperaba, pero fue un logro sin precedente que hizo posibles avances muy importantes.

El aprendizaje más importante de esta experiencia fue la comprensión de que el problema más difícil por resolver en IA era el de construir una máquina capaz de funcionar con altos ni<,eles de incertidumbre, como lo hace un ser humano.

Se hizo claro que construir una máquina que no lidiara efectivamente con la incertidumbre sería una de dos: o trivial, por la simpleza de la tarea, o imposible por la complejidad de la misma.

Hacia mediados de los sesentas la IA se convierte en un área en la que se interesan e interactúan especialistas de diversas disciplinas: lógicos, psicólogos, matemáticos, lingüistas, filósofos, etc. Uno de los grandes temas de IA en la década de los 90´s fue el estudio del lenguaje. En la mayoría de los estudios iniciales sobre lenguaje, se atacó el problema de diseñar una máquina que fuera capaz de traducir de un idioma a otro.

El énfasis se hizo en el análisis de la sintaxis, en lugar del significado, estrategia que se abandonó relativamente pronto. Los investigadores interesados en esta área de la IA pronto descubrieron que traducción no es lo mismo que transformación, y que, como consecuencia de ello, de alguna manera la máquina tendría que "entender" un texto antes de poder traducirlo.

Los esfuerzos se orientaron hacia una definición de la comprensión que pudiera simularse en una computadora.

Con esta idea como lineamiento heurístico, se desarrolló una serie de programas que exploraban esta serie de habilidades relacionadas con el lenguaje y la comprensión: el "oráculo" de Phillips, "Sad Sam" de Lindzay que fue uno de los más exitosos, y "Eliza" de Wizenbaum y Colby. (ver Boden, 1977). El programa de Lindzay se orientó hacia el estudio del significado, connotativo y denotativo, y es capaz de hacer inferencias, conocer, y parafrasear sobre el mundo conocido.

Eliza, por otra parte, consistió en un análisis de la conversación que se basa en la simulación de una terapia psicológica interactiva en la que el programa es el analista y el interlocutor es el analizado. Un ejemplo de una conversación sostenida con Eliza en la computadora de la Universidad de Stanford es el siguiente (el diálogo de la computadora en letras mayúsculas):

BUENAS TARDES, POR FAVOR ESCRIBA SU NOMBRE

buenas tardes, me llamo Javier

ES UN PLACER CONOCERTE, JAVIER

el placer es mio

¿QUE TE TRAE POR AQUI?

nada en especial, he tenido trabajo y me siento agotado

¿QUE TE HACE PENSAR QUE ESTAS AGOTADO?

he tenido problemas en mi trabajo

¿PROBLEMAS?

es decir, he estado preocupado con mi trabajo

CUENTAME TUS PREOCUPACIONES...

Según Colby, psicoanalista por entrenamiento, el programa resultó tan exitoso en términos clínicos que valdría la pena intentar reproducirlo masivamente como instrumento terapéutico.

Intento al que Weizenbaum, coautor del programa, se opuso decididamente con el argumento de que no puede decirse desde ningún punto de vista que "Eliza" es capaz de entender lo que se le dice. "Eliza", Weizenbaum insiste, es un programa capaz de responder, que no es lo mismo que comprender.

Como resultado de este y otros debates, las preguntas de investigación siguieron girando alrededor de la noción de comprensión.

Los seres humanos comprendemos en una plática mucho más de lo que se nos dice explícita y verbalmente (en parte porque la comunicación no~verbal juega un papel muy importante y en parte por nuestras capacidades para obtener información del contexto y para operar con altos niveles de incertidumbre) y pareció importante que estas habilidades fueran reproducidas en una computadora con pretensiones de inteligencia. Los investigadores se dedicaron entonces al estudio de los aspectos no-verbales del lenguaje y al de reconocimiento de patrones.

## 5.3 Ciencia (Teiresias, Dental, Crysalis, Macayma, Prospector)

Los sistemas de pronóstico deducen consecuencias posibles a partir de una situación. Su objetivo es determinar el curso del futuro en función de información sobre pasado y presente.

Esto abarca diversos problemas, tales como predicciones meteorológicas, predicciones demográficas, o incluso previsiones de la evolución bursátil entre otros. Quizá la aplicación más conocida sea PROSPECTOR, esto es un sistema para la evaluación de emplazamientos geológicos (con el que se encontró un yacimiento de mineral importante).

Los Sistemas Expertos ofrecen procedimientos informatizados para perfeccionar la toma de decisiones de la gerencia por medio de la combinación del conocimiento que poseen los expertos acerca del tipo de acciones que tiene que efectuar y la forma y el tiempo en que debe llevarlas a cabo con la permanencia, lógica, memoria y velocidad de cálculo del computador.

En los últimos años el término "Sistema Experto" ha sido reemplazado por el término "Sistema Basado en Conocimiento" (SBC), de esta forma, el énfasis original que se le daba a la transferencia de la experiencia humana a un computador ha sido sustituido por un término más moderado en el que se deja patente el uso de todos los recursos disponibles en la construcción del sistema. Sin embargo, sigue siendo una característica importante el que un SBC sea un sistema diferente a cualquier sistema de información avanzado.

La ingeniería de conocimiento puede ser vista como una parte de la ingeniería de sistemas de información, pero un sistema de información que le da un énfasis particular al carácter epistemológico (el saber sobre el conocimiento) de la información que es clasificada como conocimiento, esto es, "creencia verdadera imposible de desmentir con las leyes de la naturaleza" o "creencia verdadera adquirida a través de los sentidos".

Estas definiciones nos dan un contexto en el cual la credibilidad y la derivación de la información son significantes, y tienen que ser tenidas en cuenta. Esto es, los componentes basados en conocimiento de un sistema de información son aquellos que requieran meta-información (conocimiento sobre la información) y meta-proceso de información (conocimiento acerca de la manipulación de la información). Tales sistemas tendrán subsistemas que no sólo procesen información sino que también procesen meta-información acerca de la información y de su procesamiento.

Una de las características más importantes de los primeros Sistemas Expertos, tales como MYCIN [Shortliffe 1976] y de los entornos de desarrollo relacionados como TEIRESIAS [Davis 1982] era la capacidad que tenían de ser reflexivos y responder a las cuestiones de por qué y cómo, y usar esta capacidad para guiar al usuario final en la evolución del sistema y para entender mejor el dominio.

Desde la perspectiva del procesamiento de la meta-información un SBC puede ser caracterizado como un sistema de información esencialmente reflexivo [Maes 1988]. Esto sugiere que la diferencia entre sistemas de información convencionales y los SBC es que éstos últimos incluyen procesamiento de la meta-información.

**Construcción de un SBC. Metodología KADS.**

Inicialmente, los constructores de SBC no tenían una serie de pasos bien definidos para seguir a la hora de construir un SBC debido a la complejidad inherente al proceso de construcción. Como resultado, los SBC se desarrollaban de manera incremental, mejorando paso a paso la organización y la representación del conocimiento del sistema.

El resultado usual en un desarrollo incremental es que éste llega a un punto en el cual la base de conocimiento alcanza un tamaño inmanejable; el control no es efectivo y además es lento, y el sistema parece que está construido a partir de parches y construcciones no integradas. Ante este panorama, es necesario considerar el rediseño y la reimplantación del sistema. El ingeniero del conocimiento y el experto en el dominio deberán reexaminar el problema y rehacer su esquema de representación.

Esto dejaba patente que la construcción de SBC utilizando la metodología de construcción rápida de prototipos presentaba serios problemas.

Además, las bases de conocimientos que se construían se diseñaban para resolver un trabajo específico en una aplicación, y tenían poca o ninguna generalidad. De esta forma, el conocimiento que incorporaba un SBC no podía ser reutilizado para un propósito más general.

Otro problema existente era debido a la forma en que se representaba el conocimiento (principalmente con reglas de producción) que ocultaba importantes propiedades del proceso de razonamiento y de la estructura del conocimiento en el dominio de la aplicación. Esto dificultaba la adquisición y refinamiento del SBC, así como la reutilización del conocimiento, el poder reflexivo (para justificar sus resultados) del SBC y su relación con otros sistemas.

Ante tal panorama, sobre 1983 un grupo de investigadores comienzan a trabajar en una nueva metodología que se concentra en los métodos de adquisición del conocimiento, en el modelado del conocimiento y en el desarrollo estructural de los SBC. A esta metodología se le conoce con el nombre de KADS (Aunque actualmente el acrónimo KADS se utiliza como nombre propio, este se puede interpretar como "Knowledge Analysis and Documentation System" o como "Knowledge Analysis and La Simulación y la Inteligencia Artificial en el estudio de Sistemas Complejos Design Support") [Wielinga 1994].

Con esta metodología se pueden construir los SBC de una forma comprensiva.
Para ello sigue dos principios:
· el principio de múltiples modelos;
· y el principio de describir, de forma independiente de la implementación, las distintas capas de conocimiento que existen cuando se resuelve el problema.

**Principio Múltiples Modelos**
La construcción de un SBC es una tarea complicada. La idea que subyace detrás del primer principio de la metodología KADS, es la de facilitar la construcción de los SBC considerando el problema como formado por un conjunto de modelos, de forma que cada modelo enfatiza ciertos aspectos del sistema que se va a construir, y se abstrae de otros.

Estos modelos permiten la descomposición de las tareas del ingeniero del conocimiento: mientras construye un modelo puede olvidarse temporalmente de otros aspectos.

Esto es, el ingeniero del conocimiento sigue la estrategia de divide y vencerás. Este proceso de modelado no se centra únicamente en el conocimiento experto con el que trabajará el SBC, sino que tiene en cuenta las características de cómo ese conocimiento es añadido y usado en la organización.

Los modelos a construir no son considerados como pasos necesarios para desarrollar el SBC, sino como productos independientes que juegan un papel importante durante el ciclo de vida del SBC.

Aunque el KADS propone un conjunto de modelos para el desarrollo del SBC, sin embargo el número y grado de elaboración de los modelos dependerán del contexto del proyecto específico y de la planificación y control realizado en el proceso de construcción [Hoog 1994]. El KADS define los siguientes modelos:

· Modelo organizativo
· Modelo de tareas
· Modelo experto
· Modelo de agentes
· Modelo de comunicación
· Modelo de diseño

Modelo Organizativo
Un modelo organizativo genera un análisis del entorno socio-organizativo en el cual el SBC tendrá que funcionar. Esto incluye una descripción de las funciones, tareas y cuellos de botellas en la organización. Además, debe incluir los problemas que el SBC debe resolver y cómo influirá su uso en la organización y en las personas que en ella trabajan.

Modelo de Tareas
El modelo de tareas describe las actividades, de forma abstracta, necesarias para realizar las funciones en la organización. Establecer una relación directa entre función y tareas no es trabajo fácil, ya que dado el objetivo que el sistema debe lograr podrían haber varias formas de alcanzarlo. Qué alternativa es la adecuada para una aplicación determinada depende de las características de la propia aplicación, sobre la disponibilidad de conocimiento y datos, y sobre los requisitos impuestos por el usuario o por factores externos.

Modelo Experto
La construcción del modelo experto es la actividad central en la construcción de un SBC. Esto es lo que distingue el desarrollo de un SBC del desarrollo de sistemas convencionales. Su objetivo es especificar la experiencia de resolución de problemas requerida para realizar las tareas de resolución de problemas asignadas al sistema. La metodología KADS considera el modelo experto como el comportamiento que mostrará el sistema, y se centra en el tipo de conocimiento necesario para generar tal comportamiento, abstrayéndose de los detalles de cómo el razonamiento es realizado en
la implementación.

La Simulación y la Inteligencia Artificial en el estudio de Sistemas Complejos

Modelo de agentes
En el modelo de agentes se describen todas las propiedades relevantes de los agentes que realizan las tareas identificadas en el modelo de tareas. Un agente es un ejecutor de una tarea. Un agente puede ser un humano, o un programa de computadora, o cualquier "entidad" que ejecute una tarea. Los agentes más involucrados suelen ser los usuarios y el SBC. Un agente muy común es el sistema de bases de datos que almacena y proporciona información.

Modelo de Comunicación
Los SBC deben cooperar con otros agentes (usuarios, programas de computador, ...) para realizar su trabajo. Esta distribución de tareas define comunicaciones entre los agentes para poder lograr todos los objetivos. El modelo de comunicación es el encargado de describir los conceptos y mecanismos necesarios en cada una de las comunicaciones.

Modelo de Diseño
El modelo de diseño describe las técnicas computacionales que el SBC debería usar para tener el comportamiento especificado. Este modelo tiene en cuenta los requisitos del entorno que no son considerados en los otros modelos, tales como requisitos de velocidad, hardware y software. En tanto que muchos sistemas expertos se ocupan del razonamiento técnico más que del gerencial, la gerencia puede obtener computadoras de mucha potencia, con grandes memorias, rápidos y programados para tratar problemas clave de forma efectiva en un área empresarial determinada.

Hay una tendencia creciente a desarrollarlos y utilizarlos, sin embargo, los programas son caros y tienen que ser analizados con cuidado para determinar su contribución potencial al resultado final. Una de tales aplicaciones es el Palladian Operations Advisor (de Palladian Software, Inc., en Estados Unidos), diseñado específicamente para la dirección de la producción.

Las entradas a este programa comprenden las designaciones de procesos y máquinas de fabricación de una planta, las especificaciones de productos y el flujo de producción, a partir de lo cual puede representar gráficamente la planta industrial y el flujo de cada tipo de productos. Con estas representaciones pueden organizarse y reorganizarse las operaciones de fabricación.

El programa ayuda a la planificación y programación, asesorando en lo que se refiere a los programas que reducen el trabajo no deseable en niveles de proceso, ajustan el volumen de producción a la demanda de clientes y evalúan los cambios en las operaciones desde los puntos de vista económico y de producción.

Puede crear una influencia recíproca con los planificadores y directores de planta a medida que las condiciones cambian a diario o a cada hora, como consecuencia de averías mecánicas, modificaciones en los pedidos de los clientes o crisis en el exterior. El Palladian Operations Advisor puede analizar el estado de la combinación de productos para mantener la mayor eficacia y rentabilidad posible de las operaciones.

Además de DELTA, existen numerosos sistemas de este tipo; STEAMER, por ejemplo, se creó para enseñar a los oficiales de la armada los problemas de funcionamiento de una planta de propulsión a vapor, como las que impulsan a ciertos barcos. Este era el problema de formación más importante que existía, dada la complejidad de los sistemas.

El objetivo es dar al estudiante una concepción global de lo que pasa en la planta en cualquier momento, con la ventaja de que además el modelo de presentación es gráfico (utilizando Interlisp). Con un objetivo similar al de STEAMER, Construcciones Aeronáuticas S. A. (CASA) desarrolló el Proyecto Eolo CN-235.

En este caso, el problema está en el hecho de que pilotar un avión que cuesta cientos de millones de pesetas es un asunto muy serio a la vez que peligroso, lo que exige mucho tiempo de entrenamiento, tanto para pilotos como mecánicos, suponiendo para las compañías aéreas un gran problema, dado el elevado coste de los cursos y la escasez de instructores.

El proyecto surgió de la voluntad de Construcciones Aeronáuticas S. A. de ofrecer un curso específico para pilotos y técnicos de mantenimiento, a todos los compradores del avión CN- 235. Eolo CN-235 es un sistema de enseñanza interactivo que integra gráficos, texto y vídeo.

## 5.4 Medicina (Mycin, Casnet, Internist, Digitalist, Therap, Advisor, Expert)

**EJEMPLOS DE SISTEMAS EXPERTOS**

En esta parte de la web os presentaré de forma resumida dos Sistemas Expertos que fueron decisivos para el éxito del avance investigador en el campo de los Sistemas Expertos. Así os podréis hacer una idea, ya con un ejemplo real, de para que sirve y en que materias se aplica un Sistema Experto.

**1- MYCIN SISTEMA EXPERTO PARA DIAGNÓSTICOS MÉDICOS**

MYCIN es un Sistema Experto para la realización de diagnósticos, iniciado pro Ed Feigenbaum y posteriormente desarrollados por E.Shortliffe y sus colaboradores.

Su función es la de aconsejar a los médicos en la investigación y determinación de diagnósticos en el campo de las enfermedades infecciosas de la sangre. El sistema MYCIN, al ser consultado por el médico, solicita primero datos generales sobre el paciente: nombre, edad, síntomas, etc.

Una vez conocida esta información por parte del sistema, el Sistema Experto plantea unas hipótesis. Para poder verificarlas comprueba primero la exactitud de

las premisas de la regla (Os preguntareis como hace esto). Esto se realiza mediante una búsqueda de enunciados correspondientes en la base de conocimientos.

Estos enunciados pueden a su vez estar de nuevo en la parte de consulta de otra regla. También lo realiza mediante determinadas preguntas al usuario. Aquí se hacen preguntas del tipo: ¿Se ha practicado en el paciente algún tipo de intervención quirúrgica?

Con las respuestas que recibe, el MYCIN verifica o rechaza las hipótesis planteadas. Una serie de tests han demostrado que MYCIN trabaja igual de bien que un médico.( A mi particularmente me cuesta mucho creerlo y me parece que es increíble, pero ahí están los resultados de los test).

**2- XCON SISTEMA EXPERTO PARA CONFIGURACIÓN DE COMPUTADORAS.**

XCON es un Sistema Experto para configuraciones desarrollado por la Digital Equipment Corporation.

Según los deseos individuales del cliente se configuran redes de computadoras VAX. Ya que el abanico de productos que se ofrecen en el mercado es muy amplio, la configuración completa y correcta de un sistema de estas características es un problema de gran complejidad.

Las funciones de este Sistema Experto son las siguientes:

1- ¿Pueden conjugarse los componentes solicitados por el cliente de forma conveniente y razonable?

2- ¿Los componentes de sistema especificados son compatibles y completos?

Las respuestas a estas preguntas son muy detalladas. XCON es capaz de comprobar y completar los pedidos entrantes mucho más rápido y mejor que las personas encargadas hasta ahora de esa labor.

***Ejemplos de sistemas expertos utilizados:***

- DENDRAL: Es capaz de calcular o descubrir hechos relativos a las estructuras moleculares a partir de unos datos químicos sin elaborar.
- MYCIN: El más famoso de todos, que diagnostica infecciones en la sangre y meningitis y además sugiere el tratamiento que se debe seguir en cada caso.
- PUFF: El hermano menor de MYCIN, que diagnostica y trata enfermedades del pulmón.
- MOLGEN0: "MOLecular GENetics" ayuda a los biólogos que trabajan en el campo del DNA y la ingeniería genética.

- PROGRAMMER'S APPRENTICE: Como su propio nombre lo indica, se trata de un sistema que ayuda a la escritura de programas.
- 
- EURISKO: Un sistema experto capaz de aprender a medida que funciona, que crea circuitos micro eléctricos tridimensionales.
- 
- GENESIS: Permite a los cinetíficos planificar y simular experimentos en el campo de la unión de genes.
- EXPERT SYSTEMS TO COMBAT INTERNATIONAL TERRORISM: Para ayudar a expertos a entender la totalidad de la misión en que trabajan y asistirlos en la toma de desiciones.
- LEGAL DECISIONMAKING
- TATR: Tactical Air Targeting.
- TWIRL: Tactical Warfare. Simulaciones de guerras completas y guía de mejores acciones posibles a realizar, en casi todas las situaciones.

- R1: Programa utilizado para el descubrimiento de yacimientos petrolíferos bajo aguas marinas.

**DENDRAL**

*DENDRAL* es uno de los principales programas clásicos de aplicación. El objetivo de *DENDRAL* es trabajar como un químico informado, a partir de la formula y el espectrograma, a fin de deducir y producir una estructura química.

El programa *DENDRAL* produce estructuras a partir de formulas químicas y espectrogramas de masas mediante el método de generación y prueba. El generador consiste en un enumerador de estructuras y un sintetizador que produce un espectrograma de masas sintético mediante la simulación de la acción de un espectrómetro de masas real en cada estructura enumerada.

El enumerador de estructuras asegura que el generador total esta completo y no sea redundante porque tal enumerador utilice un procedimiento probablemente completo y no redundante de enumeración de estructuras.

El generador total también esta informado porque el enumerador de estructuras utiliza la formula química y el conocimiento de las sobrestructuras necesarias y las prohibidas.
El probador compara el espectrograma de masas real con los producidos por el generador.

Las estructuras posibles son aquellas cuyos espectrogramas sintéticos concuerden con el real de manera adecuada, la estructura que se toma como correcta es aquella cuyo espectrograma sintético se asemeja más al real.

*La idea clave en el análisis de medios y metas es reducir las diferencias*.

El objetivo de análisis de medios y metas es identificar un procedimiento que ocasiona una transición del estado actual al de meta, o al menos a un estado intermedio que este más cercano al de meta.

Por tanto, el procedimiento identificado reduce la diferencia observada sobre el estado actual y de meta.

Para efectuar el análisis de medios y metas:

1. Hasta que la meta no se alcance o no se tengan más procedimientos disponibles.

   - ✓ Describe el estado actual, el estado meta y la diferencia entre los dos.
   - ✓ Utiliza la diferencia entre el estado actual o la de meta, para seleccionar un procedimiento prometedor.
   - ✓ Utiliza el procedimiento prometedor y actualice el estado actual.

2. Si se alcanza la meta, notifique el logro de otro modo notifique el fracaso.

Las tablas de diferencia-procedimiento a menudo determinan medios.

Siempre que la descripción de la diferencia entre el estado actual y el estado meta sea la clave para saber que procedimiento intentar a continuación, una sencilla tabla de diferencia-procedimiento puede ser suficiente para conectar descripciones de diferencias con los procedimientos escogidos.

Debido a que los procedimientos que ocasionan transiciones se conocen a menudo como operadores, la tabla de diferencia-procedimiento se conoce como tabla de diferencia-operador en algunos círculos.

Algunas características más de estos programas.

- Capacidad de inferencia deductiva: Esto significa que los agentes no solo de recuperar información almacenada en la base de datos sino hacer deducciones usando la información para hacer deducciones que produzcan nuevas informaciones para la base de datos.
- Crecimiento dinámico: Porque los datos y las reglas estan sujetas a constante revisión. Es decir es fácil borrar o modificar los datos y las reglas que hacen las decisiones.
- Colección integrada de Conocimiento: Los agentes que constituyen el sistema pueden representar los juícios de muchos expertos en varias partes del globo. Estas experiencias es guardada en un solo lugar por lo que es posible su utilización luego por expertos o no expertos también. Sin embargo, la real utilidad de este sistema

es que permite un constante intercambio de datos y juicios de líderes expertos, y esto permite la formación de nuevas reglas e ideas acerca del tema.

Un Sistema Experto es una aplicación informática que simula el comportamiento de un experto humano en el sentido de que es capaz de decidir cuestiones complejas, si bien en un campo restringido.

Para esto, se debe tener en cuenta que la principal característica del experto humano viene a ser el conocimiento o habilidades profundas en ese campo concreto, por consiguiente, un Sistema Experto debe ser capaz de representar ese conocimiento profundo con el objetivo de utilizarlo para resolver problemas, justificar su comportamiento e incorporar nuevos conocimientos.

Se podría incluir también el hecho de poder comunicarse en lenguaje natural con las personas, si bien esta capacidad no es tan determinante como las anteriores de lo que se puede definir como Sistema Experto.

La potencia de un Sistema Experto se basa más en una gran cantidad de conocimientos que en un formalismo deductivo muy eficaz. La idea que se persigue cuando se construye un Sistema Experto es la de automatizar la labor del experto, partiendo en ocasiones de información insuficiente o incompleta.

Según la clase de problemas hacia los que estén orientados, podemos clasificar los Sistemas Expertos en diversos tipos entre los que cabe destacar diagnosis, pronóstico, planificación, reparación e instrucción; vamos a ver algunas de las aplicaciones existentes (o en periodo de desarrollo) para cada uno de los campos citados.

Los sistemas de diagnosis siguen un proceso de búsqueda de las razones del funcionamiento incorrecto de un sistema a partir de la información disponible. Aquí se podrían tener en cuenta tanto aplicaciones de diagnóstico médico como de averías.

En lo referente al diagnóstico médico, existe una serie de aplicaciones extensa en número (FLUIDEX, EACH, TROPICAID, SPHINX, ...), pero quizá la más conocida, a la vez que la más antigua, podría ser MYCIN.

MYCIN es el primer Sistema Experto que llegó a funcionar con la misma calidad que un experto humano, dando a su vez explicaciones a los usuarios sobre su razonamiento. Antes del desarrollo de MYCIN (mediados de los 70), se criticaba a la Inteligencia Artificial por resolver únicamente problemas "de juguete", sin embargo, el éxito de MYCIN demostró que la tecnología de los Sistemas Expertos estaba suficientemente madura como para salir de los laboratorios y entrar en el mundo comercial.

MYCIN es, en definitiva, un sistema de diagnóstico y prescripción en medicina, altamente especializado, diseñado para ayudar a los médicos a tratar con infecciones de meningitis (infección que produce inflamación de las membranas que envuelven al

cerebro y la médula espinal) y bacteriemia (infección que implica la presencia de bacterias en la sangre).

Dichas infecciones pueden ser fatales y a menudo aparecen durante la hospitalización. El problema se complica por la necesidad de actuar con rapidez.

Existen además en este campo Sistemas Expertos como TROPICAID, que permiten obtener información adicional sobre los medicamentos más usados. TROPICAID selecciona un conjunto de posibles diagnósticos a partir del análisis del cuadro médico, y propone un tratamiento óptimo para el caso concreto.

Por otra parte, el campo de la diagnosis abarca otras aplicaciones además de las médicas (si bien pueden ser estas últimas las más conocidas). En este caso se trata de fallos, averías o anomalías que se producen generalmente en una máquina.

Dentro de este apartado existen también numerosas aplicaciones, pudiendo encontrar incluso empresas en la propia CAPV que han desarrollado trabajos dentro del campo de la diagnosis de equipos industriales complejos. Tal es el caso de ADICORP (con su sistema TESP para la diagnosis de robots Puma) que trabaja además en proyectos de Visión Artificial.

## 5.5 Educación ( Cai, Scholar, Why, Sophia, West, Wumpus, Guidon, Buggy, Excheck)

Como se muestra en el inciso existen mucho lenguajes y sistemas al respecto, veamos una estructura de enseñanza de nivel mundial y formal que establece vínculos desde pequeños con este tipo de enseñanza y que opera en México:
:

**FUTUREKIDS**

Es un sistema de educación de tecnología de cómputo para niños que está presente en más de 80 países alrededor del mundo y cuyo éxito se basa fundamentalmente en dos cosas:

**PLAN DE ESTUDIOS ESTRUCTURADO.**

**FUTUREKIDS** cuenta con un currículo especializado para niños y jóvenes en informática, basado en objetivos de aprendizaje específicos acorde al nivel de asimilación y a la edad de cada niño. Este currículo ha sido elaborado por un grupo interdisciplinario de profesionales (ingenieros en cómputo, pedagogos, psicólogos, artistas, diseñadores gráficos, etc) en la base de la franquicia en Los Ángeles, California.

**Metodología de enseñanza.**

En **FUTUREKIDS** se combinan interesantes proyectos con los temas que fascinan a los niños a través de un ambiente interactivo. Cuenta con una psicopedagogía original que enseña a los niños a pensar creativamente, a expresarse abiertamente y a desarrollar el enorme potencial que ofrece la computadora.

El personal está comprometido a crear una comunidad mundial de aprendizaje, integrando la tecnología para facilitar y mejorar el rendimiento de los estudiantes. Promueve la expresión de la creatividad, el pensamiento sistemático y la cooperación. En los cursos de Futurekids, los niños aprenden haciendo, no memorizando.

El Centro de Futurekids ofrece un ambiente seguro y de apoyo constante para los niños. Para maximizar el aprendizaje de nuestros alumnos, en Futurekids ofrecemos una instrucción personalizada con un máximo de 8 alumnos por profesor.

Principio Educativo: Aprender mientras que se divierten

**Principios Educativos:**

- Ofrecer un ambiente seguro y de apoyo constante.
- Promover la expresión de la creatividad.
- Instrucción personalizada.
- Aprender haciendo, no memorizando

**FUTUREKIDS** es una organización mundial experimentada y especializada que nació en Los Ángeles, California en 1983,

- Con 17 años de experiencia.
- Con más de 2500 centros de aprendizaje, colegios y escuelas
- Enseña a más de un millón de niños y jóvenes a la semana
- En 78 países alrededor del mundo
- Currículum traducido a 13 idiomas

Se ayuda a las escuelas a convertirse en **"escuelas del futuro"** y a los niños a convertirse en **"niños del futuro"**, creando una comunidad mundial de aprendizaje que haya integrado el poder de la tecnología para facilitar y mejorar el rendimiento de los estudiantes.

## ¿POR QUÉ FUTUREKIDS?

Por que han sobresalido de la Era Industrial para entrar en la Era de la Información.

La computadora y el Internet son herramientas, medios a través del cual los estudiantes pueden explorar el mundo y expandir sus límites. Con las habilidades de cómputo, los estudiantes se vuelven más activos en el proceso de aprendizaje. Su aprovechamiento del Internet depende de nuestra creatividad y eficiencia.

Más del 90% de los trabajos actuales requieren del dominio de la computadora. Hoy, casi todas las universidades requieren que los alumnos de nuevo ingreso tengan habilidades de cómputo. Más del 78% de los nuevos puestos de trabajo que existirán a partir del año 2005, estarán relacionados con la tecnología informática.

Panamá necesita preparar a los niños y jóvenes para lo que se estarán enfrentando al llegar a hacer sus estudios superiores y salir al mercado laboral. Ante la globalización que se ha venido dando, nuestros hijos necesitan estar listos para competir no sólo a nivel nacional, sino con otros países de la región y el mundo.

El objetivo de **FUTUREKIDS** es que los niños y jóvenes logren el dominio de la tecnología de cómputo. Al lograr este dominio, tendrán las bases que les permitirá comprender cualquier aspecto tecnológico que se le presenten en el ámbito escolar y posteriormente en el ámbito laboral. Más allá de tener las habilidades para el uso de las computadoras, aqui les enseña cómo aplicar dichas habilidades a proyectos específicos y en sus vidas diarias.

Se ofrece en nuestros Centros de Aprendizaje a niños a partir de los 3 años y en colegios y escuelas desde Pre-Kinder hasta secundaria.

FUTUREKIDS fue fundada en 1985 en Los Angeles, California por Peter Markovitz, David Ullendorf y un grupo de educadores del Programa Especial para el Avance Educativo de la Universidad de California, con la idea de

ofrecer una solución a la preparación de los niños del futuro. Lo innovativo de su concepto ha significado un éxito, ya que en la actualidad cuenta con más de 2500 centros de aprendizaje en más de 78 países.

El currículum Futurekids se enseña asimismo en más de 2000 instituciones educativas tanto públicas como privadas. Todas las semanas, más de un millón de niños y jóvenes de todo el mundo participan en la aventura de aprendizaje del mundo Futurekids.

FUTUREKIDS pone al alcance de los nicaragüenses las mismas herramientas que se usan en los centros y escuelas de los países más avanzados del mundo. Estas preparan al conglomerado mundial de niños Para hacer frente a la continua evolución del cada vez más exigente mercado laboral.

**Misión**

"Ayudamos a las escuelas a convertirse en 'escuelas del futuro' y a los niños a convertirse en 'niños del futuro', creando una comunidad mundial de aprendizaje que haya integrado el poder de la tecnología para facilitar y mejorar el rendimiento de los estudiantes".

**Visión**

"Seremos líderes en facilitar soluciones de educación continua ofreciendo sistemas de tecnología integrada de la más alta calidad accesible a amplios sectores de la población. Aprovecharemos la infraestructura educativa, empresarial y comunitaria existente en el país fomentando la conectividad, solidaridad y competitividad que requiere la sociedad futura".

"Lo novedoso de futurekids es que, lejos de ser un curso de computación, es todo un sistema de enseñanza que integra la tecnología en el aprendizaje desde los primeros años de primaria. Apunta hacia el dominio del cómputo dentro de su plan de desarrollo académico".


**ALGUNOS SISTEMAS TUTORIALES INTELIGENTES.**

**SCHOLAR (1970)**

SCHOLAR es considerado el primer STI. Fue diseñado por Jaime Carbonell para enseñar la geografía de América del Sur, abarcando conceptos como país, estado, población, etc.

La representación del conocimiento se hace a través de una red semántica, donde los nodos representan conceptos y objetos geográficos, y los arcos representan una jerarquía parcial de relaciones tales como superparte, súper concepto y superatributo. Se pueden realizar inferencias aprovechando las propiedades de herencia de la red semántica.

Inicialmente, SCHOLAR crea un modelo con todas las preguntas (con sus respuestas) del tópico a enseñar. El estudiante contesta una serie de preguntas y el tutor compara las respuestas con su modelo, generando la retroalimentación apropiada (aprobar o corregir la respuesta).

El sistema cuenta con una interfase de lenguaje natural con vocabulario limitado. El texto se genera a partir de plantillas de oraciones y preguntas. El control de la sesión está repartido, pues tanto el estudiante como el sistema pueden tomar la iniciativa haciendo preguntas [Wenger, 1987]. La Figura 1 muestra un ejemplo de diálogo con SCHOLAR.

**Sophie (1975)**
El *Sophisticated Instructional Environment*, diseñado por John Seely, fue el primer sistema de EAC inteligente en pretender entender su dominio. Se diseñó para enseñar a diagnosticar fallas en sistemas eléctricos, pero su arquitectura podría aplicarse en muchos otros campos. Sophie permite al estudiante hacer pruebas con un circuito descompuesto, al tiempo que evalúa sus hipótesis a partir de la información disponible y lo aconseja.

En lugar de tratar de imitar a un maestro, Sophie crea un ambiente en el cual el estudiante aprende probando sus ideas. Para lograr esto, tiene un poderoso modelo del dominio del conocimiento, junto con numerosas heurísticas para contestar a las preguntas del estudiante, generar contraejemplos y para criticar constructivamente sus decisiones. La Figura 2 muestra un ejemplo de esta interacción.

Tiene un esquema de inferencia que utiliza múltiples representaciones del conocimiento [Seely, 1975; Woolf, 1984; Molnar, 1997]:
1. Modelo de simulación del microcosmos (circuitos eléctricos).
2. “Especialistas procedurales” con reglas lógicas y estrategias heurísticas para aprovechar el modelo de simulación.
3. Redes semánticas para representar el conocimiento de hechos constantes.

4. Gramática semántica, que reemplaza las típicas categorías sintácticas (sujeto, verbo) con categorías semánticamente significantes (resistencia, transistor); para cada concepto hay una regla gramática con formas alternativas de expresarlo y comprenderlo.

La última versión, Sophie–III, cambia al experto basado en simulador por uno basado en un modelo de causa– efecto; esto permite una mayor profundidad en sus explicaciones [Polson, 1998].

**Why (1977)**
Stevens y Sollin desarrollaron este sistema tutorial para la enseñanza del proceso pluvial. El paradigma educativo en que Why se basa es el método socrático, en el que la instrucción consiste en cuestionar las respuestas del estudiante para hacerlo examinar su validez, encontrar contradicciones y hacer inferencias correctas. Para lograrlo se utilizaba una limitada interfase de lenguaje natural. La Figura 3 muestra un ejemplo de diálogo con Why.

La representación del dominio del conocimiento se basaba en *scripts* jerárquicos, series de ranuras que describen las secuencias de eventos que pueden suceder en una situación conocida, estableciendo las relaciones temporales y causales entre los distintos eventos. Los eventos representan a su vez condiciones de entrada, resultados, propiedades y actividades.

Además, cuenta con una base de reglas de producción, donde la condición es la última respuesta del alumno (el sistema no tiene memoria), y el consecuente, la acción que Why propondrá a continuación. [Beutelspacher, 1995].

**WUSOR (1977)**

Wusor, fue creado por Goldstein y Carr para asesorar al estudiante en el juego lógico de computadora Wumpus.

Este juego consiste en explorar una caverna, moviéndose de una zona a otra, evitando al temible Wumpus y otros obstáculos, y buscando el tesoro. Al entrar en una zona se pueden escuchar los ruidos (gruñidos, etc.) provenientes de las zonas vecinas; al seleccionar la siguiente cueva a visitar se necesita utilizar razonamiento lógico y probabilístico para tomar la mejor decisión.

El sistema tutorial es el primero en utilizar módulos para el experto, el psicólogo y el modelo del estudiante.

La representación del dominio de cada módulo se compone de tres tipos de reglas. Utiliza sobreposición (*overlay*) de las reglas en los tres módulos [Beutelspacher, 1995; Wenger, 1987]:

1. Reglas de habilidades: Contienen el conocimiento necesario para jugar Wumpus de forma óptima (experto).
2. Reglas de evidencia: Para modelar los criterios usados al formular distintas hipótesis sobre las habilidades del estudiante. Utiliza algoritmos de reconocimiento de planes (modelo del estudiante).
3. Reglas de explicación: Para las estrategias tutoriales.

**GUIDON (1979)**

El sistema tutorial de Clancey está diseñado para ayudar a aprender la relevancia de los datos clínicos y de las pruebas de laboratorio en el diagnóstico de infecciones bacteriológicas y en la recomendación del tratamiento adecuado. La Figura 4 ejemplifica este proceso.

La base de conocimiento de su módulo experto está basada en Mycin, uno de los primeros sistemas expertos realmente exitosos. Se utiliza para resolver el caso y generar un árbol AND / OR que capture toda la información que le sea relevante. El modelo del alumno es simplemente un subconjunto de este conocimiento (modelo de sobreposición); en este sentido, su conducta sólo se puede entender en función de dicho conocimiento.

Originalmente, el módulo experto se trató de implementar directamente a partir de Mycin. Sin embargo, se descubrió que hacía falta más información específicamente importante para el aprendizaje. Por ejemplo, relaciones lógicas abstractas entre las reglas de producción permiten al sistema recomendar un enfoque particular para atacar el problema.

Guidon-2 utiliza a Neomycin, descendiente de Mycin que separa las estrategias de diagnóstico de los hechos médicos [Woolf, 1984; Polson, 1988; Beutelspacher, 1995].

El objetivo es hacer una revisión de las diferentes técnicas utilizadas para aplicar la inteligencia artificial en los sistemas de enseñanza inteligentes. Se analizarán las distintas técnicas en torno a los cuatro componentes de una arquitectura tradicional: módulo del dominio, modelo de estudiante, módulo tutorial e interfaz. Se explicarán las distintas formas de control incluyendo un control para sistemas de enseñanza reactivo. Lo anterior se acompañará con la descripción de distintos sistemas de enseñanza inteligentes.

En este escrito se hace una revisión de algunos de los Sistemas de Enseñanza Inteligentes existentes (SEIs), describiendo su comportamiento con base en las técnicas de diseño utilizadas en sus cuatro componentes.

Empezaremos por el tipo de conocimiento que puede ser transmitido por un sistema de enseñanza inteligente (SEI): Procedimental (relacionado con habilidades), declarativo (relacionado con hechos o conceptos) y cualitativo (relacionado con la habilidad mental de simular y razonar con base en procesos dinámicos).

El objetivo de un SEI es tener la capacidad de adaptación durante el proceso tutorial. Este objetivo se logra inter-relacionando los cuatro componentes básicos del sistema: 1) módulo del dominio o módulo experto, 2) modelo del estudiante, 3) módulo tutorial y 4) interfaz.

Este tipo de sistemas necesitan utilizar técnicas de inteligencia artificial para: a) representar el conocimiento que el sistema tiene del dominio y b) controlar el proceso tutorial y los métodos bajo los cuales son aplicados estos principios.

A continuación se describen las características y las diferentes técnicas de diseño que los componentes de un SEI pueden utilizar para lograr el objetivo antes mencionado.

.

El módulo del dominio o módulo experto

En este componente vamos a encontrar el conocimiento específico y detallado, obtenido de los expertos humanos que llevan años dedicándose a la tarea cognitiva (TC) que se pretende enseñar. Las investigaciones en este campo se centran en cómo codificar el conocimiento y cómo representar la experticia. Existen diferentes técnicas para abordar el problema: a) caja negra (Black Box), b) caja de cristal (Glass Box) y c) modelos cognitivos. Las tres tienen ventajas y desventajas cuando se utilizan en la construcción de un SEI.

Experto caja negra

En este tipo de representación, el conocimiento se encuentra compilado, esto es, la respuesta se dirige basándose en entradas y salidas, lo que proporciona una medida de lo correcto del resultado, pero no tenemos acceso al detalle de por qué se tomó una determinada decisión.

El ejemplo clásico de este tipo de tutores es SOPHIE Burton y Bell (1975), que utilizaba un simulador electrónico de propósito general. Se trataba de enseñar a los estudiantes cómo encontrar fallas en un circuito electrónico. El tutor utilizaba su simulador para determinar si las medidas que el estudiante realizaba sobre varios puntos del circuito eran correctas.

El simulador trabajaba con base en un conjunto de ecuaciones, no como lo haría un humano, por lo que no era posible para SOPHIE explicar las decisiones en detalle.

Este tipo de representación puede utilizarse con entornos de aprendizaje reactivos, que indique al estudiante si la operación realizada es correcta o incorrecta y, posiblemente, cuál es el siguiente movimiento apropiado. Esta combinación de tutor reactivo y caja negra es una filosofía interesante a la hora de crear un SEI basado en sistemas expertos (no limitados a cajas negras), ya que permite ahorrar tiempo en la fase de implementación.

En el caso de SOPHIE el tutor funcionó bien con su entorno reactivo, aunque se hicieron versiones posteriores de SOPHIE: Brown, Burton y deKleer (1982), en los que se incorporó un modelo causal de circuitos orientado a permitir la explicación de decisiones en detalle.

Sin embargo, un SEI, no debe limitarse a decir correcto/incorrecto, sino que tiene que saber qué puntos son críticos, o cuándo se debe explicar con más detalle. Existen otras técnicas, como el experto caja de cristal, que contemplan este otro aspecto de los sistemas de adiestramiento.

Experto caja de cristal

La metodología básica para construir este tipo de experto involucra a un ingeniero del conocimiento y a un experto en el dominio. Este último identifica el área y el alcance del problema. En esta etapa se enumeran y formalizan los conceptos claves del dominio, formulando un sistema para implementar el conocimiento y finalmente de manera iterativa probarlo y refinarlo.

Estos sistemas se caracterizan por contener de forma articulada una gran cantidad de conocimiento experto. Su construcción se divide básicamente en dos etapas: la primera se refiere a la adquisición del conocimiento (tiempo consumido para construir el componente experto), y la segunda trata de automatizar este conocimiento.

Debido a que el sistema experto resultante contiene conocimiento menos compilado, la enseñanza resulta más factible que con un experto de caja negra. En este tipo de sistema experto la experticia del conocimiento humano queda representada de forma articulada. Lo anterior conlleva la ventaja de poder llegar a niveles de detalle más profundos durante la enseñanza del conocimiento, motivo del sistema.

Una de las principales conclusiones de GUIDON fue que no sólo se debe atender a la representación del conocimiento en el módulo experto, sino también a la forma en que se presenta ese conocimiento. Para que un tutor actúe de forma apropiada el módulo experto debe desplegar ese conocimiento en la forma en que lo hacen los humanos, de acuerdo con sus restricciones.

Los trabajos de Clancy, de gran importancia, mostraron que los SEI están limitados si sólo usan sistemas expertos para su proceso tutorial. Esta última reflexión es la que ha conducido a la utilización de modelos cognitivos.

Los modelos cognitivos

El objetivo de un modelo cognitivo es desarrollar una efectiva simulación de la solución del problema en un determinado dominio desde el punto de vista del humano. En esta técnica el conocimiento se divide en componentes que guardan una relación directa con la forma en que el humano los clasifica y los utiliza.

El mérito de esta aproximación es que proporciona un módulo experto cuya taxonomía permite un proceso tutorial y una comunicación más profunda con el estudiante. Aunque ha habido avances importantes en las ciencias cognitivas en los últimos diez años, estos modelos requieren gran cantidad de tiempo en su desarrollo, por el gran número de detalles que tienen que ser incorporados.

En esta técnica se tienen que considerar básicamente tres cuestiones: qué componentes procedentes del análisis cognitivo son importantes para el proceso tutorial, con qué nivel deben ser representados los componentes, y, finalmente, cómo deben ser tratados los diferentes tipos de conocimiento: procedimental, declarativo y cualitativo en esta técnica de modelado.

Makatsiná Laureano y de Arriaga (1999) y Laureano y de Arriaga (2000), es un SEI donde se utiliza un modelo cognitivo cuyo análisis y diseño se desarrolló de acuerdo a las investigaciones realizadas por Castañeda (1993), Redding (1992) y Ryder & Redding (1993). El resultado de este análisis es conocer el número de habilidades que se integran en una de mayor complejidad. El sistema Makatsiná centra su enseñanza en el aprendizaje de la habilidad para resolver estructuras triangulares por el método de los nodos, de acuerdo a la filosofía reactiva Beer ( 1990).

Existen otros SEIs en que se han utilizado modelos cognitivos con el fin de esclarecer el proceso de la enseñanza Fletcher y Harris (1996), Gott (1989).

Diferentes tipos de conocimiento

Otro de los aspectos importantes de este módulo experto es el referente al tipo de conocimiento que manejará, ya que ello conlleva la forma en la que será representado.

El conocimiento procedimental

Se refiere básicamente al conocimiento que subyace al desarrollo de una tarea y está directamente relacionado con las reglas de producción consideradas por los investigadores como una representación que capta lo esencial del proceso humano con su mecanismo cíclico de válida-actúa (si - entonces).

Tal es el caso de los sistemas LMS, que enseñan procedimientos algebraicos Sleeman y Brown (1982), BIP; enseña programación en BASIC Barr, Beard y Atkinson (1976), MENO; enseña programación en Pascal Woolf y McDonald (1984), WEST; enseña aritmética a nivel elemental Burton y Brown (1982), CAPRA; enseña programación Fernández (1989) y Makatsiná; enseña la habilidad de la resolución de estructuras triangulares por el método de los nodos Laureano, Arriaga y Martínez (1999).

El conocimiento declarativo

Este tipo de conocimiento se refiere básicamente a hechos que guardan relación con un uso especializado para un caso particular. Es como decir América fue descubierta por Cristóbal Colón, pues esta frase representa un hecho concreto y aislado.

Tal es el caso de los sistemas: SCHOLAR, de enseñanza de geografía de Sudamérica Carbonell (1970), WHY, de enseñanza de meteorología (básicamente posibilidades de lluvia y los factores que en ellas influyen), LISP Tutor, que enseñaba el conocimiento declarativo del lenguaje LISP Anderson y Reiser (1985) y GUIDON.

El conocimiento cualitativo

Es el conocimiento que subyace en la habilidad de los humanos para simular y razonar con respecto a procesos dinámicos de manera mental. Un ejemplo de este tipo de conocimiento lo encontramos en SOPHIE Brown, Burton y Bell (1975).

Representación del conocimiento

Dentro de los formalismos que existen para representar el conocimiento procedimental, contamos con: reglas de producción y expertos compilados. El tipo de representación elegida está íntimamente ligada a la técnica que será utilizada para detectar los errores y para la intervención tutorial. Por eso es tan importante.
Reglas de producción

Dentro de los sistemas de producción que utilizan reglas de producción Laureano (1995) existen muchas variaciones, pero todos involucran un conjunto de reglas, que emparejan con una memoria de trabajo de hechos, para ser disparadas. Dentro de las ventajas para el proceso tutorial se encuentra el hecho de que el módulo tutor

pueda realizar sus decisiones con base en correr una simulación del aprendizaje deseado.

Otra es la modularidad de poderlas aplicar por bloques, algo inherente al proceso de enseñanza, y, finalmente, el estado de conocimiento del estudiante puede ser diagnosticado como un conjunto de reglas de producción. Ejemplo de esto último son LISP Tutor Anderson y Reiser (1985) y LMS Sleeman y Brown (1982).

El experto compilado

Se utiliza cuando por limitaciones en los requerimientos del equipo o por no resultar conveniente, no se utiliza el modo de representación de reglas de producción. En este caso se compilan las soluciones de los problemas o la mayor parte de los cálculos de la solución. Este tipo de representación se conoce en la literatura como 'compiling the expert out' Anderson (1988) y obliga a un diseño que tenga en cuenta estructuras de archivo que permitan un fácil y rápido acceso a disco. Existen implementaciones de estas bases de datos en hardware. Con esta técnica se pierde la capacidad de resolver cualquier problema propuesto por el estudiante. Sus características permiten implementar el proceso tutorial desde otra perspectiva.

Como ejemplo de SEI que utiliza al experto compilado, está Makatsiná Laureano, Arriaga y Martínez (1999), en el que se combina este último con reglas, básicamente por el tipo de proceso tutorial que se realiza: entrenador-deportivo, aplicado a una tarea cognitiva que conlleva integrar habilidades. Uno de los factores que influyó en la elección del experto compilado, es la restricción de que un sistema reactivo debe conocer el siguiente paso de antemano.

El modelo de estudiante

Este módulo contiene todos los datos en un instante dado del estudiante y sirve para diagnosticar los efectos del proceso tutorial. Esta información se utiliza para elegir el siguiente tema de enseñanza y qué tipo de táctica será la adecuada para el adiestramiento. En caso de error será considerada una táctica remedial.

El modelo del estudiante consta de dos componentes: a) la base de datos que representa el comportamiento del estudiante durante el proceso tutorial y b) el proceso de diagnóstico que manipula la base de datos.

Dimensiones del espacio, a ser consideradas en el modelo del estudiante

De acuerdo a VanLehn (1988) existen tres espacios a considerar en la modelización del estudiante: 1) el ancho de banda, 2) tipo de conocimiento a ser enseñado y 3) diferencias entre el estudiante y el experto.

El ancho de banda

Se refiere a la entrada de diferentes tipos de información, que permiten saber qué es lo que el estudiante está haciendo o diciendo. La cantidad y la calidad de esta información es importante, ya que ésta se comunica al componente de diagnóstico. Este componente utiliza esta información, en sus inferencias y creencias, con respecto al estado del usuario.

VanLehn propone los siguientes niveles de información, autocontenidos, en orden decreciente, de acuerdo a la información que pueden proporcionar:

· Nivel. Estados Mentales.
· Nivel. Estados Intermedios.
· Nivel. Estados Finales.

En el nivel más alto se encuentran los estados mentales por donde atraviesa el estudiante cuando resuelve un problema. Estos a su vez tendrán estados intermedios, los cuales a su vez contendrán a los estados finales.

Un estado mental podría ser aquel que reflejase los estados mentales por donde el usuario puede atravesar durante la resolución de un problema. En el caso de los estados intermedios, éstos están formados por los cambios de estados que guían la solución de un problema, desde su estado inicial hasta su estado final. Estos estados son los que interesan al proceso tutorial. Los estados finales son, como su nombre lo indica, el estado final de la solución de un problema.

Entre más información tenga el módulo de diagnóstico, obviamente éste será mejor. Pero no siempre se puede tener acceso a los estados intermedios. Por otro lado, no existe forma de accesar los modelos mentales desde un SEI (sería tanto como poder ver lo que el estudiante tiene en cada momento del desarrollo del problema en su mente). A través de preguntas se puede llegar a una aproximación.
La información obtenida del ancho de banda es importante en el sentido que de ella depende el algoritmo a usar en el diagnóstico.

El Tipo de conocimiento

Se refiere a las diferentes técnicas de tratamiento existentes, de acuerdo a la clasificación del conocimiento: procedimental, declarativo o cualitativo.

En el caso del conocimiento procedimental, que está incluido en la resolución de un problema, hace falta una clase de interpretación que relacione al modelo del estudiante con el conocimiento de resolución del problema, debido a que el intérprete tiene que tomar decisiones basadas en conocimiento local. Existen dos tipos de conocimiento procedimental: a) ordenado o jerárquico y b) no ordenado.

Cuando hablamos de conocimiento ordenado, sabemos que lleva implícito subobjetivos. Esta característica va íntimamente ligada con el tipo de diagnóstico utilizado, esto es, se necesita conocer las condiciones que disparan otro estado del problema, así como el conjunto total de estados y los subobjetivos.

Las diferencias entre el estudiante y el experto

Están representadas de manera general por el modelo del experto, más una lista de los elementos que el estudiante no tiene (missing conception) y el experto sí, así como los que el estudiante tiene y el experto no (misconception). Existen diferentes técnicas de implementación: a) overlay, b) archivos de errores y c) librería de errores hechas en partes.

En la forma overlay, el conocimiento del modelo del estudiante queda representado como un subconjunto del conocimiento del experto, lo que es equivalente al modelo del experto, más una lista de los elementos que el estudiante no tiene.

En los archivos de errores se resaltan las diferencias, teniendo en cuenta los misconceptions y los missing conceptions. Aquí el modelo del estudiante está representado por el modelo del experto, más una lista de errores. El sistema diagnostica al estudiante, encontrando los errores en esta lista.

Dado que los errores son muy importantes para el diagnóstico, existen diferentes técnicas para encontrarlos: a) los errores se obtienen a partir de los trabajos en educación sobre el tema, b) los errores se encuentran observando directamente a los estudiantes, realizando la Tarea Cognitiva (TC) y c) si existe una teoría de aprendizaje sobre el dominio, se utiliza para predecir los errores en el desarrollo de la TC.

Tipo de conocimiento Procedimiento no-ordenado
Procedimiento ordenado Declarativo
Ancho de Banda
Estados Mentales Seguimiento de una Ruta (1)
Estados Intermedios Seguimiento de Asuntos (5)
Reconocimiento de un plan (4)
Sistema Experto (6)
Estados finales Encontrando la Ruta (2)
Árbol de Decisiones (7)
Generación y Prueba (8)
Condición por Inducción (3)
Generación y prueba (8)
Diagnóstico Interactivo (9)

Técnicas de diagnóstico

En la siguiente sección abordaremos el segundo componente de un modelo de estudiante que es el procedimiento de diagnóstico, empleado sobre la base de datos. Según la clasificación de VanLehn (1988), existen nueve técnicas de diagnóstico, que se numeran en la Tabla 1, donde además se hace una recomendación de su uso de acuerdo a dos de las dimensiones de modelización del estudiante (ancho de banda y tipo de conocimiento).

1. Seguimiento de una Ruta (Model-Tracing).
2. Encontrando la Ruta (Path Finding).
3. Condición por Inducción (Condition Induction).
4. Reconocimiento de un Plan (Plan Recognition).
5. Seguimiento de asuntos (Issue-Tracing).
6. Sistemas expertos (Expert systems).
7. Árboles de decisión (Decision Trees).
8. Generación y Prueba (Generate and Test).
9. Diagnóstico Interactivo (Interactive Diagnosis).

Se abordarán la 1, la 4 y la 5 para explicar cómo funcionan en un SEI. En este caso Makatsiná. Para profundizar en el tema, consultar la bibliografía relacionada.

En la técnica de seguimiento de una ruta propuesta por Anderson, J. y comentada en VanLehn (1988), se supone que todos los estados mentales significativos del estudiante están disponibles para el procedimiento de diagnóstico. La idea básica radica en el uso de un intérprete no determinista para modelar la solución del problema. Así, en cada paso del proceso de resolución, el intérprete puede sugerir un conjunto de reglas aplicables. El procedimiento de diagnóstico disparará estas reglas y obtendrá de esta forma el conjunto de posibles estados sucesores (en el caso de un intérprete determinista sólo contaremos con un estado sucesor). Uno de estos estados será el que corresponda a la solución del estudiante. Como todas las técnicas, ésta tiene sus ventajas y desventajas. Entre las desventajas está qué hacer si ninguno de los estados disparados coincide con el del estudiante, ya que esto no tiene sentido en el caso del determinista.

En la técnica de reconocimiento de un plan se necesita cumplir dos condiciones: 1) que el conocimiento sea procedural y ordenado y 2) que todos o casi todos los estados físicos observables del estudiante puedan ser utilizados por el procedimiento de diagnóstico. En esta técnica el dominio se representa en un árbol, en el que los nodos terminales son las acciones más primitivas. Los nodos no-terminales incorporan los subobjetivos y la raíz el objetivo final.

La técnica seguimiento de asuntos Burton y Brown (1982) esta basada en el análisis de episodios cortos en la solución de un problema, dividiendo su

observación en conjuntos de micro-habilidades o asuntos que se emplean durante ese episodio. Este tipo de análisis no explica cómo estos asuntos interactúan, ni el papel que desempeñan en la solución global del problema. Sólo interesa saber si se usan o no.

En Makatsiná el objetivo era lograr un tutor reactivo. Se implementó el seguimiento de una ruta, dado que sólo tenemos que enseñar una estrategia para la solución de estructuras triangulares, debido a la restricción de que en los sistemas reactivos es de primordial importancia saber cómo actuar. Se combinó con el seguimiento de asuntos, dadas las características del conocimiento procedimental-ordenado. Finalmente, debido a la división de la solución del problema en etapas más finas, representadas por subtutores que revisarán el uso de varias habilidades, fue implementado el reconocimiento de un plan dentro de esos micromundos.

Módulo tutorial

A este módulo le concierne todo lo referente a los problemas en el desarrollo del curriculum y de la forma de enseñar ese curriculum. El curriculum se refiere a la selección y a la secuencia del material de enseñanza. La enseñanza también conocida como proceso tutorial se refiere a los métodos para presentar ese material, de acuerdo a Halff (1988). De aquí que los SEI, puedan utilizar diferentes técnicas de enseñanza.

En general las intervenciones tutoriales deben contener, si no todas, al menos alguna combinación de las siguientes características: a) tener algún control sobre el curriculum y su secuencia, b) ser capaz de responder a preguntas hechas por el usuario, y c) darse cuenta de cuándo el usuario necesita ayuda y de qué tipo.

Aspectos centrales

Los aspectos centrales que se tratan en el diseño del módulo tutorial, son: el control sobre la selección y la secuenciación, el estilo de aprendizaje, el estilo de la enseñanza y el tipo del dominio de la TC. Estos tres aspectos íntimamente ligados son fundamentales en el diseño de este módulo.

Formas de control de la selección y secuenciación en los SEIs

De acuerdo a Fernández (1989) existen 3 tipos de control donde se pueden agrupar los diferentes SEI:

1. Sistemas basados en reglas.
2. Sistemas basados en autómatas de estados pedagógicos.
3. Sistemas basados en planificación.

En el trabajo realizado por Fernández (1989) se explica a fondo en qué consisten cada uno de ellos. A continuación se presenta un resumen de las ideas básicas de los tres tipos de control existentes.

A los sistemas con control basado en reglas pertenecen aquellos que han sido construidos con base en un sistema experto. A este tipo corresponden los sistemas: GUIDON, NEOMYCIN y HERACLES. Comentaremos como funciona GUIDON.

GUIDON Wenger (1987) fue diseñado para satisfacer los siguientes objetivos: 1) demostrar la utilidad pedagógica de la base de datos del sistema experto MICYN, 2) descubrir el conocimiento adicional en un sistema de enseñanza inteligente y 3) expresar las estrategias para el proceso en términos independientes del dominio.

En GUIDON Clancy (1982) se separan las capacidades de diálogo tutorial de las de resolución del problema. El adiestramiento se logra en base a un programa de consulta basado en reglas y las capacidades de diálogo.

GUIDON compara las preguntas del estudiante con las del experto y las critica en base al siguiente patrón. Cuando el usuario plantea una hipótesis, la compara con la conclusión a la que llegó el experto (MYCIN) a partir de los mismos datos. La transferencia de conocimiento se realiza a través de diálogos de diferentes casos. El propósito del sistema es ampliar el conocimiento del estudiante, señalando líneas inapropiadas de razonamiento y haciendo sugerencias sobre aspectos que el estudiante no consideró.

Existen básicamente dos paquetes de reglas: las que componen a la base del conocimiento del sistema experto y las reglas encargadas del proceso tutorial, las cuales a su vez se dividen en reglas para acumular certeza (premisa) y reglas para seleccionar un procedimiento de discurso (acción).

Como sistema basado en autómatas de estados pedagógicos, comentaremos a MENO-Tutor Woolf y McDonald (1984). Este sistema fue desarrollado con el fin de proporcionar un marco de trabajo adecuado para definir y probar reglas pertenecientes al proceso tutorial, con el fin de generalizar estas reglas. El sistema utiliza dos componentes diferentes para la planificación y generación del proceso tutorial: el componente encargado del proceso tutorial y el generador de lenguaje natural.

El componente encargado del proceso tutorial está formado por un conjunto de unidades de decisión organizadas en tres niveles de planificación que refinan sucesivamente las acciones del tutor. En el nivel superior (Nivel Pedagógico) lo que se decide es la frecuencia de interrupción al estudiante y la frecuencia con que se verificará su conocimiento.

En el segundo nivel (Nivel Estratégico) se refina la decisión pedagógica en estrategia, especificando el método que se va a usar, como podría ser el determinar la competencia del estudiante con preguntas. En el tercer nivel (táctico) se selecciona una táctica para implementar la estrategia.

La implementación de este componente es similar a una red de transición que se atraviesa de estado a estado, mediante una rutina iterativa a través de un espacio de caminos. Los caminos entre estados no son fijos, ya que la estructura de control provee un conjunto de metareglas que producen cambios en los caminos por defectos.

Los sistemas basados en planificación nacen como una necesidad para construir sistemas de enseñanza, que utilicen estrategias para cursos largos. La planificación puede ser estática o dinámica.

Peachey y McCalla (1986) proponen usar técnicas de planificación para crear cursos largos e individualizados que traten campos más amplios. Este sistema pertenece al tipo de planificación estática.

Las técnicas de planificación se han utilizado principalmente en robótica García-Alegre, Ribeiro, Gasos y Salido (1993) y Torra (1993). También en áreas como la modelización del razonamiento Hayes-Roth (1979) y comprensión del lenguaje natural Wilensky (1983) entre otras.

El proceso de planificación consiste en decidir el curso de una acción antes de que se realice, y un plan es la representación del curso de dicha acción. Un planificador es un programa que parte de un estado inicial e intenta alcanzar el estado final deseado, mediante la aplicación de un conjunto de operadores sobre los objetos que componen el mundo en el que opera. Esto es, construye la secuencia de operadores que permite transformar el estado inicial en el estado final.

Un planificador incorpora una base de conocimientos formada por: 1) los operadores de cambio de estado, 2) una base de datos en la que se caracteriza el estado final y los objetivos y 3) un mecanismo de inferencia.

El sistema tutor propuesto por Peachey y McCalla (1986) está formado por cinco componentes: 1) base de conocimientos del dominio, 2) modelo del estudiante, 3) colección de operadores de enseñanza, 4) planificador y 5) ejecutor.

El planificador construye el plan mediante una secuencia de pasos para lograr un objetivo instruccional. Cada paso es una instancia de un operador y cada operador tiene una acción asociada. El ejecutor ejecuta el plan invocando las acciones asociadas a cada paso del plan. El planificador crea su plan de enseñanza simulando los efectos de las acciones de los operadores sobre el modelo del estudiante. Un operador es semejante a una

regla en un sistema de producción e incluye un conjunto de precondiciones y un conjunto de efectos esperados. Mediante el uso del operador el planificador simula la llamada de los pasos del plan de enseñanza y considera los resultados que puede obtener sobre un modelo de estudiante virtual.

Realmente no hay conexión directa entre los efectos esperados de un operador y los efectos reales que produce. El ejecutor debe detectar y recuperar estas desviaciones, haciendo uso de las opciones construidas dentro del plan o inclusive llegando a re-invocar al planificador para revisar un plan que no ha tenido éxito.

Fernández (1989) propone un tipo de planificador dinámico. En este tipo de sistema se separan las técnicas de planificación, de la resolución del problema. Una de las características de este sistema es que evita los costos de re-planificación cada vez que el camino trazado no se ajuste a la realidad.

Se establecen unos objetivos pedagógicos en la sesión. Para lograrlos se eligen unas estrategias de enseñanza relacionadas con un conjunto de planes. Cada vez que se interacciona con el alumno, el sistema es capaz de detectar si existen conflictos, en cuyo caso se reconsideran los objetivos a nivel local o global.

Contiene estrategias de enseñanza basada en estados, que son independientes del dominio a enseñar, con el fin de que pueda ser utilizado en otro dominio de naturaleza estructurada. El control de las estrategias se presenta de forma más flexible, ya que no existe una forma preestablecida de transiciones entre estados. Las transiciones se establecen en términos de los planes y las reglas de resolución de conflictos.

El planificador dinámico propuesto por Fernández (1989) está compuesto por la cooperación de 4 módulos: 1) decisión pedagógica, 2) decisión temática, 3) módulo de enseñanza y 4) módulo supervisor.

La decisión pedagógica se encarga de: 1) seleccionar la combinación de estrategias de enseñanza más adecuada, para conseguir los objetivos del alumno, teniendo en cuenta información del usuario, 2) desarrollar las estrategias establecidas, generando subobjetivos progresivamente y 3) comunicarse con el alumno sobre la elección de distintas actividades didácticas a desarrollar.

La decisión temática es el componente que refina los objetivos establecidos por otros módulos, tomando en consideración el domino objeto de la enseñanza.

El módulo de enseñanza explica o verifica los conceptos seleccionados por decisión temática. También se ocupa de la comunicación y la actualización del modelo del alumno.

El módulo supervisor actúa cada vez que se produce una interacción usuario-sistema. Para detectar posibles conflictos entre los objetivos del usuario y los objetivos establecidos por el tutor (en curso o en espera). Selecciona una estrategia local o notifica la necesidad de replantear la estrategia.

A partir de los ochenta apareció una forma diferente de hacer frente a los problemas de la inteligencia artificial. Su nombre es la filosofía reactiva Brooks (1991a), Brooks (1991b), y Laureano (1998). Uno de los objetivos en los sistemas reactivos es lograr la distribución del control entre los agentes, para lograr que el ciclo de toma de decisión sea lo más rápido posible. Hay que hacer hincapié que en el caso de los agentes reactivos no existe un modelo simbólico exhaustivo del entorno sobre el que el sistema pueda razonar.

El modelo sobre el que toma decisiones es el mundo real. De aquí que el control de un sistema reactivo dependa de los objetivos de cada agente y su interacción con el mundo real.

En Makatsiná no existe el significado de control en los términos mencionados anteriormente. Este se diseñó de acuerdo a los principios de la filosofía reactiva Laureano y de Arriaga (2000). De forma general consiste en: agentes que disparan su acción en base a la percepción del entorno (errores) y a los objetivos de cada agente. Una de las principales aportaciones del desarrollo de Makatasiná, consiste en la propuesta de una arquitectura MultiAgente, basada en agentes reactivos Laureano y de Arriaga (1997) y Laureano y de Arriaga (1998b).

En Makatsiná el control funciona en base a una jerarquía basada en errores, que se estableció de acuerdo al orden de aprendizaje de diferentes subhabilidades.

El estilo de aprendizaje

Se refiere a la mejor forma de enseñar un determinado dominio. Sobre esta cuestión existe gran cantidad de teoría desarrollada dentro del campo de la educación por pedagogos y psicólogos. Generalmente se acude a experiencias para desarrollar técnicas que de una u otra forma den mejores resultados. Por ejemplo es conocida la capacidad de los estudiantes de ingeniería para captar en un problema aspectos visuales implícitos, significativos, y utilizarlos en la resolución de éste.

En este caso el uso de métodos visuales resaltaría estas características. Actualmente se puede recurrir a técnicas que permiten conocer más acerca

de la visualización y con ello orientar a los estudiantes hacia el desarrollo de esta capacidad, que les permitiría un mejor aprendizaje.

En Alonso y Gallego (1994) se hace una reseña histórica de definiciones y aportaciones, que finaliza con el desarrollo de un cuestionario. Este cuestionario puede ser aplicado a grupos de estudiantes y con ello conocer las preferencias del estilo de aprendizaje. El cuestionario distingue entre cuatro estilos de aprendizaje: activo, reflexivo, teórico y pragmático.

Estos estilos a su vez representan las experiencias que se tienen durante el proceso del aprendizaje: vivir la experiencia, reflexión, generalización y elaboración de hipótesis y, finalmente, aplicación. Sin embargo, existen diferencias claras con base en las diferentes áreas del conocimiento. Podríamos pensar que el hecho de que las distintas Facultades Universitarias, agrupadas de acuerdo a como manejan el conocimiento en carreras: técnicas, de humanidades y experimentales, se inclinen hacia un tipo dado de aprendizaje, está directamente relacionado con el tipo de dominio y la forma de utilizarlo en la práctica. Conocer el estilo de aprendizaje de los estudiantes y sus preferencias, implica diseñar cursos que aporten mejores resultados. En el caso concreto de los SEIs, podríamos diseñarlos haciendo énfasis en estos estilos de aprendizaje.

El estilo de enseñanza

Representa la contraparte del aprendizaje y debe ser diseñado de forma tal que saque ventaja del estilo de aprendizaje. Debemos ver el proceso de enseñanza-aprendizaje como un mecanismo de comunicación que debe darse entre dos agentes. De aquí que el proceso tutorial se aborde como un proceso de comunicación Girard, Gauthier y Levesque (1992), más que como un proceso de enseñanza convencional.

El tipo del dominio

Se refiere a la diferencia que existe entre los diversos tipos de tutores: expositores y procedimentales. En un tutor expositor el énfasis se encuentra en el conocimiento factual (representa los hechos que conforman al conocimiento declarativo) y a partir de ese conocimiento se infieren las habilidades de primer orden que pertenecen a la clasificación de conocimiento declarativo.

Los tutores procedimentales se encargan de enseñar habilidades y procedimientos, que tienen aplicaciones en el mundo real. Estos tutores funcionan más como entrenadores, presentando ejemplos y mostrando el uso de las habilidades a través del desarrollo de problemas. Dentro de su contenido se encuentran problemas orientados a pruebas y prácticas especiales.

Este problema, como ya se mencionó, se aborda desde dos perspectivas: 1) encontrar una adecuada representación para el material a enseñar, 2) Seleccionar y secuenciar ese material.
El primer problema está relacionado con el conocimiento referente a las instrucciones de la enseñanza y con el módulo experto. Y el segundo está enlazado con el tipo de control (sección 4.1.1).

Selección y secuenciación

En cuanto a la selección y secuenciación, existen diferencias de acuerdo al tipo de tutor. En los expositores el problema se centra en mantener la coherencia en la presentación del material para su posterior reflexión. Los tutores procedimentales tienen además el problema de tener que ordenar las subhabilidades integrantes de la habilidad-objetivo y seleccionar correctamente los ejercicios y ejemplos que reflejen ese orden.

Selección y secuenciación en los tutores procedimentales

Este tipo de tutores basan su enseñanza en ejercicios y ejemplos. En este caso el punto central es contar con mecanismos adecuados de selección y secuencia del material. Lo ideal sería elegirlos de acuerdo al estilo de aprendizaje, pero, según Anderson (1988), no existe una teoría de aprendizaje que sea lo suficientemente precisa y poderosa para sustentar un proceso tutorial interactivo.

Las investigaciones recomiendan tener en cuenta los siguientes puntos con relación a los ejercicios y ejemplos: a) Manejabilidad. Todos los ejercicios deben tener solución y ser comprendidos por un estudiante que haya cubierto el material previo, b) Transparencia estructural. La secuencia de ejercicios y ejemplos debe reflejar la estructura del proceso a ser enseñado y deberá tener la cualidad de guiar al estudiante en la adquisición de la habilidad-objetivo y c) Individualización. El ejercicio presentado al estudiante debe poseer la(s) subhabilidad(es) que el estudiante ya maneja y poder relacionarse de forma fácil con la(s) subhabilidad(es) que quiere adquirir en esos momentos.

VanLhen (Halff, 1988) expone una teoría llamada 'step-theory', en la que habla de que el curriculum debe estar dividido de acuerdo a aspectos o características importantes. Estas divisiones representan los pasos-de-teoría. Cada lección podrá utilizar solo un paso-de-teoría, de forma que el procedimiento a ser aprendido saque ventaja de la división del curriculum por pasos. Asimismo, se habla de las felicity conditions o la capacidad de encontrar las divisiones del dominio que más ayuden al proceso tutorial. Esta división coincidirá con un paso en el procedimiento que deba ser aprendido.

En relación con esto, se considera que lo que se necesita es la ayuda de un buen análisis cognitivo de tareas (ATC) para encontrar esas rupturas críticas

del dominio que ayuden a esclarecer puntos en la enseñanza como los expuestos en Castañeda (1993), Redding (1992) y Ryder & Redding (1993).

En cuanto al proceso de selección y secuenciación del material, es importante diseñar el curriculum para que cumpla con las siguientes funciones: 1) contener el material dividido, de forma que sea fácil su manipulación, con base en objetivos instruccionales, 2) secuenciar el material, de forma que su estructura convenga al usuario, 3) asegurar que los objetivos planteados en cada unidad puedan ser logrados, y 4) actualizar estos últimos como consecuencia de los mecanismos de evaluación del impacto del proceso tutorial en el estudiante.

El proceso tutorial (enseñanza)

Este componente incluye, las funciones de cubrir la presentación del material, formas de poder contestar al estudiante preguntas y las condiciones y el contenido de la intervención del proceso tutorial.

Métodos de presentación

Los métodos de presentación dependen del tipo del dominio a enseñar y de los objetivos que se desean alcanzar durante el proceso tutorial.

Los tutores expositores utilizan diferentes formas de diálogo. Los tutores procedimentales orientados al manejo de habilidades, utilizan ejemplos y ejercicios con entrenador, para lograr el manejo de esas habilidades.

Las diferentes formas de diálogo implican distintos objetivos de enseñanza Halff (1988). Collins propone una guía para la selección del dialogo, de acuerdo a los objetivos de enseñanza. Esta clasificación coincide con las propuestas de Gagné y Merrill, y se muestra resumida en la Tabla .

Tabla Estrategias de Diálogo, de acuerdo a diferentes objetivos instruccionales según Collins (Halff, 1988)

| Objetivos Instruccionales | Estrategias |
|---|---|
| Enseñar hechos y conceptos | Separar hechos o conceptos<br>Explicar hechos o conceptos |
| Enseñar reglas y relaciones | Estrategias de selección<br>Trampear |
| Enseñar habilidades por inducción | Ejercicios y ejemplos orientados a mostrar la subhabilidades |

Intervención tutorial

La intervención tutorial se necesita para mantener el control de la situación durante el desarrollo del proceso tutorial, con el objetivo de situar al usuario lejos de un aprendizaje inapropiado o incorrecto y colocarlo al margen de caminos fuera del objetivo de enseñanza. Automatizar este proceso implica diseñar reglas para decidir la intervención o la no intervención, además de formular el contenido de ésta.

Existen básicamente dos formas para guiar una intervención tutorial, ya comentadas como técnicas de diagnóstico. Estas son: 1) seguimiento de una ruta (Model-Tracing) y 2) seguimiento de asuntos (Issue-Tracing).

Intervención con seguimiento de una ruta

En este tipo de intervención se cuenta con los caminos que podría elegir el usuario en la solución de un problema. Se revisa el comportamiento de ambos tratando de emparejar el desarrollo del usuario con alguno de los caminos que se pueden tomar. Cuando el emparejamiento fracasa, el tutor interviene emitiendo un consejo que permite al alumno retomar el camino correcto. La desventaja es que el tutor intervendrá siempre que no pueda reconocer el camino seguido por el usuario, aún cuando éste sea mejor.

Intervención con seguimiento de asuntos

Esta forma de intervención tutorial fue desarrollada por Burton y Brown (1982) e implementada en WEST. Aquí las habilidades o conocimientos que se desea aprenda el estudiante, se llaman asuntos (Issues).

Cada asunto representa una mini-teoría articulada, esto es, un pedazo del experto visto como caja de cristal, que se caracteriza por dos procedimientos.

El primer procedimiento tiene el papel de observador del desarrollo del estudiante. Su misión es obtener evidencia para saber si el estudiante usa o no ese concepto o habilidad particular. A este procedimiento se le denomina reconocedor de asuntos y sus observaciones se utilizan para construir un modelo del desarrollo del estudiante.

El segundo procedimiento sabe cómo utilizar la información vertida en el modelo del estudiante para decidir si éste domina o no el asunto. A este procedimiento se le denomina evaluador de asuntos.

En el caso concreto de WEST, se utilizó el tutor tipo entrenador. Para conocer el progreso del estudiante se ejecutaban todos los evaluadores del modelo.

Cuando el estudiante ejecuta una mala jugada, ésta conlleva una debilidad o carencia de habilidad, ya que no fue buena. Entonces su debilidad es comparada con las habilidades (asuntos) necesarias para ejecutar una mejor jugada y de esta forma encontrar por qué no la realizó.

La Figura 1, compuesta por una parte 1(a) y otra 1(b), presenta el proceso del modelado tutorial, utilizando asuntos y ejemplos. En la Parte 1(a) se muestra el proceso de construcción de un modelo del desarrollo del estudiante durante la resolución de una serie de problemas. En el caso particular de WEST, son jugadas, ya que el aprendizaje se realiza en el contexto de un juego.

Cada vez que el estudiante realiza una jugada, el reconocedor de asuntos elabora la abstracción de los aspectos importantes del desarrollo en el estudiante y en el experto del sistema en el mismo entorno, utilizando para ambos los mismos reconocedores de asuntos. Se comparan las dos abstracciones para obtener un modelo diferencial que permita conocer los asuntos que el estudiante no domina.

Una vez que ha sido determinado en qué asunto falla, el entrenador presenta una explicación de ese asunto, seguida de un ejemplo de uso. Lo que se persigue con este tipo de intervención es que el estudiante reciba la información cuando esté más receptivo a procesarla, en otras palabras, nada más cometer la falla.

Es importante resaltar que sin el experto no sería posible determinar el grado de dominio de un asunto (habilidades o conocimiento), por parte del estudiante o si el estudiante no lo ha utilizado por falta de experiencia en su uso.(ver figura 1)

La Parte 1(b) representa el nivel de abstracción mas alto en la intervención tutorial con un tipo entrenador. Cuando el estudiante realiza una jugada no considerada como óptima (comparándola con el experto), el entrenador utiliza al evaluador de asuntos para crear una lista de asuntos en los cuales el estudiante anda mal, por otro lado cuenta con la lista del experto sobre las mejores jugadas.

Entonces el entrenador invoca al reconocedor de asuntos para determinar que asuntos están registrados en las mejores jugadas. A partir de estas dos listas, el entrenador selecciona un asunto de la lista de asuntos que le fallan al estudiante y una jugada buena del experto que contenga este asunto para mostrarla. Si no existen asuntos en común en las dos listas, se deduce que el problema del estudiante esta fuera del alcance del tutor y el entrenador no dice nada.

Con base en otros principios tutoriales se decide interrumpir o no al estudiante, pero, si el entrenador lo decide, el asunto y el ejemplo se pasan

al alumno. Se espera que esta acción genere una retroalimentación en el estudiante.

Un SEI puede utilizar una de estas dos técnicas para la intervención tutorial o combinarlas, lo que sucede frecuentemente. Un ejemplo es Makatsiná Laureano y de Arriaga (1998b), que utiliza la técnica de seguimiento de asuntos para la intervención tutorial, y ésta es combinada con un tutor tipo entrenador, que detecta las habilidades que el estudiante utiliza en forma errónea. Esta última se basa en el mecanismo propuesto por Burton y Brown (1982).

La Interfaz

Representa un elemento muy importante en la arquitectura de los SEI, ya que cubre varias actividades en el funcionamiento global del sistema:

· Es el puente de comunicación entre el estudiante y el sistema.
· Es el único medio físico para captar el desarrollo del estudiante.

Las siguientes actividades tienen enfoque de herramientas didácticas.

· Representa el medio a través del cual el tutor (sistema) realizará las intervenciones.
· De acuerdo al dominio de enseñanza, el potencial de la interfaz debe ser explotado al máximo, utilizando los medios más adecuados (video, audio, etcétera) para la mejor comprensión de los conceptos o el manejo de habilidades.

La ayuda

Esta ayuda se refiere básicamente a la proporcionada cuando el estudiante esta resolviendo algún problema. Los SEI cuentan con diferentes tipos de ayuda:

Ayuda del sistema

La gran mayoría de las interfaces de los SEI cuentan con este tipo de ayuda que es útil cuando el estudiante la solicita o cuando se cometen errores. Por ejemplo, MACSYMA Genesereth (1978) construye un plan que explica las acciones que pueden haber conducido al error y propone una lista de probables "misconceptions" causantes de que el usuario cometiera ese error.

Asistencia

Se trata de mecanismos que permiten desarrollar parte de la tarea o la totalidad de ésta. Este tipo de ayuda deja al estudiante concentrarse en las partes donde encuentra dificultad y dejar las partes que domina al sistema. En Algebra Land de Brown & Foss y Algebra Tutor de Anderson Burton (1988), los estudiantes cuentan con esta opción que les permite ver como son realizadas las operaciones.

Herramientas de poder

Se refiere a la captura de decisiones y acciones del estudiante en estructuras para su posterior observación. Este tipo de mecanismo ayuda al estudiante a reflexionar en las actividades desarrolladas durante la solución del problema. Algebra Land y Geometry Tutor de Anderson & Skwarecki Burton (1988) cuentan con esta ayuda. En el caso particular de Algebra Tutor cada decisión del estudiante es capturada en una estructura de árbol, que le permite visualizar el espacio de búsqueda.

Reactivo

En este caso el sistema dispone de mecanismos que responden inmediatamente a la acción del estudiante, en el contexto de la situación específica. SOPHIE-I de Brown , Burton y Bell (1975). Un ejemplo de entorno reactivo es reaccionar a la solicitud de evaluar las hipótesis relativas a las medidas que el estudiante tiene. No le dice que esta mal, dado que la lógica puede ser correcta de acuerdo a las medidas que tiene el estudiante. En el caso de que sea inconsistente la hipótesis propuesta, lo confronta con ejemplos que tienen esa característica (de forma correcta) y que el estudiante está pasando por alto. Una característica de estos sistemas es que permiten al estudiante articular sus hipótesis en forma opuesta a aquellas que ellos realizan.

Modelado

En este caso el sistema desarrolla la tarea mientras el estudiante observa. La ayuda permite al estudiante observar cómo se comportaría un experto en el desarrollo de la tarea. Para lograrlo es necesario que el modelo del experto articule las decisiones y que éstas coincidan con las estrategias de selección de esas decisiones. El experto articulado de SOPHIE - II de Brown, Burton y deKleer (1982) es un buen ejemplo, ya que permite al estudiante crear una avería en el circuito y hacerlo funcionar. El estudiante realizará las medidas hasta donde le sea posible y posteriormente el experto explicará cada medida: el por qué se ha realizado (con base en sus estrategias) y al análisis cualitativo del sistema.

Entrenador

El tutor se ha desarrollado con el objetivo de seguir de cerca al estudiante en la realización de la tarea, e interrumpirlo cuando esta realización no sea óptima o se haya equivocado, proporcionándole sugerencias en estos casos. Es importante mencionar que para la implementación de este tipo de mecanismo, es necesario contar con el camino del experto. Ejemplos ya clásicos son WUSOR y WEST de Burton y Brown (1982). Ambos tutores se interrumpen cuando el estudiante realiza un mal movimiento y dan consejos.

Crean el modelo del estudiante comparando el desarrollo de éste con el del experto. Con WEST se demostró que, aunque no se pueda rastrear todo lo que el estudiante hace, se pueden reconocer patrones de un desarrollo no óptimo.

Existe también la posibilidad de combinar varios tipos de ayuda, como es el caso de Makatsiná Laureano y de Arriaga (1998a) y Laureano y de Arriaga (1999), donde se manejan varios tipos de ayuda, que son herramientas de poder (se guardan todas las acciones desarrolladas por el estudiante, permitiéndole observar en cualquier momento los puntos de decisión), modelado (en cualquier instante cuenta con la opción de poder ejecutar parte o toda la tarea por el experto) y entrenador (filosofía con la que fue desarrollado el modelo tutorial).

Multimedios y cognición

La interfaz es tan importante que podríamos pensar en ella en función de los sentidos que captan su funcionamiento al interactuar con algún sistema.

Los Mutimedios como herramienta didáctica

En Rickel (1989) se menciona que las personas retienen aproximadamente el 25% de lo que escuchan, el 45% de lo que escuchan y observan, y el 70% de lo que escuchan, observan y desarrollan. Estos datos representan un potente argumento para desarrollar interfaces que incluyan texto, sonidos y gráficas, además de que tengan la capacidad de generar una interacción con el estudiante como si éste fuese parte del sistema.

Un SEI que utiliza este tipo de interfaz es el citado en Rickel (1989). Los autores del sistema consideran que la descripción gráfica utilizada en relaciones causales y en topologías, son muy importantes. Algunos conceptos pueden ser presentados utilizando gráficas, lo que facilita la comprensión del alumno. Por otro lado, manejar una interfaz con base en ideogramas permite una conexión más profunda entre el estudiante y el sistema.

El aprendizaje por asociación (cognición)

Es importante mencionar que el uso de multimedia tiene una relación directa con el aprendizaje por asociación, como se menciona en Laureano (1993). La memorización por asociación puede ser mecanizada. Esta reflexión nos conduce directamente al uso correcto de multimedia, con el objeto de mecanizar conocimiento procedimental.

El método de aprendizaje por asociación, descrito en el año 55 a. de c. por Cicerón y ampliado por Godden y Baddeley, indica que la memoria no sólo depende de un conjunto de trucos formalizados para recordar elementos ubicados en distintas localidades físicas, sino que está basada en algo más profundo que depende del contexto de aprendizaje. De la reflexión anterior se desprende la importancia de modelar interfaces con una fidelidad de display profunda, sobre todo en principiantes.

Sin embargo, el hecho de que la asociación juegue un papel importante en la forma de ligar nuestros esquemas mentales personales en un todo coherente, no implica que ésta sea la mejor forma de impartir información y conocimiento. Una evidencia indirecta contra los métodos asociativos es el éxito, durante mas de 500 años, del libro convencional y el desarrollo de la narrativa durante más de mil años.

Esto por otro lado tiene las siguientes desventajas: a) no existe una respuesta inmediata al estudiante, b) los estudiantes son entes pasivos y c) no se generan ejercicios de distintas complejidades de acuerdo a la habilidad de los estudiantes.

Estas desventajas se convierten en ventajas al hablar de los SEI. De ahí que los sistemas multimedia, por el momento, no sean más que un conjunto de técnicas experimentales útiles en casos específicos, que se separan de la corriente principal del pensamiento educacional convencional.

Resumiendo, en la interfaz de un SEI se pretende que la entrada de la información sea robusta, pero a su vez que permita una flexibilidad en la entrada de información sintetizada, que se desprende de la observación del desarrollo del estudiante. En cuanto a la información de salida, ésta debe ser rica, esto es, deletreada al máximo, utilizando la potencia de los sistemas multimedia y aprovechando así todos los sentidos del estudiante, sin caer en el exceso.

Como ejemplo de interfaz que cumple con los aspectos anteriores podemos citar la de Makatsiná Laureano y de Arriaga (1998a) y Laureano, Arriaga y Martínez (1999) donde se hizo especial hincapié en el desarrollo de una interfaz controlada a través de ideogramas.

Las intervenciones tutoriales, y la interacción del estudiante, se hacen a través de gráficas, debido a que el dominio de enseñanza es considerado ad-hoc para ser manejado por ideogramas. En el pizarrón se hace una

explicación con gises de colores y un continuo escribir al lado de las líneas que conforman la estructura triangular, para mostrar los sentidos de las fuerzas de equilibrio, donde también éstas tienen diferentes colores, de acuerdo al tipo de esfuerzo (tensión o compresión).

Conclusiones

En el desarrollo de cualquier sistema que conlleve técnicas de inteligencia artificial, se presentan, a nivel general, dos problemas: 1) no se toman en cuenta aspectos emocionales y sociales y 2) derivado del punto anterior y por la naturaleza propia de estos sistemas, en realidad se está trabajando con información incompleta.

Parte de estos problemas se tratan de solucionar con las técnicas explicadas a lo largo del artículo, pero aún estamos lejos de percibir las emociones humanas. En investigaciones recientes se ha tratado de subsanar estos dos problemas con herramientas de psicología cognitiva y métodos para tratar con información incompleta, como son los conjuntos difusos, o la misma filosofía reactiva.

Por ejemplo, para llevar a cabo la evaluación dentro de los SEI, hasta el momento sólo se están tomando en cuenta aspectos concretos, producto de la interacción de enseñanza-aprendizaje. Sin embargo, quiero hacer hincapié en que las técnicas que permiten la implementación de la evaluación en los SEI, porvienen de campos como: la educación y la psicología cognitiva.

La mayoría de las veces que se desarrolla un plan de enseñanza-aprendizaje, pasamos por alto aspectos como el nivel de competencia con que llega un alumno a aprender un determinado dominio, inclusive aspectos tan importantes como los motivacionales, sociales y emocionales. Al pasar por alto este tipo de aspectos, estamos dejando sin validez una evaluación real al plan de enseñanza-aprendizaje, concebido sólo tomando aspectos cognitivos del dominio.

Si los usuarios no están motivados o tienen problemas emocionales, así les apliquemos el mejor plan de enseñanza-aprendizaje, fracasarán. Sería muy interesante tratar de evaluar en este tipo de sistemas los aspectos de habilidades afectivas, motivacionales y sociales Castañeda et al. (1999), además de las cognitivas para lograr un incremento en la capacidad de aprendizaje.

Actualmente los autores trabajan en el desarrollo e implementación de un SEI, que incorporará un tipo de evaluación basada en conjuntos difusos, esto enmarcado en una arquitectura Multiagente diseñada para SEI Laureano y de Arriaga (2000) y Laureano y de Arriaga (1998b).

**Buggy (1979)**

Un sistema Tutor Inteligente (ITS como comúnmente se conoce en inglés) es un sistema experto en una materia, diseñado con el fin de impartir conocimiento y que exhibe alguna forma de inteligencia para guiar al estudiante en el proceso de aprendizaje. En este artículo se presenta una introducción al área ITS y se describen y comentan sus componentes.

Un sistema Tutor Inteligente (ITS) es un sistema experto en una materia, con el fin de impartir conocimiento (ITS). El propósito es exhibir un comportamiento parecido al de un tutor humano, es decir, que se adapte al comportamiento mostrado por el estudiante en vez de responder rígidamente a un patrón previamente establecido.

Un ITS debería ser capaz de identificar la forma en que el estudiante está solucionando el problema y brindarle la ayuda necesaria cuando cometa errores, determinar el conocimiento que necesita para resolver un problema y explicar ese conocimiento en el momento apropiado; aprender por medio de la interacción con el estudiante.

La idea es, entonces, producir sistemas expertos que puedan simular el comportamiento del estudiante, de manera que cuando el alumno dé una respuesta, el sistema pueda simular el comportamiento del tutor para poder guiarlo adecuadamente. Enseñar es una tarea muy difícil e intuitiva, y la construcción de un sistema que pueda simular totalmente el comportamiento de un maestro, es, tal vez, imposible, a pesar de que estos sistemas llevan por nombre “tutores inteligentes”.

Su propósito no es de ninguna manera reemplazar al maestro, sino servir como una herramienta eficaz y de valor tecnológico en el proceso de enseñanza.

El enfoque de los ITS difiere, sustancialmente, del enfoque tradicional de Instrucción Asistida por Computadora (CAI), en el cual tanto la materia como el conocimiento del maestro están pre-programados en el sistema, sin ofrecer flexibilidad para adaptarse a diferentes situaciones y comportamientos de los estudiantes.

La idea de los ITS es desarrollar “software” que capture el conocimiento mismo que le permita a los expertos componer y desarrollar una situación de enseñanza-aprendizaje.

Wenger (1987) define la tarea de enseñanza como una labor de comunicación de conocimiento. Esta labor se refiere a una situación que involucra a un sistema tutor (o a un maestro) y a un estudiante, donde el objetivo de comunicación es el conocimiento de alguna materia.

En teoría, aparecen diferentes propuestas para la arquitectura de los ITS (Bregar y Farley, 1980, Lawler y Yasdani, 1987; Wenger, 1987). A pesar de las diferencias de formato, todas estas propuestas sugieren la presencia de cuatro componentes principales, enmarcados en la propuesta de Wenger: el experto en la materia, el modelo del estudiante, el modelo pedagógico y la interfaz con el usuario. Además, se incluye otro elemento importante: el proceso de diagnóstico, entendido como el reconocimiento de la forma que el estudiante usa para resolver problemas y de su conducta general.

Sirve para construir el modelo del estudiante e influye en las decisiones pedagógicas que se tomen. Todos estos componentes aparecen de alguna manera en la estructura de un ITS. Ha continuación se describe cada uno de ellos.

EL EXPERTO EN LA MATERIA

Este componente del sistema contiene la representación del conocimiento que va s ser comunicado. Se encarga de generar problemas con sus respectivas soluciones, incluyendo pasos intermedios y vislumbrar diferentes formas de solucionar el problema.

En sistemas CAI tradicionales, el conocimiento de la materia está contenido en episodios pre-programados, los cuales aparecen en secuencia, independientemente de la forma particular que tenga el estudiante de resolver los problemas que se presentan.

Por el contrario, en los ITS, el módulo experto contiene la representación del conocimiento que se va a comunicar, es decir, una descripción de los conceptos y las habilidades que se van a transmitir al estudiante, la forma en que estos conceptos y de los cuales depende el educando en la materia.

Obviamente, el lenguaje particular de la materia que está siendo estudiada va a estar pre-determinado (por ejemplo, si el sistema trata de la ebullición del agua, entonces manejará términos como “punto de ebullición” “temperatura” y “vapor”); además, el sistema deberá asumir una visión de la materia, así como el maestro tiene también su propia visión.

Tanto más amplia sea esta visión, más flexible será el sistema y más conocimiento tendrá que ser presentado e identificado, lo cual conlleva a problemas comunes en el área de Inteligencia Artificial, como son proveer un diagnóstico preciso y alcanzar soluciones en un tiempo apropiado.
ITS de uso práctico tendrán que hacer un balance entre la cantidad de conocimiento que se presente y la efectividad del sistema.

Una característica deseable del experto es la capacidad de hacerse transparente al estudiante, por medio de explicaciones de sus acciones, a cada paso, si la explicación se solicita o si el módulo experto considera que la explicación es conveniente para el estudiante. SHOPIE, un sistema que fue diseñado para

diagnosticar fallas en la materia de circuitos electrónicos (Brown et al., 1982), es un ejemplo que incluye esta característica, pues permite simular un modelo de un circuito electrónico capaz de ofrecer demostraciones para que el estudiante reconozca la forma correcta de solucionar el problema.

## EL MODELO DEL ESTUDIANTE

El propósito de este modelo es proveer información sobre el conocimiento que tiene el estudiante de la materia. Esa información será usada por el modelo pedagógico, el cual la compara con el conocimiento que tiene el estudiante de la materia.

Esa información será usada por el modelo pedagógico, también compara con el conocimiento del experto para determinar cuáles son los caminos posibles que la sesión puede tomar y así brindar la guía apropiada. Este modelo se deriva paulatinamente por medio de la interacción del estudiante con el sistema.

De la misma forma en que el maestro puede determinar cuáles son los aspectos de la materia que el estudiante domina y cuáles los que todavía no domina, el modelo del estudiante también identifica esos aspectos y distingue las versiones correctas de las incorrectas, necesarias para remediar las fallas del estudiante.

Brown y Burton (1978) hicieron una contribución significativa al respecto. Desarrollan BUGGY, una teoría de los conceptos erróneos (“bugs”) que usualmente tienen los estudiantes en aritmética, y DEBUGGY, un sistema que, basándose en BUGGY, hace diagnósticos de los errores que cometen los estudiantes.

En este último, el procedimiento de diagnóstico da como resultado el estado actual del conocimiento de la materia que tiene el estudiante, e incluye tanto el conocimiento correcto como el incorrecto. A pesar de que podría parecer injusto representar el conocimiento del estudiante solo como un conjunto de conceptos correctos y erróneos (Yasdani, 1978), el trabajo desarrollado por Brown u Burton fue un gran avance para la representación del modelo del estudiante, ya que a los “bugs” sirven como parámetro de comparación para poder derivar el modelo.

Si se sigue esta idea, una fase es el desarrollo de un ITS sería la construcción de una base de conocimiento, que incluya posibles conceptos erróneos que los estudiantes podrían manifestar al aprender una materia.

El proceso de diagnóstico y el modelo del estudiante son partes complejas en el desarrollo de un ITS (Burton, 1979, Wenger, 1978). La razón principal es que, cuando el estudiante aprende, las hipótesis que se tenían sobre el estado del conocimiento del estudiante ya no se pueden mantener, y el sistema debería ser, entonces, capaz de actualizar, dinámicamente, el modelo del estudiante.

Además, el proceso de diagnóstico en sí podría requerir de mucho tiempo, ya que, tanto más información maneje el sistema, más podría ser el tiempo requerido para

buscar y comparar datos, con el fin de identificar una falla o conducta del estudiante. Sin embargo, hay métodos que se usan en Inteligencia Artificial, como los "heurísticos" (métodos empíricos que se usan para buscar información por medio de caminos que aparentemente pueden conducir a una solución y que son, generalmente, más rápidos que el uso de un algoritmo, pero también más falibles) que se podrían tratar de encontrar para aliviarse este problema.
Esta es, sin duda, un área de gran interés para la investigación.

Una vez que se tiene el diagnóstico, será la labor del experto y del modelo pedagógico determinar la forma en que se va a guiar al estudiante, con el fin de que pueda corregir su error.

EL MODELO PEDAGÓGICO

Como se mencionó una de las grandes ventajas de los ITS es que se pueden adaptar a las diferencias individuales de los estudiantes. Por ejemplo, un ITS brindará explicaciones, ejemplos y ejercicios, dependiendo del conocimiento que el estudiante tenga de la materia. El curso que siga el sistema, dependerá de las decisiones didácticas que se tomen, con base en las referencias del modelo del estudiante y del experto en la materia.

Las decisiones didácticas pueden ser de dos tipos: generales y específicas. Las generales se basarán en principios que son independientes de la material que se esté estudiando. Por ejemplo "si el estudiante está cometiendo muchos errores, ajuste el nivel de los problemas", o diga frases de aliento cuando el estudiante supera un problema o cuando está trabajando muy bien.

Las decisiones específicas dependen de la materia en sí. Por ejemplo, en porcentajes, si el estudiante ha estado confundiendo el uso de la división con la multiplicación y, al resolver el problema, dividió en vez de multiplicar, entonces hará referencia a un problema anterior ya resuelto correctamente e inducirá al estudiante a razonar para que reconozca su error.

De esa forma, el conocimiento pedagógico estaría contenido en módulos didácticos, los cuales adaptarían la presentación de la materia de acuerdo con las necesidades de los estudiantes, y determinaría cuándo y cómo el tutor debería intervenir.

Otro aspecto pedagógico es ¿cuándo el tutor debería intervenir? ¿Será el autor el que tenga el control de la sesión o será el estudiante? ¿Hay investigaciones que indican que los estudiantes que tienen poco conocimiento de la materia, generalmente, no se desempeñan bien cuando tienen todo el control de la sesión, ya que, obviamente, cuanto menos conozcan la materia, más instrucción necesitarán. (Steinberg, 1989).

Estas investigaciones también sugieren que, como aprender es un proceso dinámico, el control de la sesión cambie del estudiante a la computadora y viceversa, de acuerdo con las necesidades de cada individuo.

La mayoría de las investigaciones en ITS presentan sistemas que permiten al estudiante tener algún grado de control sobre la sesión. Por ejemplo, SCHOLAR (Wenger, 1987), desarrollado por Jaime Carbonell en 1970 y considerando la primera contribución importante para el área de educación asistida por computadora, es un sistema que enseña geografía de América del Sur, y conduce la sesión por medio de un "diálogo de iniciativa mixta", en donde tanto el sistema como el estudiante intercambian preguntas y respuestas.

Otros sistemas permiten que el estudiante tome el control de la sesión y lo interrumpa cuando el sistema determine que es necesario dar asistencia. Este tipo de sistemas se conocen como "sistemas entrenadores", ya que actúan en forma similar a un entrenador.

Un ejemplo es WUSOR (Goldstein, 1982), un sistema que actúa como un entrenador para jugar WUMPUS. Este juego consiste en un viaje de cuevas vecinas y con esa información, el jugador tiene que decir cuál es la siguiente cueva que va a visitar para escapar de los peligros. El propósito del sistema es ejercitar la capacidad del estudiante para hacer inferencias.

También se han desarrollado sistemas simuladores que proveen un ambiente para experimentar y que, por su naturaleza, permiten que el estudiante tenga bastante control sobre la sesión, SOPHIE es un ejemplo de este tipo de sistemas.

Otro ejemplo clásico es STEAMER (Wenger , 1978), el cual consiste en una simulación de una planta de vapor y un laboratorio para experimentar con diferentes componentes de la planta.

En este sistema, el estudiante manipula componentes de la planta y datos, como presión y temperatura, y el tutor hace comentarios sobre las manipulaciones que hizo el estudiante.

Veremos en las siguiente sección que, en general del tipo de control que se le dé al estudiante depende el tipo de interfaz que el sistema tenga.

## INTERNET 2

Terreno de disputas
A casi 15 años de que el acceso a internet se abriera al público en general, cuando todavía mucha gente empieza a conocerlo y a cambiar hábitos de entrenamiento, comunicación y consumo, la aparición de la segunda versión de la red de redes ya ha generado grandes expectativas sobre su capacidad y nuevas formas de conexión. Y, por supuesto, las empresas ya la miran como un atractivo campo de comercios. Todos quieren llevarse la mayor tajada del nuevo pastel llamado Internet 2.

Creada en Estados Unidos para experimentar en diferentes campos, desde la medicina, la astronomía, la comunicación y la ciencia en general, así como en el desarrollo de Tecnología de Información (TI), esta nueva red se prepara para salir en México en noviembre próximo bajo la batuta del Consorcio Universitario para el Desarrollo de Internet 2 (CUDI).

Además de las siete universidades participantes en este proyecto (UNAM, UAM, ITESM, Universidad de las Américas, U de G, UNL, IPN) y el CONACYT, algunas empresas vinculadas a la TI ya tienen boleto para iniciar a su vez experimentos y aportar nuevos sistemas que den mayor capacidad, velocidad e interconexión.

Sin embargo, la participación de empresas privadas en un plazo de aproximadamente un año y medio en esta nueva red de redes, ha generado algunas preocupaciones por parte de firmas como Sun Microsystems que abogan por un acceso libre y rechazan la creación de sistemas cerrados como los de Microsoft e IBM.

Ante la crítica de que el gigante azul se ha “apropiado” de algunos conceptos de Internet como e-business, Alfredo Capote, director general de IBM afirma que no esta de acuerdo con esa perspectiva. “Nosotros vimos los planteamientos, establecimos la visión, cambiamos la conceptualización de Internet como un medio de ayuda al sector tecnológico. De hecho cuando nadie hablaba de las posibilidades comerciales de Internet hace cinco años, el presidente de IBM visualizó su potencial de negocios.”

En ese sentido, Alejandro Pisanti, presidente del consejo directivo del CUDI, señala que es delicado que algunos conceptos generales de Internet se conviertan en propiedad tácita y explícita de una empresa.
Al considerar delicado el tema, el también director de cómputo de la UNAM, afirma que hasta ahora no ha visto en la práctica que ninguna empresa de TI esté en proceso de utilizar conceptos generales para usos comerciales. La relación entre las empresas y las universidades involucradas en la conformación de Internet 2 ha sido de amplia colaboración.

La creación de esta red de redes, agrega, permitirá por primera vez realizar un proyecto grande entre el sector académico y el privado para probar nuevos equipos de alto almacenaje, ruteadores, etcétera.

Pero, mas allá de cuestiones comerciales, la investigación y pruebas que permitirá la segunda versión de Internet abrirá una gama diversa de posibilidades para las comunicaciones, ciencia, negocios, entretenimiento y redes, entre otras cosas.
Los siete centros académicos serán los encargados de organizar la red y proveer de servicio al grupo de socios (integrados por las propias universidades) y los afiliados (las empresas participantes), donde teléfonos de México ha donado el uso de su red de fibra óptica, es decir, la columna vertebral de los sistemas de telecomunicaciones. Nada mas ni nada menos que 33,000 kilómetros de hilos conductores.

“Más que pedir dinero a las empresas -afirma Pisanti- queremos colaboración de desarrollo, pruebas de tecnología con la participación de ingenieros de ambos sectores”.

Entre las empresas que colaboran en este proyecto de Internet 2, están Sun Microsystems, IBM, Cisco Systems Nortel, 3COM, Fore, entre otras. Pisanti señala que a diferencia del resto de la industria de TI, la participación de Hewlett-Packard no ha sido muy activa.

Para Rafael Hernández Corro, director general de México y Centroamérica de 3COM, el punto central de la naciente red de redes estará en el aprovechamiento de aplicaciones, más que en el ancho de banda o redes grandes. “El problema de poder acceder a estas aplicaciones, que serán mas robustas, es que requerirán de redes locales muy potentes, especialmente las de LAN en donde se necesitan otro tipo de aplicaciones de convergencia. Por ejemplo, en una cirugía remota no se pueden tener retardos y las respuestas deben ser inmediatas”.

Y agrega: “Cuando ya se tiene acceso a amplios anchos de banda el problema ya no es cuánto tarda en desplegarse la página, sino que realmente se haga uso de la información de manera efectiva, y aquí es donde 3COM se ha involucrado mucho”.

**INTERNET 2: Probando 1, 2, 3. Probando.**

Al oir hablar a algunas personas sobre ella, la Internet2 suena como una red desarticulada, una nueva Internet. Pero ésta no es una Internet reinventada, en su lugar es un espejo microcósmico de la arquitectura actual de Internet, diseñada para proveer un ambiente de desarrollo.

La idea de acuerdo a dos de sus mayores patrocinadores, Bay Networks y Cisco Systems, es utilizarla para probar cosas como multiemisión (multicasting). Si algo va terriblemente mal durante una prueba en una red separada que simplemente es una versión de Internet en menor escala, no afectará el espacio público que se ha vuelto Internet. También, si una nueva característica es desarrollada en la mini-

Internet, que es arquitectónicamente la misma, sería muy simple montarla sobre Internet.

Dicho lo anterior, las tres prioridades en estos días para los desarrolladores de Internet2 son aumentos (augmentation), señales de ocupado y multiemisión. Esta es la última palabra de cada uno de ellos.

- Aumentos: la idea aquí es crear un esquema para diferenciar entre el tráfico de voz y el de video colaborativo. Para este fin, los desarrolladores están utilizando cajas llamadas agentes de ancho de banda (bandwidth broker).

- Los administradores de red les hablan a través de una interfaz java y expresan la política de prioridad del tráfico. Así, por ejemplo, la voz fluyendo a través de la red es tratada con diferente prioridad que el video. La caja broker habla con los ruteadores, switches y otros equipos a través de los cuales pasarán los paquetes IP. Entonces los paquetes llegan y son marcados por el ruteador del extremo que los etiqueta como video o voz en forma diferente.

- Señales ocupadas: la red habilitada con QoS (Quality of Service; Calidad del Servicio) está peleando contra la congestión de la red, instigando un cambio en el paradigma para los desarrolladores de aplicaciones, usuarios y planificadores de red.

Existen dos formas esencialmente en las que la red responde cuando se presenta congestionamiento. Nuevas conexiones pueden ser admitidas y todos experimentan un desempeño degradado, que es, por supuesto, una característica de la Internet actual y de un sistema de servicios diferenciados basados en precedencia. En forma alternativa, las nuevas conexiones pueden ser negadas y la calidad de las conexiones existentes mantenida, que es una característica del sistema telefónico.

También se esta convirtiendo en característico de las aproximaciones QoS que ofrecen garantías de transmisión no relativas.

Para muchas aplicaciones, puede ser mejor encontrar ocasionalmente una señal de ocupado durante la preparación de una llamada que enfrentar una degradación inesperada del servicio.
Por ejemplo, los investigadores visualizando una simulación computacional en una supercomputadora utilizando una herramienta de visualización colaborativa distribuida, no pueden darse el lujo de cortar su debate técnico debido a un congestionamiento de la red inesperado. Al reservar explícitamente los recursos de un extremo al otro desde la preparación de la llamada, la arquitectura puede soportar una señal de ocupado cuando los recursos de red requeridos no puedan ser garantizados.

Multiemisión: Se ha hecho más ruido acerca de multiemisión que sobre cualquier otra prueba en Internet2.

La idea es: si se tiene un sistema que envía datos hacia fuera y lo hace una sola vez, el sistema no debería necesitar saber sus receptores específicos, sólo que muchos sistemas están recibiendo los datos.

La multiemisión es una transmisión de uno a muchos, como el radio. Los usuarios finales solamente sintonizan una estación. Con Internet, los usuarios pueden sintonizar un canal virtual, que podría ser una liga desde un sitio web.

La multiemisión es utilizada ahora en la Bolsa de Nueva York (New York Stock Exchange) para alimentación en vivo del mercado.

Los desarrolladores de Internet también están enfocados a proveer QoS para la multiemisión, porque el mayor obstáculo en Internet sigue siendo el inadecuado ancho de banda, ya que algunos lugares en Internet tienen un alto ancho de banda y otros tienen bajo, y por supuesto, como hacer para que crezca. El intercambio de acciones emite a miles de sistemas, no a millones, así que escalarlo todavía no ha sido probado.

El reto es planchar los algoritmos que manejan los árboles de multiemisión y determinar como definir su ruta y como proveer QoS para datos sensibles al tiempo. La premisa es que Internet manejará todos los servicios telefónicos, video y comercio.

Hasta ahora, los desarrolladores de Internet2 tienen una sensación general de cómo se ve la arquitectura. Pero algunos sectores,, tales como los temas de seguridad peculiares de la multiemisión, tomarán tiempo antes de cuajarse. Digamos que alguien llama cada tres segundos para hacer telemercadeo o sólo por molestar, esto es un enorme consumo de recursos si es algo que comúnmente está ocurriendo.

Esto no es nuevo, y tampoco lo es un caballo de troya que puede robar datos anónimamente. Pero nuevo o no, existen amenazas que todavía crean problemas.

**CALL (Computer Assisted Language Learning)**

La instrucción asistida por computadora (CAI) se refiere al área de estudio sobre la forma a través de la cual una computadora asiste a un usuario en el proceso de aprendizaje. En los últimos 40 años se ha visto un crecimiento exponencial en las aplicaciones de enseñanza asistida por computadora.

Durante esta rápida evolución tecnológica CAI se ha vuelto mas refinado como resultado de sus transiciones paradigmáticas del pensamiento conductista al cognitivo al constructivismo. Las computadoras ahora funcionan como medios sofisticados para la instrucción.

La inteligencia artificial simbólica (IA) ha propuesto esquemas interesantes en el área de ICAI (Intelligent CAI), pero las redes neuronales no se han utilizado más que en muy pocas ocasiones. En general, el conocimiento manejado en CAI es representado más bien de manera explícita, por lo que las redes neuronales no han encontrado aplicación.
CAI da a lugar al modo de enseñanza y aprendizaje por descubrimiento, en dónde el estudiante esta involucrado en una exploración intelectual del conocimiento de una manera mucho más libre y autodirigida. Es entonces una herramienta a través de la cual el aprendizaje puede ocurrir. En los últimos años más y más sistemas CAI han incorporado herramientas multimodales, interfaces que manejan audio, imágenes video, y programas interactivos que invocan a diversas aplicaciones.
Recientemente, la integración de interfaces de voz a los sistemas CALL o ICALL (Intelligent CALL) ha creado un mundo nuevo de posibilidades para el desarrollo de herramientas para la enseñanza.

El ser humano desde que nace necesita de guías que le muestren el camino para desarrollar sus habilidades. Estos guías, tutores o maestros con sus enseñanzas forman o crean inquietudes en las cuales se basa nuestro desarrollo cognoscitivo. Los tutores humanos proveen una alta eficiencia en el entorno de la enseñanza debido a que son capaces de estimar el grado de entendimiento de sus alumnos sobre un tema. Esto es posible gracias a la interacción directa tutor-alumno.
Con el desarrollo de la tecnología se planteó una cuestión interesante ¿Será posible duplicar un tutor humano mediante alguna máquina? Si fuera así, la única máquina capaz de emular las labores humanas en la actualidad es la computadora. Los sistemas inteligentes de enseñanza o mejor conocidos por su nombre en inglés "*Intelligent Tutoring Systems*" o solo ITS, son herramientas de software que utilizan técnicas descubiertas en la inteligencia artificial para lograr la enseñanza de algún tema o habilidad. Los sistemas inteligentes de enseñanza son comúnmente referidos como CAI que son las siglas de Instrucción Asistida por Computadora.

Los sistemas inteligentes de enseñanza son verdaderamente sistemas de comunicación de conocimiento. Pueden ser definidos de esa manera debido a que el énfasis principal en el desarrollo de estos sistemas es proveerlos con acceso a la representación del conocimiento que es lo que el sistema intenta de comunicar al estudiante. En un ITS el énfasis está puesto en el conocimiento (qué) a ser comunicado al estudiante y no en el mecanismo (cómo) de comunicación usado para presentar el conocimiento al estudiante.

**Propuestas**

A continuación se describen algunas propuestas pedagógicas de autores que han influido en el desarrollo de sistemas educativos. Aunque se encuentran agrupadas por autor, de ninguna manera se pretende describir la totalidad de su trabajo.

**Skinner**

Ante la imposibilidad de saber con certeza lo que sucede dentro del cerebro, Burrhus F. Skinner decidió tomarlo como caja negra y preocuparse únicamente por las entradas (estímulos) y salidas (respuestas).

Cuando la respuesta es correcta, se debe otorgar un refuerzo positivo. Esta teoría llevó a Skinner a diseñar una máquina de enseñar. Este artefacto mecánico plantea una pregunta al alumno, quién debe presionar el botón que corresponde a la respuesta correcta. Si acierta, la lección continúa, pero si falla el ejercicio recomienza.
"Cada información nueva suministrada por la máquina da lugar, así, a elecciones que testimonian la comprensión obtenida, con tantas repeticiones como sean necesarias y con un progreso ininterrumpido en caso de éxitos constantes. Cualquier disciplina puede, pues, programarse de acuerdo con ese principio, ya se trate de razonamiento puro o de simple memoria" [Piaget, 1967].

**Piaget**

Jean Piaget es el creador del paradigma constructivista, cuya idea principal es que el estudiante construye activamente su propio conocimiento. La mente del estudiante transforma lo que recibe del mundo externo para determinar lo que se aprende. Es decir, el aprendizaje no es la recepción pasiva de la enseñanza, sino un trabajo activo de parte del estudiante, en el que el maestro juega un papel importante apoyando, cuestionando y actuando como modelo o entrenador [Crotty, 1997].

Piaget encuentra cuatro métodos básicos de enseñanza [Piaget, 1967]:
1. Método receptivo: El profesor se encarga de dar la lección.
2. Método activo: Adquisición de conocimientos por la acción. El alumno es activo "en el sentido de un redescubrimiento personal de las verdades por conquistar, haciendo recaer esta actividad en una reflexión interior y abstracta".

3. Método intuitivo: Se suministra a los alumnos representaciones audiovisuales de actividades, sin conducir a una realización efectiva de éstas.
4. Método programado: Aprendizaje basado en estímulo–respuesta. Puede enseñar un saber eficientemente, más no un razonamiento.

**Vigotsky**

En su teoría de aprendizaje social, Vigotsky define el aprendizaje como la internalización de conocimiento que ocurre cuando un proceso extrapersonal se transforma en un proceso intrapersonal individual [Ayala, 1996].

El aprendizaje es un proceso complejo que se da en el contexto de la interacción entre medio ambiente y pensamiento. El estudiante es parte de un sistema, y su medio para interactuar consiste en los lenguajes. [Gamboa, 1997].

Vigotsky plantea el concepto de la *zona de desarrollo proximal* de un estudiante como el espacio entre el nivel de desarrollo actual y el nivel de desarrollo potencial con el apoyo de un experto. Los esfuerzos de la enseñanza deben enfocarse en esta zona [Ayala, 1996; Bruner, 1984].

**Bloom**

Para Bloom, la conducta se compone de tres dominios, el cognoscitivo, el afectivo y el psicomotor. En el dominio cognoscitivo se puede hacer una clasificación en seis niveles de actividad intelectual. Esto se conoce como taxonomía o jerarquía de Bloom [Dale, 1985; Beutelspacher, 1995]:
1. Conocimiento: Implica el proceso de memorización. El estudiante repite la información tal como se le presentó. Comprende el conocimiento de datos y hechos específicos, terminología, convenciones, secuencias, categorías, metodologías, criterios, principios, teorías, etc.
2. Comprensión: Incluye los procesos para entender e interpretar el mensaje literal de una comunicación. Comprende habilidades de traducción, explicación, extrapolación, etc.
3. Aplicación: Abarca los procesos caracterizados por la transferencia (generalización) del conocimiento a situaciones parecidas, es decir, por la habilidad para llevar a la práctica el conocimiento adquirido.
4. Análisis: Se refiere a los procesos en que la información recibida se fracciona en sus elementos constitutivos, de modo que se expresen explícitamente la organización, las relaciones y las jerarquías entre las ideas.
5. Síntesis: Comprende los procesos en que se fusionan varios elementos de tal manera que constituyan un todo que antes no estaba presente, como en el desarrollo de una comunicación original, en la planeación y en la deducción de relaciones abstractas.
6. Evaluación: Los procesos que implican juzgar el grado de satisfacción de criterios específicos, a partir de evidencia tanto interna como externa.

**Dubinsky**

Desarrolló una metodología para enseñar las matemáticas, fundamentada en las teorías de Piaget, basada en el principio de tener vivencias para tener aprendizaje. Las vivencias pueden ser tanto físicas como intelectuales. Su ciclo metodológico busca estimular la abstracción reflexiva, la herramienta intelectual que impulsa al proceso mental, a través de actividades, clase teórica y ejercicios (por lo que se denomina ACE).
Dubinsky utilizó un lenguaje de programación matemático (ISETL) para generar un complejo ambiente de intercambio entre conocimiento, profesor, alumnos, problemas y computadoras, que fomente la abstracción reflexiva [Gamboa, 1997].

**Bruner**

Jerome Bruner creó un modelo de la mente humana en el que hay tres distintas mentalidades que compiten por el control:
1. Enactiva: Hacer, manipular.
2. Icónica: Reconocer, visualizar, comparar, configurar, concretar.
3. Simbólica: Abstraer, cadenas de razonamiento.

El aprendizaje debe comenzar en lo concreto y ser llevado a lo abstracto. Alan Kay lo resume en el lema “al hacer con imágenes, se generan símbolos” [Kay, 1990].
Bruner define un paradigma de la transmisión de conocimientos, desde el punto de vista del instructor, usando situaciones estereotipadas que llama *formatos* [Bruner, 1984]:
1. Realizar la tarea a enseñar, como ejemplo.
2. Inducir a que el alumno intente lo mismo.
3. Reducir la complejidad del problema, segmentándolo y ayudando.
4. Dominada la tarea, animar a iniciar otra de orden superior.
5. Sólo cuando la tarea es dominada termina la instrucción, es decir, la incorporación del conocimiento adquirido al conocimiento verbalizado.
6. Ahora es posible el “discurso” entre maestro y discípulo: hay conocimiento compartido.

**Papera**

Partiendo de las ideas de Piaget, Seymour Papert llega a la conclusión de que es más importante ayudar a los niños a aprender cómo desarrollar y depurar sus teorías, que enseñarles las teorías que nosotros consideramos correctas. Con este fin, Papert supervisó la creación del lenguaje de programación LOGO (2.5.1.3), que aprovecha los requerimientos epistemológicos de las representaciones computacionales (programas) para ayudar a los estudiantes formular conceptos [Wenger, 1987].

**Anderson**

Para John Anderson, cualquier conocimiento se puede representar bajo la forma de reglas de producción [Anderson, 1995]. En este sentido, propone la teoría ACT de adquisición de habilidades, basada en:

1. Diferenciar: El primer paso es distinguir el conocimiento declarativo (datos, hechos) del conocimiento procedural (aplicar los datos conocidos). El conocimiento declarativo es aprendido de la observación e instrucción; la habilidad cognoscitiva radica en convertirlo a conocimiento procedural.

2. Compilar: El conocimiento procedural se adquiere únicamente al utilizar conocimiento declarativo en un contexto de resolución de problemas, en un proceso denominado “compilación de conocimiento”.

3. Practicar: Ambos tipos de conocimiento se fortalecen con la práctica. Después de adquirir un conocimiento, practicarlo genera un desempeño más fluido, rápido y seguro.

**Gagné y Briggs**

Gagné y Briggs proponen nueve eventos de enseñanza como los componentes de una lección efectiva [Gagné, 1974]:

1. Llamar la atención y motivar.
2. Presentar el objetivo.
3. Recordar los prerrequisitos.
4. Mostrar el estímulo.
5. Guiar el aprendizaje.
6. Evaluar el desempeño.
7. Dar retroalimentación.
8. Medir el desempeño.
9. Fomentar el aprendizaje.

**Computadoras y Educación**

La tecnología puede tener un impacto gigantesco en la educación. Un ejemplo muy claro es la imprenta. A partir 1436, fecha de su invención por Gutemberg, se pudo generalizar el uso de los libros como herramientas educativas. La computadora puede tener un impacto aún más trascendente en la educación. James Lockard propone una clasificación según el papel que juega la computadora en el proceso de enseñanza-aprendizaje.

**La Computadora como Herramienta**

La computadora es una herramienta que ayuda a maestros y alumnos en tareas sencillas. El uso de aplicaciones típicas como procesadores de texto, hojas de cálculo y bases de datos para aprender (por ejemplo, a redactar) es una forma de usar a la computadora como herramienta. Otra forma sería que el maestro utilice programas especializados para la producción de material didáctico (por ejemplo, crucigramas) o para obtener información estadística del grupo.

Con la difusión de Internet, se multiplican los usos de la computadora como herramienta; por ejemplo, para buscar de información en el *World Wide Web* y para comunicarse a través del correo electrónico.

**La Computadora como Maestro**

La educación asistida por computadora (EAC) es un “ambiente de aprendizaje caracterizado por la interacción educativa entre la computadora y el estudiante” [Wright, 1985]. Lockard define distintos tipos de sistemas de EAC:

1. Ejercicio y Práctica: Sirve para reforzar conocimiento adquirido anteriormente. Se presenta un estímulo, el estudiante responde y recibe retroalimentación inmediata. Este tipo de EAC es heredero directamente de las máquinas de enseñar de Skinner. Pueden o no adaptarse al desempeño del alumno.

2. Tutorial: Sirve para presentar material nuevo al alumno. Son “programas de computadora que enseñan manteniendo un diálogo con el estudiante. Presentan información, hacen preguntas y deciden mostrar nueva información o revisar la actual basándose en el nivel de comprensión del estudiante” [Alessi, 1985]. Lockard distingue entre los tutóriales lineales, que siempre presentan la misma información en el mismo orden, y los multilínea les (*branching*), donde el estudiante o la computadora eligen un camino individual.

3.-Simulación: El estudiante interactúa con una analogía de la realidad. Debe tomar decisiones basándose en la situación presentada, provocando que esta se transforme, simulando la realidad, en una nueva situación sobre la que hay que volver a decidir. Se pueden distinguir cuatro categorías de simulaciones [Alessi, 1985], las físicas (aviones, instrumentos, laboratorio de física), las procedurales (diagnóstico), la situacionales (descubrir reglas jugando escenarios), y de proceso (el estudiante sólo define la situación original).

4. Juego Educativo: Sirve para enseñar o reforzar algún objetivo educativo predeterminado. El juego y el material didáctico son inseparables. La intención no es reflejar la realidad, como la simulación, sino presentar un reto divertido.

5. Resolución de problemas: Sirve para mejorar las habilidades del estudiante para resolver problemas que, en general, no se relacionan con alguna materia en particular.

*A esta clasificación se puede agregar una sexta categoría:*
6. Libros Electrónicos: Sistemas cuya única función es presentar, de una o más formas, información didáctica solicitada por el estudiante.

**Primeros Sistemas**

Los primeros sistemas de EAC se concentraron en mostrar la viabilidad de sistemas masivos de educación a través de terminales. La única excepción es LOGO, el lenguaje de programación que inventó un nuevo papel de la computadora en la educación.

**PLATO (1959)**

Donald Bitier dirige el primer proyecto en gran escala para el uso de la computadora en la educación: PLATO (*Programmed Logic for Automatic Teaching Operation*). El sistema creció rápidamente de una a miles de terminales conectadas a un *mainframe*. El objetivo alcanzado fue demostrar la viabilidad técnica, económica y administrativa de una red educativa de gran tamaño basada en computadoras [Molnar, 199; Lockard, 1987].

**Stanford Drill and Practice System (1963)**

Patrick Suppes y Richard Atkinson establecen un programa de investigación y desarrollo de la EAC para matemáticas y lectura. Se intentaba liberar al alumno de la rigidez de la instrucción de grupo, con especial preocupación para los estudiantes discapacitados. El sistema se usaba para ejercitar individual e intensivamente al estudiante [Molnar, 1997; Lockard, 1987].

**LOGO (1967)**

Seymour Papert (2.1.1.7) se propuso desarrollar una nueva forma de ver el uso de la computadora para la educación. Inventó el lenguaje LOGO para fomentar el razonamiento y la lógica. Quiso hacerlo accesible para niños, por lo que debía ser fácil definir procedimientos para tareas sencillas, no numéricas, que fueran difíciles de implementar con lenguajes tradicionales.

Papert usaba LOGO para educar en una gran variedad de “micro mundos”, como música, física, poesía y matemáticas. El objetivo era enseñar a pensar. Por ejemplo, no se pretendía instruir matemáticas, sino enseñar a ser matemático. Pronto, este lenguaje se convirtió en *el* lenguaje del movimiento de la computadora en la primaria.

Actualmente se utiliza LOGO para programar robots hechos con piezas de LEGO; así, el estudiante aprende a definir un problema y definir los pasos necesarios para resolverlo [Lockard, 1987; Delval, 1986].

**TICCIT (1971)**

El proyecto TICCIT (*Time–shared Interactive Computer Controlled Information Televisión*) pretendía demostrar que la EAC daba una mejor y más barata instrucción para adultos, en los campos de inglés y matemáticas. Se basaba en la instrucción controlada por el alumno [Lockard, 1987].

## 5.6 Programación (Pacos, Dedales; Nipq, Libra)

## SISTEMAS EXPERTOS EN CONTABILIDAD

Los sistemas expertos se han venido aplicando con éxito en múltiples campos: medicina, geología, química, ingeniería, etc. (Waterman, 1986; Harmon y King, 1988; Rauch-Hindin, 1989), para realizar tareas muy diversas (e.g., interpretación, predicción, diagnóstico, diseño, planificación, instrucción, control, etc.) (Hayes- Roth en al., 1983, pp. 13-16; Waterman, 1986, p. 33).

Las actividades administrativas, financieras y contables también son campos en los que se pueden aplicar los sistemas expertos, pues se realizan muchas de las tareas antes descritas y, además, éstas cumplen la mayoría de los requisitos que son necesarios para poder desarrollar un sistema experto (i.e., las tareas requieren conocimiento especializado, existen auténticos expertos en la materia, los expertos son escasos, la pericia necesita ser localizada en distintos lugares, la mayoría de las tareas requieren soluciones heurísticas, ...)

(Waterman, 1986, pp. 127-134). Ahora bien, no en todas las tareas que se realizan en el campo de la contabilidad y las finanzas es necesario utilizar los sistemas expertos.

Así, en las tareas de auditoría que están perfectamente estructuradas, son muy mecánicas y pueden expresarse en forma algorítmica (e.g., preparación de balances, cálculo de ratios, muestreo, circularización,...) se puede, y es conveniente, utilizar la informática convencional (e.g., programas informáticos normales, tratamientos de textos, bases de datos, ...);

Como podemos observar, todo el proceso se puede representar y por ende automatizar, continuando con el ejemplo: en las tareas que estén semiestructuradas se pueden utilizar los sistemas de ayuda a la decisión (e.g., hojas de cálculo, sistemas de consulta de archivos, sistemas de representación y análisis de datos, ...); reservándose los sistemas expertos para las tareas que estén muy poco o nada estructuradas, pues en este tipo de tareas se requiere mucho del juicio de un experto y se utilizan reglas heurísticas para llegar rápidamente a una solución, dado que el campo de soluciones puede ser muy amplio (Sánchez, 1993).

En principio, los sistemas expertos se pueden aplicar en todas las áreas de la contabilidad (Montesinos, 1976; Cañibano, 1982). Ahora bien, como esta clasificación resultaría muy amplia y, además, es poco práctica, vamos a clasificar las aplicaciones potenciales de los sistemas expertos en contabilidad de acuerdo con las siguientes áreas (Anderson y Bernard, 1986; Connell, 1987; Elliott y Kielich, 1985; Englard y Kiss, 1989; O'leary 1987a; Shim y Rice, 1988):

* Auditoria: Análisis de la materialidad y del riesgo, evaluación del control interno, planificación de la auditoria, evaluación de la evidencia, análisis de cuentas

concretas, formación de opinión, emisión del informe, auditoria interna, auditoria informática, etc.

* Contabilidad de costes y de gestión: Cálculo y asignación de costes, asignación de recursos escasos, control y análisis de desviaciones, planificación y control de gestión, diseño de sistemas de información de gestión, etc.

* Contabilidad financiera: regulación legal, normas y principios contables, recuperación y revisión analítica de registros contables, diseño de sistemas contables, imputación contable, consolidación de estados contables, etc.

* Análisis de estados financieros: Análisis patrimonial, financiero y económico de los estados contables, salud financiera de la empresa, cálculo e interpretación de ratios, cálculo y análisis de tendencias, etc.

* Planificación financiera e industria de los servicios financieros: Planificación financiera corporativa, planificación financiera personal, análisis de inversiones, gestión de tesorería, mercado de valores, seguros, banca, concesiones de crédito, etc.

A continuación, pasamos a analizar brevemente la aplicación de la tecnología de los sistemas expertos en cada una de estas ramas de la contabilidad.

**Sistemas Expertos en Costes**

Al igual que ha ocurrido en auditoría, los cambios tecnológicos actuales (i.e., perfeccionamiento del tratamiento y comunicación de la información, aumento de la importancia de la información como fuente para la toma de decisiones, nuevas técnicas de gestión, nuevas tecnologías de la producción, ...) también han afectado a la contabilidad de costes y a la contabilidad de gestión.

Como consecuencia de estos cambios, en la década de los ochenta empezaron a surgir voces críticas que ponían en tela de juicio la información que suministraba la contabilidad de costes tradicional, pues ésta no suministraba información relevante y oportuna para poder tomar decisiones en el entorno actual (Johnson y Kaplan, 1987).

Sin embargo, estas críticas han servido para producir una revitalización del papel de la contabilidad de costes y de gestión, considerándose ahora una de las principales fuentes de información para las empresas.

De acuerdo con la Asociación Español de Contabilidad y Administración de Empresas (AECA), la contabilidad de costes es la que "suministra la información analítica relativa a los costes de los productos y servicios generados por la empresa, asumiendo el nivel de desagregación que se considere oportuno en cada situación, a efectos de determinar el valor de las existencias, y el coste de los productos

vendidos, para poder transmitir esta información a los administradores o gerentes a fin de que se pueda proceder a la confección de los estados contables"

(AECA, 1990, pp. 21-22); mientras que la contabilidad de gestión "es una rama de la contabilidad, que tiene por objeto la captación, medición y valoración de la circulación interna, así como su racionalización y control, con el fin de suministrar a la organización la información relevante para la toma de decisiones empresariales" (AECA, 1990, p. 23).

La contabilidad de costes persigue principalmente los siguientes objetivos básicos:

(1) calcular los costes de las funciones, actividades, secciones, etc. del proceso productivo de la empresa,

(2) calcular el coste de los productos y servicios generados por la empresa,

(3) calcular el coste de los productos vendidos y analizar los resultados analíticos,

(4) valorar los activos derivados del proceso productivo, con el fin de confeccionar los estados contables anuales.

Mientras que la contabilidad de gestión, además de recoger los objetivos anteriormente citados, se centra principalmente en el proceso de racionalización, planificación y control de los costes, con el fin de suministrar información relevante y oportuna para apoyar la toma de decisiones empresariales.

La contabilidad de gestión es, pues, una evolución lógica de la contabilidad de costes, como consecuencia de las mayores demandas de información de las empresas modernas para poder tomar decisiones.

Por razones obvias de extensión, en el presente trabajo estudiaremos los sistemas expertos en contabilidad de costes y en contabilidad de gestión como un solo subconjunto de la contabilidad.

Los subdominios o campos potenciales de la contabilidad de costes y de gestión en los que se pueden aplicar los sistemas expertos son muy amplios y variados, destacando como más significativos los siguientes:

- asignación de recursos escasos,
- gestión de la función de compras y de aprovisionamiento,
- gestión de inventarios,
- gestión de recursos humanos;
- calculo del coste de las órdenes de trabajo,
- planificación y control de la gestión,
- costes estándares y análisis de desviaciones,

- control presupuestario,
- análisis de productos y combinaciones de ventas,
- presentación de informes económicos
- estadísticas de costes y resultados,
- control de nuevas tecnologías de la producción,
- diseño de sistemas de información de contabilidad de gestión,
- ayuda para la determinación de los costes pertinentes para la toma de decisiones,
- justificación de las inversiones en nuevas tecnologías. etc.

## SISTEMAS EXPERTOS EN CONTABILIDAD FINANCIERA

De acuerdo con el profesor Cañibano, la contabilidad financiera "tiene por objeto la preparación de los estados contables que informan sobre la renta y la riqueza de la empresa, para lo cual somete a tratamiento y análisis a cuantas transacciones se producen entre la citada unidad económica y el mundo exterior a ella" (Cañibano, 1982, p. 28).

La contabilidad financiera tiene como objetivo principal la preparación de los estados financieros de la empresa, con el fin de suministrar la información que necesitan los distintos usuarios de la empresa sobre el resultado y el patrimonio de la empresa.

Una característica muy importante de la contabilidad financiera es la estrecha vinculación que tiene con la legislación mercantil y contable, ya que está regulada por una serie de normas y principios de contabilidad generalmente aceptados, emitidos por el estado y por entidades de emisión de normas de contabilidad de reconocido prestigio (e.g., la SEC, el FASB, la AECA), que garantizan la homogeneidad y fiabilidad de la información contable frente a terceros.

## SISTEMAS EXPERTOS EN ANÁLISIS CONTABLE

De acuerdo con el profesor Urías, "el Análisis de Estados Financieros, utilizando ciertas técnicas, trata de investigar y enjuiciar, a través de la información contable, cuáles han sido las causas y los efectos de la gestión de la empresa para llegar a su actual situación y, así, predecir, dentro de ciertos límites ..., cuál será su desarrollo en el futuro, para tomar decisiones consecuentes". (Urías, 1992, p. 179).

El análisis de estados financieros se puede dividir en tres fases:

(1) exámen,
(2) análisis e interpretación,
(3) prescripción.

En la primera fase, se realiza un examen o revisión de los documentos contables (i.e., balance, cuenta de pérdidas y ganancias, memoria, estado de origen y aplicación de fondos, ...), utilizando para ello una serie de técnicas o procedimientos específicos, tales como: comparaciones de masas patrimoniales, porcentajes, números índices, ratios, tendencias, etc.

A continuación, en la segunda fase, se analiza e interpreta la información obtenida en la etapa anterior, y se realiza un diagnóstico de la situación actual de la empresa y de la gestión realizada.

Y, por último, en la tercera fase, se realizan unas predicciones y se indican los remedios o soluciones que se pueden aplicar con el fin de mejorar la situación futura. Además, todos estudios y análisis se suelen realizar desde tres ángulos diferentes: el análisis patrimonial, el análisis financiero, y el análisis económico. (Alvarez, 1990; Rivero Romero, 1990; Rivero Torre, 1991; Urías, 1992).

## SISTEMAS EXPERTOS EN PLANIFICACIÓN

El entorno financiero actual es muy complejo y ha sufrido importantes cambios en los últimos tiempos (e.g., inflación, integración de la economía a nivel mundial, liberalización de la economía y de los mercados financieros, gran variedad de productos financieros, aumento de la presión fiscal y cambio continuo de su legislación, ...).

Debido a esta complejidad, las empresas y los individuos demandan, cada vez más, servicios de asesoramiento financiero que les permitan llevar una adecuada política de inversiones y de planificación financiera.

Dentro de la Economía Financiera de la Empresa, hay una rama que se encarga específicamente de los problemas de inversión y financiación empresarial. Esta disciplina, conocida en el área anglosajona con el nombre de Financial Management, se suele denominar en España con el nombre de Dirección (Administración o Gestión) Financiera.

A su vez, dentro de esta última disciplina, existen otras ramas importantes, que se refieren a aspectos concretos, como por ejemplo: el Análisis de Inversiones, o Presupuesto de Capital (Capital Budgeting) es su acepción más moderna, que trata en general todos los problemas relacionados con las inversiones empresariales; el Análisis Financiero, que trata de analizar los problemas de carácter financiero producidos como consecuencia de dichas inversiones, y la Planificación Financiera, que estudia la planificación a corto, medio y largo plazo de la financiación de las inversiones realizadas (Suárez, 1991).

En el presente trabajo nos vamos a centrar únicamente en el área de la Planificación Financiera, pues la mayoría de los sistemas expertos se centran especialmente en resolver los problemas de esta disciplina, aunque también citaremos algún sistema relacionado con el Análisis Financiero y la Análisis de Inversiones, campos éstos que están estrechamente relacionados con la Planificación Financiera.

La planificación financiera empresarial es aquella parte de la planificación empresarial que trata de los aspectos financieros de la empresa, aunque sin olvidar sus aspectos no financieros (Suárez, 1991, p. 608).

Por otra parte, por analogía con la empresa, la planificación financiera también puede realizarse a nivel personal, dando lugar a la planificación financiera personal, que abarca, entre otras, las siguientes materias: planificación de inversiones, planificación fiscal, gestión de patrimonios, planificación de seguros, planificación de la jubilación, gestión de tesorería, gestión de deudas, etc.

La planificación financiera, el análisis de financiero, y el análisis de inversiones requieren una metodología muy específica.

Entre las materias, métodos o procedimientos que se utilizan en estas disciplinas hay que destacar, por ejemplo: los flujos netos de caja, las tasas de rendimiento interno, el plazo de recuperación de la inversión, la deflación de los valores actuales, los árboles de decisión, el análisis bayesiano, los modelos de simulación, la programación lineal y la programación dinámica, el análisis de sensibilidad, la valoración de activos, el coste del capital, el riesgo económico y financiero, análisis de balances y cálculo de ratios, presupuestos, etc. (Suárez, 1991; Fernández, 1991).

## 5.7 Conocimiento. (Redes, Quillian, Ham, Act, Memod)

***CONOCIMIENTO***

Puede ser definido como el conjunto de hechos y principios acumulados por la humanidad, o el acto, hecho o estado de conocer. Es la familiaridad con el lenguaje, conceptos, procedimientos, reglas, ideas, abstracciones, lugares, costumbres y asociaciones, unida a la habilidad de utilizar estas nociones en forma efectiva para modelar diferentes aspectos del universo que nos rodea.

Los conceptos de conocimiento e inteligencia están íntimamente ligados. La inteligencia requiere de la posesión y acceso al conocimiento. Conocimiento no debe ser confundido con datos o información. El conocimiento incluye y requiere del uso de datos e información. Además, combina relaciones, dependencias, y la noción del saber con datos e información.

A veces es también útil o más aun necesario distinguir entre conocimiento y otros términos como *creencia* e *hipótesis*.

Esencialmente, se define como creencia a toda expresión que tiene significado, que es coherente y puede ser representada. En consecuencia una creencia puede ser verdadera o falsa. Hipótesis se define como una creencia justificada que no se conoce sea verdadera. Una hipótesis es una creencia que está respaldada por cierta evidencia, pero todavía puede resultar falsa. Finalmente, se puede también decir que conocimiento es una creencia justificada como verdadera.

Otros términos relacionados con el conocimiento y que son ocasionalmente empleados son *epistemología* y *metaconocimiento*. Epistemología es el estudio de la naturaleza del conocimiento, mientras que metaconocimiento es conocimiento acerca del conocimiento, esto es, conocimiento acerca de lo que se conoce.

El conocimiento puede ser de tipo *procedimental*, *declarativo* o *heurístico*. Conocimiento procedimental es aquel conocimiento compilado que se refiere a la forma de realizar una cierta tarea (*el saber como hacerlo*). Por ejemplo, los pasos necesarios para resolver una ecuación algebraica son expresados como conocimiento procedimental.

Por otro lado, el conocimiento declarativo es conocimiento pasivo, expresado como sentencias acerca de los hechos del mundo que nos rodea (*el saber que hacer*). La información personal en una base de datos es un típico ejemplo de conocimiento declarativo. Tales tipos de datos son piezas explícitas de conocimiento independiente.

El conocimiento heurístico es un tipo especial de conocimiento usado por los humanos para resolver problemas complejos. El adjetivo *heurístico* significa *medio para descubrir*. Está relacionado con la palabra griega *heuriskein* que significa descubrir, encontrar. Se entiende por *heurístico* a un criterio, estrategia, método o truco utilizado para simplificar la solución de problemas. El conocimiento heurístico usualmente se lo adquiere a través de mucha experiencia

***REPRESENTACIÓN DEL CONOCIMIENTO***

Dado que el conocimiento es importante y primordial para el comportamiento inteligente, su representación constituye una de las máximas prioridades de la investigación en IA. El conocimiento puede ser representado como imágenes mentales en nuestros pensamientos, como palabras habladas o escritas en algún lenguaje, en forma gráfica o en imágenes, como cadenas de caracteres o colecciones de señales eléctricas o magnéticas dentro de un computador.

En nuestro estudio de IA, consideraremos las representaciones escritas y sus correspondientes estructuras de datos utilizadas para su almacenamiento en un computador. La forma de representación que se escoja dependerá del tipo de problema a ser resuelto y de los métodos de inferencia disponibles.

Una representación del conocimiento puede ser un esquema o dispositivo utilizado para capturar los elementos esenciales del dominio de un problema. Una representación manipulable es aquella que facilita la computación.

En representaciones manipulables, la información es accesible a otras entidades que usan la representación como parte de una computación.

Debido a la variedad de formas que el conocimiento puede asumir, los problemas involucrados en el desarrollo de una representación del conocimiento son complejos, interrelacionados y dependientes del objetivo. El términos generales, se debe tratar que el conocimiento esté representado de tal forma que:

- Capture generalizaciones.
- Pueda ser comprendido por todas las personas que vayan a proporcionarlo y procesarlo.
- Pueda ser fácilmente modificado.
- Pueda ser utilizado en diversas situaciones aún cuando no sea totalmente exacto o completo.
- Pueda ser utilizado para reducir el rango de posibilidades que usualmente debería considerarse para buscar soluciones.

El conocimiento declarativo puede ser representado con modelos relacionales y esquemas basados en lógica. Los modelos relacionales pueden representar el conocimiento en forma de árboles, grafos o redes semánticas. Los esquemas de representación lógica incluyen el uso de lógica proposicional y lógica de predicados.

Los modelos procedimentales y sus esquemas de representación almacenan conocimiento en la forma de cómo hacer las cosas.

Pueden estar caracterizados por gramáticas formales, usualmente implantadas por sistemas o lenguajes procedimentales y sistemas basados en reglas (sistemas de producción).

Las representaciones declarativas son usualmente más expansivas y costosas, en el sentido que la enumeración puede ser redundante e ineficiente. Sin embargo, la modificación de las representaciones declarativas es usualmente muy fácil; simplemente se agrega o se elimina conocimiento.

Las representaciones procedimentales, en cambio, pueden ser más compactas, sacrificando flexibilidad. Representaciones prácticas pueden incluir elementos tanto declarativos (listado de hechos conocidos), como procedimentales (un conjunto de reglas para manipular los hechos).

***MANIPULACIÓN DEL CONOCIMIENTO***

Existen tres paradigmas que los investigadores han utilizado tradicionalmente para la resolución de problemas de IA:

Programación Heurística.- Está basado en el modelo de comportamiento humano y su estilo para resolver problemas complejos. Existen diversos tipos de programas que incluyen algoritmos heurísticos. Varios de ellos son capaces de aprender de su experiencia.

Redes Neuronales Artificiales.- Es una representación abstraída del modelo neuronal del cerebro humano. Las redes están formadas por un gran número de elementos simples y por sus interconexiones. Una red neuronal artificial puede ser simulada o ser real. Al elemento procesador de la red, se lo denomina neurona artificial.

Evolución Artificial.- Su modelo está basado en el proceso genético de evolución natural, propuesto por Charles Darwin. Se utilizan sistemas simulados en computador que evolucionan mediante operaciones de reproducción, mutación y cruce (Algoritmos Genéticos).

Cada paradigma comprende una colección de métodos, configuraciones y técnicas desarrolladas para manipular el conocimiento. En general, una técnica de IA está caracterizada por incluir los siguientes componentes:

Procesos de Búsqueda.- Proporciona una forma de resolver problemas para los cuales no hay un método más directo, así como también se constituye en un marco de trabajo dentro del cual cualquier técnica directa puede ser incorporada.

Uso del Conocimiento.- Proporciona una forma de resolver problemas explotando las estructuras de los objetos involucrados.

Abstracción.- Proporciona una forma de separar rasgos importantes y variaciones, de los tantos que no tienen importancia.

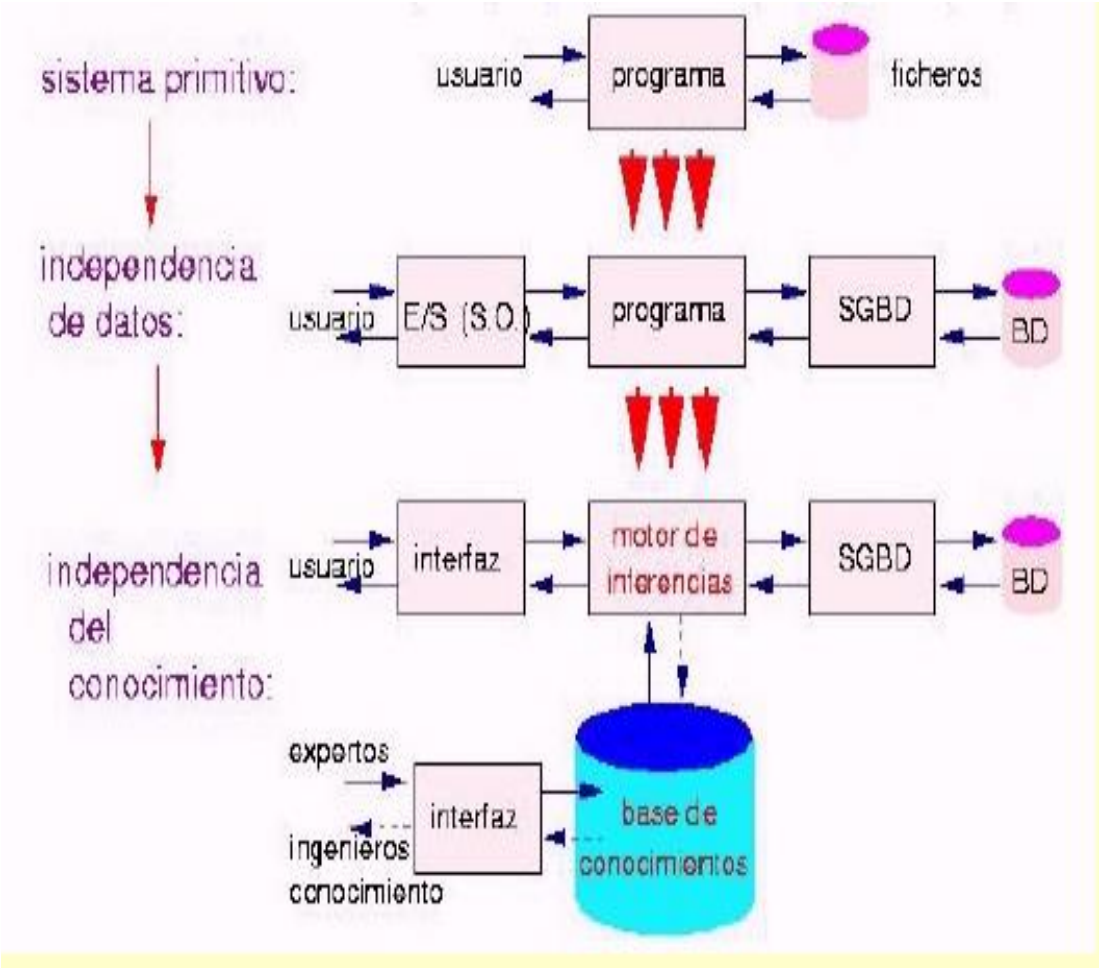
La manipulación del conocimiento involucra además la selección de objetos, entidades y rasgos que son claves para las representaciones. Al nivel más bajo de representación del conocimiento, estas entidades pueden consistir de simples datos de entrada, por ejemplo: grabaciones cuantizadas y digitalizadas de audio, datos de una imagen, valores captados por un sensor, información de interruptores o hechos elementales. Estos bloques constructivos de la representación se los denomina comúnmente *primitivas*.

La representación escogida para las primitivas, puede determinar las capacidades del sistema, el éxito, la corrección y sus posibilidades de expansión.

En lo sucesivo, este tutorial se centrará en el estudio de las aplicaciones de la programación heurística, para el desarrollo de sistemas de IA.

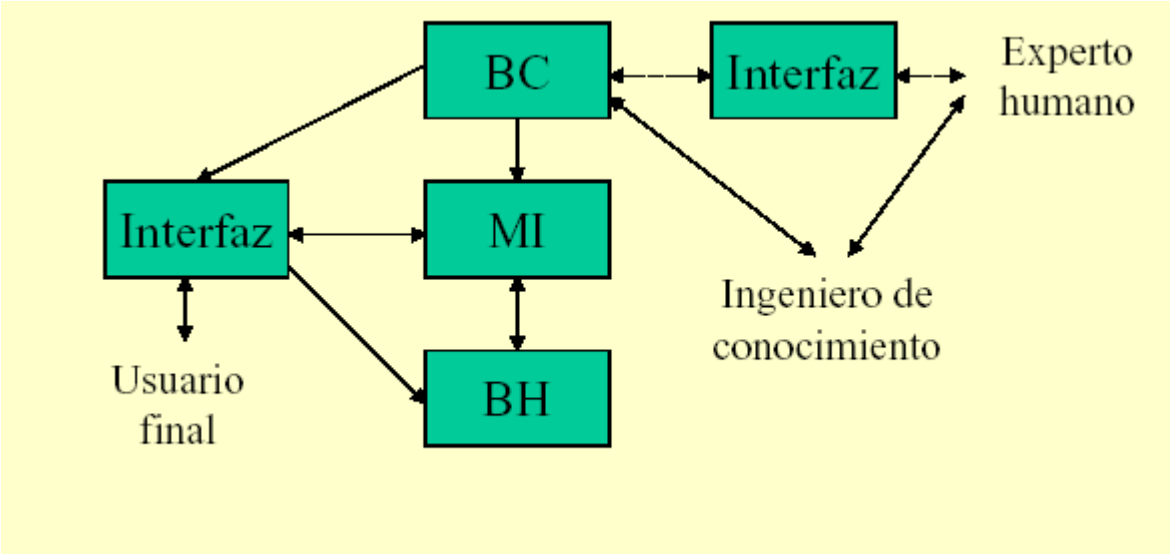
Sistema basado en conocimiento:

Evolución de modelos estructurales de sistemas software




Sistema basado en conocimiento:

Arquitectura básica:




**TIPOS DE REDES**

**Redes Asociativas**

Las *redes semánticas*, fueron originalmente desarrolladas para representar el significado o semántica de oraciones en inglés, en términos de objetos y relaciones. Actualmente, el término *redes asociativas* es más ampliamente utilizado para describirlas ya que no sólo se las usa para representar relaciones semánticas, sino también para representar asociaciones físicas o causales entre varios conceptos u objetos.

Las redes asociativas se caracterizan por representar el conocimiento en forma gráfica. Agrupan una porción de conocimiento en dos partes: objetos y relaciones entre objetos. Los objetos se denominan también nodos (elementos del conocimiento) y las relaciones entre nodos se denominan enlaces o arcos. Cada nodo y cada enlace en una red semántica, deben estar asociados con objetos descriptivos.

Son muy apropiadas para representar conocimiento de naturaleza jerárquica. Su concepción se basa en la asociación de conocimientos que realiza la memoria humana. Las principales aplicaciones son: comprensión de lenguaje natural, bases de datos deductivas, visión por computadora, sistemas de aprendizaje.

***Ventajas de las Redes Asociativas***

Las redes asociativas tienen dos ventajas sobre los sistemas basados en reglas y sobre los basados en lógica:

1. Permiten la declaración de importantes asociaciones, en forma explícita y sucinta.
2. Debido a que los nodos relacionados están directamente conectados, y no se expresan las relaciones en una gran base de datos, el tiempo que toma el proceso de búsqueda por hechos particulares puede ser significativamente reducido.

***Desventajas de las Redes Asociativas***

Entre las desventajas de las redes asociativas, se pueden mencionar:

1. No existe una interpretación normalizada para el conocimiento expresado por la red. La interpretación de la red depende exclusivamente de los programas que manipulan la misma.
2. La dificultad de interpretación a menudo puede derivar en inferencias inválidas del conocimiento contenido en la red.
3. La exploración de una red asociativa puede derivar en una explosión combinatoria del número de relaciones que deben ser examinadas para comprobar una relación.

**Representación mediante Plantillas**

Una plantilla (*frame*) es una estructura de datos apropiada para representar una situación estereotípica. Las plantillas organizan el conocimiento en objetos y eventos que resultan apropiados para situaciones específicas. Evidencia psicológica sugiere que la gente utiliza grandes plantillas para codificar el conocimiento de experiencias pasadas, o conocimiento acerca de cosas que se encuentran comúnmente, para analizar y explicar una situación nueva en su cotidiana actividad cognoscitiva.

Una *plantilla* representa un objeto o situación describiendo la colección de atributos que posee. Están formadas por un nombre y por una serie de campos de información o *ranuras* (*slots*). Cada *ranura* puede contener uno o más *enlaces* (facets). Cada *enlace* tiene un valor asociado. Varios *enlaces* pueden ser definidos para cada *ranura*, por ejemplo:

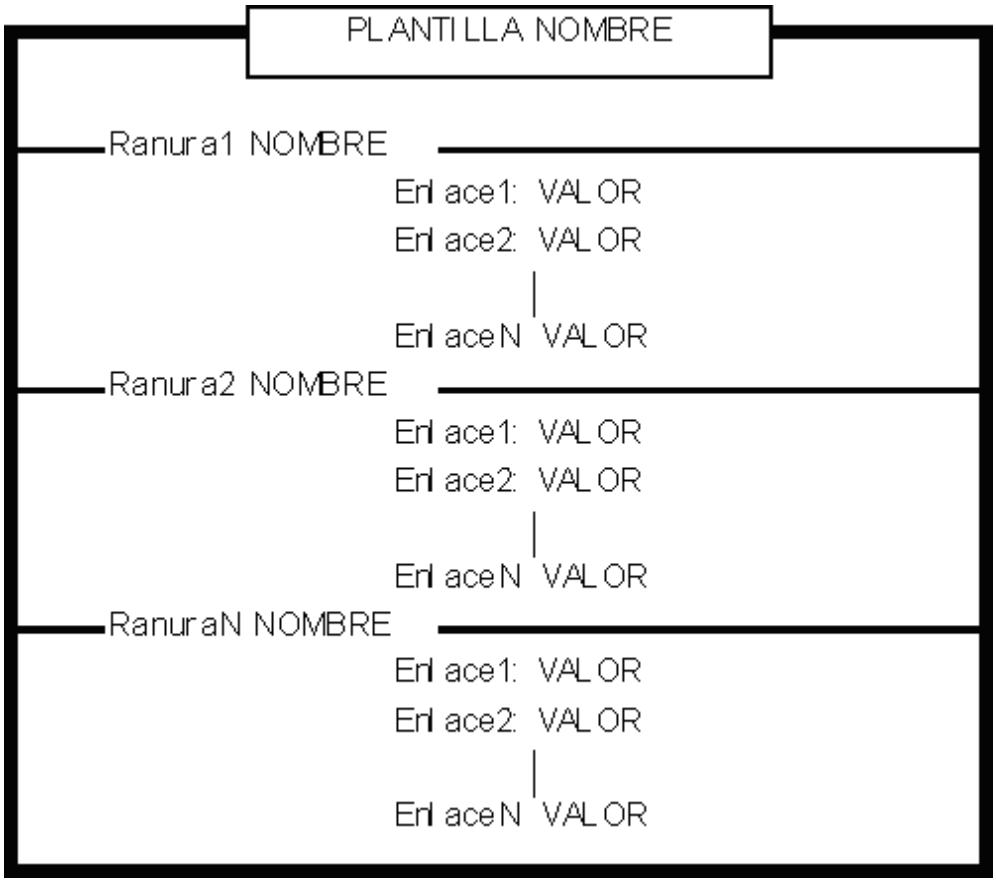
- *Rango*.- El conjunto de posibles valores para la ranura.
- *Valor*.- El valor de la ranura.
- *Default*.- El valor a ser asumido si no se especifica alguno.

Además los *enlaces* pueden ser procedimientos que residen en la base de datos y están aguardando para ser utilizados cuando se los necesite. Entre los más comunes se pueden mencionar:

- *Si-Necesitado*.- Procedimiento(s) para determinar el valor actual de una ranura.
- *Si-Agregado*.- Procedimiento(s) a ejecutarse cuando un valor es especificado para una ranura.
- *Si-Modificado*.- Procedimiento(s) a ejecutarse si el valor de una ranura es cambiado.

A estos procedimientos también se los denomina *demons* y representan un concepto poderoso en las plantillas, esto es, la habilidad de combinar conocimiento procedimental dentro de la estructura de conocimiento declarativo de la plantilla. Esto sugiere que una plantilla puede ser un medio poderoso de representación del conocimiento, especialmente si se la incorpora en una red de plantillas.

En la siguiente figura se muestra una representación abstracta de una plantilla.




Se pueden establecer ciertas similitudes entre un sistema basado en plantillas y un sistema de bases de datos. Aparentemente los dos representan "datos" (a través de las ranuras de una plantilla y de los campos de una tabla de datos), sin embargo las plantillas representan en realidad conocimiento, mientras que las bases de datos representan sólo datos.

La investigación que se realiza actualmente en bases de datos está examinando la posibilidad de aplicarlas a la representación del conocimiento, incorporando herencia y *demons* (*Bases de Datos Inteligentes*), similar a lo que se tiene en sistemas basados en conocimiento.

***Ventajas de las Plantillas***

Las ventajas que se pueden establecer para los sistemas basados en plantillas son las siguientes:

1. Facilidad de proceso guiado por las expectativas. Un sistema basado en plantillas, mediante los *demons* es capaz de especificar acciones que deben tener lugar cuando ciertas condiciones se han cumplido durante el procesamiento de la información.
2. El conocimiento que posee un sistema basado en plantillas es significativamente más estructurado y organizado que el conocimiento dentro de una red asociativa.
3. Las plantillas pueden ser estructuradas de tal forma que sean capaces de determinar su propia aplicabilidad en determinadas situaciones. En el caso de que una plantilla en particular no sea aplicable, puede sugerir otras plantillas que pueden ser apropiadas para la situación.
4. Se puede fácilmente almacenar en las ranuras valores dinámicos de variables, durante la ejecución de un sistema basado en conocimiento. Esto puede ser particularmente útil para aplicaciones de simulación, planeamiento, diagnóstico de problemas o interfaces para bases de datos.

***Desventajas de las Plantillas***

Las principales desventajas que se pueden establecer para la representación del conocimiento mediante plantillas, son:

1. Dificultad de representar objetos que se alejen considerablemente de estereotipos.
2. No tiene la posibilidad de acomodarse a situaciones u objetos nuevos.
3. Dificultad para describir conocimiento heurístico que es mucho más fácilmente representado mediante reglas

**Representación mediante Objetos**

Los objetos, son similares a las plantillas. Ambos sirven para agrupar conocimiento asociado, soportan herencia, abstracción y el concepto de procedimientos agregados. La diferencia radica en los siguiente:

1. En las plantillas, a los programas y a los datos se los trata como dos entidades relacionadas separadas. En cambio en los objetos se crea una fuerte unidad entre los procedimientos (*métodos*) y los datos.

2. Los *demons* de las plantillas sirven sólo para computar valores para las diversas ranuras o para mantener la integridad de la base de conocimientos cada vez que una acción de alguna plantilla, afecta a otra. En cambio, los *métodos* utilizados por los objetos son más universales ya que proporcionan cualquier tipo general de computación requerida y además soportan encapsulamiento y polimorfismo.

Un objeto es definido como una colección de información representando una entidad del mundo real y una descripción de cómo debe ser manipulada esta información, esto es los *métodos*. Es decir, un objeto tiene un nombre, una caracterización de clase, varios atributos distintivos y un conjunto de operaciones. La relación entre los objetos viene definida por los *mensajes*. Cuando un objeto recibe un *mensaje* válido, responde con una acción apropiada, retornando un resultado.

| | |
|---|---|
| NOMBRE OBJETO | Limpiador Izquierdo |
| UN-TIPO-DE | Limpiador |
| ATRIBUTOS | Hecho de metal y caucho<br><br>Longitud: 14 pulgadas<br><br>Color: negro y plateado<br><br>Localización: inferior izquierda<br><br>Función: remover humedad de parabrisa |
| OPERACIONES | Activado: se mueve en arco sobre el parabrisa repetidamente de izquierda a derecha<br><br>Desactivado: se mueve a posición de reposo |

***Ventajas de la Representación mediante Objetos***

Los objetos, como forma de representación del conocimiento ofrece las siguientes ventajas:

1. Poder de abstracción.
2. Encapsulamiento o capacidad de esconder información.
3. Herencia, es decir pueden recibir características o propiedades de sus ancestros.
4. Polimorfismo, que permite crear una interfaz común para todos los diversos objetos utilizados dentro del dominio.

5. Posibilidad de reutilización del código.
6. Mayor facilidad para poder trabajar eficientemente con sistemas grandes.

***Desventajas de la Representación mediante Objetos***

Las desventajas son similares a las que se indicaron para las plantillas:

1. Dificultades para manejar objetos que se alejan demasiado de la norma.
2. Dificultades para manejar situaciones que han sido encontradas previamente.

**Artículo I. REDES DE INFORMACIÓN.**

Cuando ejecuta una red de área local, su objetivo fundamental es, normalmente, proporcionar a sus usuarios un entorno que haga a la red transparente.

Para este fin una importante piedra de toque es mantener datos vitales, como la información de cuentas de usuario, sincronizadas entre todos los nodos. Hemos visto anteriormente que para resolver nombres de nodos existe un potente y sofisticado servicio denominado DNS.

Para otras tareas, sin embargo, no existe un servicio especializado similar. Más aún, si usted sólo está administrando una pequeña LAN sin conexión a Internet, puede que no merezca la pena el esfuerzo de instalar un DNS.

Esta es la razón por la que Sun desarrolló NIS, el *Sistema de Información de Redes*. NIS proporciona facilidades de acceso genérico a bases de datos que puede ser usado para distribuir información como la contenida en los ficheros passwd y groups entre todos los nodos de su red.

Esto hace que la red aparezca como un sistema único, con las mismas cuentas en todos los nodos. De forma similar usted puede usar NIS para distribuir el fichero de información de nombres de nodos /etc/hosts entre todas las máquinas de la red.

NIS está basado en RPC, y comprende un servidor, una librería para la parte del cliente, y varias herramientas de administración. Originalmente NIS era denominado *Yellow Pages*[10.1], o YP, que todavía son términos ampliamente usados para referirse informalmente a este servicio. Por otra parte Yellow Pages es una marca registrada de la British Telecom, la cual pidió que Sun dejara de utilizar ese nombre. Mientras tanto, algunos nombres impactan mucho entre la gente, YP continúa viviendo como prefijo en los nombres de la mayoría de los comandos relacionados con NIS como ypserv, ypbind, etc.

Hoy en día NIS está disponible en, virtualmente, todos los , y hay incluso implementaciones gratuitas de él. Una de ellas es de la emisión BSD Net-2, que ha derivado de una implementación de referencia de dominio público donada por Sun.

El código de la librería cliente de ésta versión existe en la GNU libc desde hace mucho tiempo, mientras que los programas de administración han sido recientemente portados a por Swen Thümmler. Falta, sin embargo, un servidor NIS en la implementación de referencia. Tobias Reber ha escrito otro paquete NIS incluyendo todas las herramientas y un servidor; se llama yps.

Actualmente, él código del NIS está siendo reescrito por completo por Peter Eriksson, será denominado NYS y soportará ambos, tanto el NIS normal como la amplia revisión de Sun, el NIS$+$. NYS no sólo proporciona un conjunto de herramientas NIS y un servidor, sino que también añade un nuevo y completo conjunto de funciones de librería que muy probablemente lo convertirán con el tiempo en el estándar libc. Esto incluye un nuevo sistema de configuración para resolver nombres de nodos que reemplace el sistema actual que usa el fichero host.conf. Las características de esas funciones serán discutidas más adelante.

Este capítulo se enfocará en NYS más que en los otros dos paquetes, a los que nos referiremos como el ``tradicional'' código NIS. Si usted desea utilizar alguno de esos paquetes, las instrucciones de este capítulo podrían ser suficientes o tal vez no. Para obtener información adicional, por favor, obtenga un libro sobre NIS estándar como el *NFS y NIS* de Hal Stern (ver [#!stern-nfs!#]).

Por el momento NYS está todavía en desarrollo y por lo tanto las utilidades estándar de como los programas de red o el programa de login todavía no tienen en cuenta el sistema de configuración de NYS. Por lo tanto, hasta que NYS no sea incluido en la libc principal tendrá que compilar todos esos programas usted mismo si quiere conseguir que usen NYS. Para ello, en los Makefiles de cualquiera de esas aplicaciones deberá especificar -lnsl como la última opción antes de libc al linkador. Esto enlazará las funciones relevantes de libnsl, la librería NYS, en lugar de la librería C estándar.

**REDES INTELIGENTES.**

A pesar de la existencia del concepto desde la década del setenta, la Red Inteligente solamente empezó a recibir una atención seria cuando la liberalización fue presentada en los Estados Unidos a principios de los años ochenta. Cuando los operadores de redes públicas encontraron competencia entre ellos mismos, se volvió importante proveer rápidamente servicios que los clientes estaban demandando. La Red Inteligente solucionó el problema suministrando una plataforma para la implementación rápida de los servicios. En donde la entrega del servicio, desde el concepto a la activación, podría convencionalmente tomar dos o más años, los proveedores de Red Inteligente prometen un tiempo de entrega menor a un año. La Red Inteligente requiere una gran inversión inicial y la experiencia parece mostrar que la inversión se recupera en un largo plazo.

En la mayoría de los países de occidente, el mercado para los operadores públicos ha estado completamente saturado. La Red Inteligente provee a los operadores la habilidad de generar nuevos ingresos a través de innovación en los servicios,

generando innumerables beneficios para los operadores, proveedores del servicio, subscriptores, usuarios, etc.

En cuanto a las redes móviles el control es idealmente apto para la Red Inteligente, donde el Punto de Datos del Servicio (SDP, Service Data Point) de la Red Inteligente puede actuar como el registro de localización visitante (VLR) y el registro de localización local (HLR); y el Punto de Control del Servicio (SCP, Service Control Point) puede controlar el "roaming" y tal vez, el "handover". La Red Inteligente es una arquitectura de red de telecomunicaciones creada para soportar múltiples servicios, es por ello que se han diseñado los servicios con una estructura modular, lo que conlleva que sean creados y modificados fácilmente.

Adicionalmente, les da la posibilidad a los proveedores de diferenciar sus servicios de los de su competencia. Los operadores de redes están siendo forzados a evaluar sus estrategias e infraestructuras actuales para determinar cómo deberían proceder en el sentido de prepararse para futuros servicios de telecomunicaciones.

En este documento se realizará un estudio de los servicios que ofrece la Red Inteligente a estos operadores, basados en las recomendaciones de la UIT-T e información disponible por los proveedores de plataformas de Red Inteligente como Alcatel, Ericsson, Lucent, Nortel y Siemens. En la última parte de este documento se presentará una visión global de las tendencias evolutivas de los servicios de la Red Inteligente.

**GENERALIDADES.**

A nivel mundial la Unión Internacional de Telecomunicaciones - Sector de normalización (ITU-T) regula todo lo relacionado con Red Inteligente (RI), en el conjunto de capacidades uno y dos. Otro organismo regulador de RI es el Instituto de Estándares de Telecomunicaciones Europeo (ETSI), este documento se basa en CS-1 y CS-2 de la ITU-T.

En el conjunto de capacidades 1 (CS-1) normaliza la RI como concepto arquitectural para la creación y suministro de servicios de telecomunicaciones.

Un servicio es una oferta comercial autónoma constituida por una o más características de servicio modulares.

Una característica de servicio (SF, Service Features) es un aspecto específico de un servicio que puede utilizarse también junto con otros servicios/características de servicio como parte de la oferta comercial.

La caracterización de los servicios y las características de los servicios sirven para identificar capacidades independientes del servicio que se requieren para construir y/o personalizar servicios por los usuarios o los operadores de red.

Como ejemplos de capacidades de servicio requeridas cabe citar, desde el punto de vista del usuario, la cola de espera de llamadas, anuncios personalizados, etc.

Aunque por su propia naturaleza, la RI tiene una arquitectura independiente del servicio, resulta pertinente describir las capacidades de servicio generales del CS-1.

Los servicios y las características de servicio que ha de admitir el CS-1 son fundamentalmente para los bloques de construcción independientes del servicio (SIB, Service-independent Building Blocks) del CS-1, el modelo de tratamiento de la llamada y los principios de control del servicio.

La arquitectura del CS1 tiene en cuenta las necesidades de evolución, es decir, admite los servicios previstos del CS1 pero sus funcionalidades están diseñadas de manera tal que puedan evolucionar hacia los futuros conjuntos de capacidades (CS-2 y posteriores).

Por tanto, en la definición de las capacidades del CS1 se ha evitado toda hipótesis capaz de limita.Todo esto le permite al proveedor poder ofrecer muchos servicios diferentes a los que el CS-1 especifico o normalizó.

a) Servicios de tipo A y tipo B. Las características de servicio y los servicios del CS-1 se incluyen en la categoría de "un solo extremo" y "un solo punto de control" considerados como de tipo A, mientras que todos los demás servicios se incluyen en una categoría denominada tipo B. Se aplican las siguientes definiciones: una característica de servicio de un solo extremo se aplica exclusivamente a una parte de una llamada y es ortogonal (independiente) en los niveles de servicio y topológico respecto a cualquier otra parte que pueda participar en la llamada.

Por un solo punto de control se entiende una relación de control en la que ciertos aspectos de una llamada dependen exclusivamente de una sola función de control de servicio en cualquier momento. Las normas del CS-1 no abarcan los servicios de "tipo B" por los siguientes motivos.

***Complejidad operacional.***

En los servicios de tipo B varios abonados RI pueden estar asociados dentro de una sola llamada. En el curso de la llamada se pueden añadir o retirar abonados. Esas asociaciones se producen físicamente en las conmutaciones implicadas en la llamada (funciones SSF/CCF) bajo el control de una SCF. Esta necesitará reglas para tratar de arbitrar las características de los abonados participantes en la llamada (por ejemplo lista de selección incompatible). Ello puede exigir consultas en tiempo real entre las SCF que "representan" las distintas partes implicadas en la llamada. También se necesitarán reglas para abordar las decisiones topológicas (por ejemplo, ¿qué conmutaciones físicas se deben elegir para "unir" a los grupos de abonados dispersos a lo largo de una red?).

***Complejidad de la realización.***

Los servicio de tipo B pueden exigir la manipulación de recursos de conexión de conmutación por parte de una lógica de servicio situada en una función de control de servicio. Ello significa que la lógica de servicio externa debe disponer de una visión "abstracta" de los recursos de conexión de la conmutación.

***Complejidad del control.***

Los servicios de tipo A se caracterizan por una relación de control relativamente sencilla entre las SSF y SCF. La SSF es "cliente" de la información relacionada con el servicio proporcionada por la SCF, pero sin embargo, la conmutación conserva el control de la conexión en todo momento.

Por el contrario, la relación de control entre las funciones SCF y SSF en los servicios de tipo B puede requerir la compartición del control de la conexión entre la conmutación y la lógica de servicio externa.

El flujo de información tiene que ser rico en parámetros para gestionar lo que es fundamentalmente una relación entre homólogos de tratamiento distribuido. Dadas las considerables diferencias existentes en la complejidad operacional, de realización y de control de los servicios de tipo A y B, el CS-1 está ideado para admitir sólo los servicios de tipo A. 1.1.1 Conjunto previstos de servicios y características de servicio CS-1.

Las Tablas 1 y 2 detallan los conjuntos previstos de servicios y características de servicio CS-1. En las Tablas 3 y 4 se muestran ejemplos de las relaciones y correspondencia entre esos servicios y características de servicio CS-1. 1.1.2 Servicios y características de referencia del CS-2 de RI.

Con respecto al conjunto de capacidades dos (CS-2) de RI es la segunda etapa normalizada de la Red Inteligente como un concepto arquitectural para la creación y prestación de servicios, que incluye servicios de telecomunicaciones, servicios de gestión de servicios y servicios de creación de servicios, los dos últimos tipos de servicios se presentan por primera vez en el CS-2 de RI. Para el CS-2 de la RI, los servicios de telecomunicaciones, servicios de gestión y servicios de creación de servicios están contenidos en el plano de servicio; se pueden describir al usuario de extremo por medio de características del servicio de telecomunicaciones, características de servicio de gestión y características del servicio de creación de servicios.

Los servicios y características de servicios incluidos en el CS-2 de la RI son un súper conjunto de los incluidos en el CS-1 de la RI, es decir, incluyen servicios y características adicionales a los ya identificados en el CS-1 de la RI.

La definición de estos tres tipos de características de servicio son las siguientes: Características de servicios de telecomunicaciones.

Los aspectos específicos de servicios de telecomunicaciones que pueden ser usados individualmente o junto con otros servicios/características de servicio de telecomunicaciones como parte de una oferta comercial.

Pueden ser partes esenciales o facultativas ofrecidas como mejoras del servicio de telecomunicaciones.

## SERVICIO IMPLEMENTADOS EN LAS PLATAFORMAS

A continuación se nombraran los servicios que están disponibles en las plataformas de los principales proveedores de Red Inteligente como Alcatel, Ericsson, Lucent, Nortel y Siemens, se ha escogido el estudio de estas empresas debido a su alcance en el mercado y porque muchas de las operadoras de telecomunicaciones nacionales poseen plataformas de RI de dichos proveedores.

Muchos de los servicios están especificados como aparece en la recomendación, sin embargo algunos tienen características especiales que por su importancia se mencionarán, así como también, servicios totalmente novedosos y que no son nombrados en las recomendaciones de la UIT-T.

- SERVICIOS DE LA PLATAFORMA ALCATEL.
- SERVICIOS DE LA PLATAFORMA ERICSSON.
- SERVICIOS DE LA PLATAFORMA LUCENT.
- SERVICIOS DE LA PLATAFORMA NORTEL.
- SERVICIOS DE LA PLATAFORMA SIEMENS.

## TENDENCIAS DE LA RED INTELIGENTE

La Red Inteligente ha sido introducida con el objetivo de una creación eficiente, despliegue y gestión de servicios suplementarios que mejoren los servicios básicos.

Las recomendaciones de la ITU-T para Red Inteligente CS-1 y CS-2 soportan servicios de banda estrecha, la tendencia evolutiva de la RI es hacia servicios banda ancha es por ello que se estudia la integración de Red Inteligente y RDSI banda ancha la cual ha sido considerada en CS-3 y CS-4.

Uno de los organismos que estudia esta evolución de la RI es EURESCOM la cual maneja proyectos relacionados con el tema que se serán tenidos en cuenta en las futuras actualizaciones de los estándares de la ITU-T.

El proyecto 607 busca integrar y unificar las técnicas de RDSI y RI, este proyecto es la continuación del 506 y analizará y continuará desarrollando la provisión de servicios multimedia.

Para la realización de estos servicios se han definido los HLSIBs (SIBs de alto nivel) los cuales pueden ejecutar secuencias de nuevas operaciones.

Los servicios referenciados en el proyecto de Eurescom son:

- Video conferencia por demanda.
- Ingreso en video conferencia.
- Video en demanda.
- TV distribuida.
- VPN banda ancha.

Es bien conocido por todos el auge de Internet en el mercado de las telecomunicaciones, ofreciendo grandes posibilidades y aumentando cada día el número de personas que acceden a este red.

Como se puede notar muchas de los proveedores de plataformas de Red Inteligente tienen la tendencia de implementar servicios de Red Inteligente para Internet, que le permita a los usuarios realizar comunicaciones de voz por este medio, así como también manejar la gestión del servicio, permitiendo al usuario de la red acceder a

cambiar su perfil desde una página web que le permita una gran gama de posibilidades en un entorno amigable.

Software de monitoreo inteligente en redes

**MICROMUSE BRINDA SOLUCIÓN DE MONITOREO INTELIGENTE PARA INFRAESTRUCTURAS DE REDES**

Micromuse, Inc., proveedor líder de software para garantizar y asegurar la disponibilidad de los servicios de las empresas, presenta Netcool(r)/OMNIbus(tm) Versión 3.5, plataforma líder de la industria para detección de fallas en tiempo real y monitoreo de la ejecución de infraestructuras TI.

La nueva versión brinda un nuevo nivel de rendimiento y escalabilidad. En implantaciones iniciales, las innovadoras arquitecturas distribuidas de Netcool/OMNIbus v3.5 ayudan a soportar varios cientos de operadores de equipos de escritorio simultáneos y a relacionar más de 30 millones de alarmas en bruto por día.

La escalabilidad única de Netcool/OMNIbus v3.5 satisface los retos de administrar las más grandes infraestructuras TI y redes. Brinda una ventaja competitiva en varios entornos TI en un amplio rango de industrias para garantizar los servicios y la disponibilidad del negocio; en el cómputo móvil, donde se integra a nuevos dispositivos; en el comercio electrónico, donde el uso de Internet alcanzan cada vez más usuarios de lo anticipado; en servicios de carriers, donde se pueden experimentar fallas masivas en la infraestructura poniendo el negocio en riesgo; en el gobierno, las empresas financieras y otras empresas, donde se enfrentan regularmente violaciones de seguridad basadas en Internet.

Para mejorar el rendimiento Netcool/OMNIbus v3.5 ofrece las siguientes ventajas técnicas y de negocio:

- Capacidad de archivo de alta velocidad Netcool Gateway para Oracle, con rendimiento mejorado de 3 a 10 veces.
- Porciones protegidas en los equipos de los usuarios para prevenir cambios inadvertidos en el operador que puedan resultar en pérdida en la visibilidad de la infraestructura.
- Configuraciones mejoradas en un solo punto de administración de fallas y arquitecturas distribuidas.
- Incremento en la eficiencia de equipos de escritorio con los nuevos menús que aparecen automáticamente y botones de clic que permite a los equipos de operación aislar problemas rápidamente.

- Biblioteca de módulos de software propios que reúne y consolida información de fallas de más de 1,000 entornos incluyendo seguridad, voz sobre IP, ancho de banda DSL, inalámbrico, switches, ruteadores y sistemas de administración empresarial.

Netcool/OMNIbus es una base de datos de alta velocidad residente en memoria que recopila información de fallas procedentes de toda la infraestructura de red y permite a los operadores establecer relaciones con la disponibilidad de los servicios empresariales.

Las ventajas que ofrece Netcool/OMNIbus v3.5 son la variedad en conectividad de tecnología y plataformas, mayores beneficios en escalabilidad y ejecución, aceleración en tareas administrativas, mejoramiento de la eficiencia de usuarios de equipos de escritorio y reducción de tiempo para reparaciones, ayudando con una reducción inteligente de los eventos al brindar capacidades de administración eficaces.

Requisitos de plataformas para instalar Netcool/OMNIbus v.3.5:

- HP PA RISC, basado en plataformas HP/UX 10.20, HP/UX 11, HP/UX 11i con 120 MB de espacio en disco.
- IBM Power PC y SR/6000, basado en plataformas AIX 4.3.3 y AIX 5L con 130 MB de espacio en disco.
- Sun Microsystems Sparc, basado en plataformas Solaris 2.6, Solaris 2.7 y solaris 2.8 con espacio de 200 MB de espacio en disco.
- Servidores Microsoft con plataformas Windows NT 4.0, Windows 2000 y Windows XP (solo desktop) con 70 MB de espacio.
- Intel, basados en plataformas Linux Red Hat como Red Hat Linux 6.2 y Red Hat Linux 4.x con 110 MB de espacio.

*"Como parte de la solución completa Netcool, la versión 3.5 de Netcool/OMNIbus amplía el liderazgo tecnológico de Micromuse en el área de administración inteligente de infraestructura y ayuda a satisfacer las necesidades de los entornos de redes más demandantes del mundo,"* afirmó Gonzalo Zermeño, Country Manager de Micromuse México. *"Micromuse actualiza sus versiones de software con la finalidad de estar a la vanguardia y ofrecer al mercado lo más nuevo en soluciones tecnológicas."*

Acerca de Micromuse

Micromuse Inc. (Nasdaq: MUSE) es el proveedor líder de software que garantiza el más alto nivel de calidad en los procesos de negocio y servicios. Entre los reconocimientos más recientes que ha recibido Micromuse se encuentra el premio Crossroads A-List otorgado a la aplicación Netcool®/Visionary(TM) mejor producto de Diagnóstico en redes IP. Además, Micromuse fue recientemente reconocida por las siguientes publicaciones: Forbes 500, Bloomberg's Tech 100, Barron's 500,

Business Week Info Tech 100, Deloitte & Touche's Technology Fast 500, San Francisco Chronicle 500 y Network World's NW200.

IBM puede ayudarle a construir una infraestructura de red inteligente que permite a las compañías -independientemente de su tamaño- manejar su inversión en IT y soportar sus objetivos de negocio. Nosotros ofrecemos soluciones e-business en cualquier ciclo de vida en la red -desde consultoría de redes e integración hasta total tercerización de su red.

A través de nuestras numerosas alianzas, incluyendo a Cisco, Avaya, AT&T, estamos listos para acercarle avances emergentes en óptica, acceso remoto o wireless, convergencia de IP, seguridad y red inteligente para ofrecer articuladas, soluciones "best-of-breed". Los servicios de Redes IBM proveen simple responsabilidad y experiencia mundial real en servicios, mientras brindamos a los clientes el poder de comprar y elegir lo que nuestros Asociados proveen.

**Multiservice Networking Infraestructure – Avaya**

Cajún es una solución para trabajo en grupo, combina una matriz de conmutación de elevado rendimiento, capacidad de expansión flexible, instalación y administración simplificadas y capacidades complejas de monitoreo de redes. Complementadas con servicios de IBM.

**VPNet Solutions – Avaya**

Servicios IPSec garantizan a las organizaciones una comunicación segura para usuarios remotos, sucursales, partners y clientes de la red corporativa.

El dispositivo VSU se emplea detrás del router de acceso para brindar una solución VPN de acceso remoto y "site-to-site". Complementada con servicios IBM.

**A3 es un Sistema Inteligente**

que permite detectar y resolver en tiempo real complejos problemas de Proceso, aumentando la eficiencia de la Planta.

**Qué es un Sistema Inteligente**.

A3 esta desarrollado con la más avanzada tecnología de Sistemas Expertos, lo que permite representar y utilizar el conocimiento disponible sobre productos y procesos para diagnosticar continuamente situaciones problemáticas o potencialmente problemáticas que son muy complejas para ser detectadas por los sistemas convencionales de control.

Estos problemas complejos generalmente requieren analizar simultáneamente muchas variables, tanto su valor actual como su comportamiento o tendencia en un

intervalo reciente de tiempo, tarea que por lo general es realizada por los operadores o supervisores del proceso.

Al ser esta una tarea típicamente humana, se entiende que la habilidad, conocimiento y experiencia, juegan un papel fundamental en la identificación y solución del Problema. Así, un Sistema Inteligente es capaz
de realizar este tipo de tareas complejas y , al igual que un ser humano, va adquiriendo experiencia y habilidad con la práctica.

**Cómo funciona A3**

A3 es un Sistema en Tiempo Real, lo que significa que puede monitorear continuamente variables del Proceso, Set points de control y parámetros de operación.

El **módulo de monitoreo .**

Permite representar gráficamente la Planta y los procesos usando tecnología de objetos además de especificar todos los atributos y variables necesarios para definir los Problemas a Diagnosticar. A3 cuenta con capacidades de integración en tiempo real a una gran variedad de sistemas
de adquisición de datos y control , así como a la mayoría de las Bases de datos comerciales.

El **módulo de Problemas.**

Permite definir los problemas que se diagnosticarán contínuamente a partir del módulo de monitoreo. Inicialmente, se pueden integrar en A3 todos los problemas de los cuales se tenga conocimiento previo.

Una vez en operación, nuevos problemas pueden agregarse en tiempo real ó bien nuevo conocimiento para el diagnóstico y análisis de los problemas existentes.

Un **Problema** es una situación compleja que se desea diagnosticar a partir del monitoreo de losprocesos.

Para **definir un problema** se requiere:

**Detalle**: Es una descripción textual del Problema

**Tipo**: Es una clasificación de los problemas para facilitar su Administración

**Área**: Es la Área de la Planta o Proceso asociado o afectado por la ocurrencia del problema.

**Condiciones**

Es la descripción de la información del Proceso que se desea almacenar cuando ocurre un evento de este problema. Esta información es tomada del módulo de monitoreo y congelada como un video para ser almacenada en la historia del problema.

**Diagnóstico**

En esta sección se define todo el conocimiento que se tiene para identificar la ocurrencia del problema en tiempo real.

**El diagnóstico**

No es un texto informativo, es conocimiento activo que se ejecuta contínuamente por el módulo de respectivo y que puede modificarse para mejorar su capacidad de identificar el problema.
A3 tiene una gran variedad de formas de representar conocimiento, incluyendo reglas y objetos dinámicos .

Cada problema puede tener un período de diagnóstico particular, el cual puede configurarse para ser ejecutado en los intervalos de tiempo requeridos por el proceso.

**Causas**

Cada problema puede estar asociado a una o más causas posibles.
A3 aprende, mediante la experiencia, las causas que han generado el problema además de mantener actualizada la probabilidad empírica de ocurrencia del problema por cada posible causa.

**Análisis de causa**:

A3 no solo puede diagnosticar problemas y mostrar sus posibles causas.
Dada la ocurrencia de un problema, A3 puede activar su conocimiento para identificar la causa específica de esa ocurrencia.
Al igual que el diagnóstico, el Análisis de causas es un conocimiento activo, aunque opcional, que ayuda en la solución del Problema.

**Acciones**

A3 es un Sistema de Tiempo Real, con la capacidad de ejecutar acciones correctivas para solucionar los problemas detectados.

Estas acciones correctivas pueden ser instrucciones al operador o rutinas que actúen sobre los sistemas de control, como prender o apagar un equipo, un procedimiento de paro de línea, o una rutina de arranque.

Las acciones pueden ser generales para el problema o especificas para cada posible causa.

**Modo**

Cada problema puede operar en 2 modos excluyentes.

**En-línea**

Este es el modo de operación en tiempo real. El módulo de diagnóstico evalúa contínuamente todos los problemas que están en-línea.

**Fuera-de-línea.**

Este modo de simulación, permite validar las reglas de diagnóstico y análisis sin esperar la ocurrencia de una situación real.

El **Modulo de diagnóstico** ejecuta las instrucciones de cada problema en-línea para identificar su ocurrencia de acuerdo con la frecuencia de diagnostico establecida.
Cuando se detecta algún problema, se realizan las siguientes funciones:

1 – Se genera un **evento**, que es una ocurrencia activa del problema y se agrega a la ventana de eventos
2 – Se lee del módulo de historia, los **casos** registrados de ocurrencia del problema.
3 – Se ejecuta una señal de **alarma** al operador.
4 – Se lee y se congela en el evento toda la información y datos definidos en las condiciones del problema al momento de la ocurrencia del evento.
5 – Se dispara el proceso de análisis de causas del problema

Un **caso** es una ocurrencia anterior del problema que ha sido registrado en el modulo de historia. Cada caso contiene información importante, como:

- Condiciones del proceso al momento de ocurrir el
- caso
- Causa identificada
- Acciones ejecutadas para resolver el problema
- Calificación de la solución.

La **ventana de eventos** muestra todos los eventos activos en el momento. Los eventos pueden ordenarse por prioridad y filtrarse por área o tipo de problema.

Cada evento puede analizarse detalladamente:

- Cuáles son las condiciones de proceso al ocurrir el
- evento
- Cuáles son los casos disponibles
- Cuál es la mejor solución conocida para cada causa (Mejor Práctica)

**Solución de un Problema**

El operador puede emplear la experiencia de los casos anteriores para resolver el problema o puede emplear otra alternativa si la experiencia previa no es del todo satisfactoria. Una vez resuelto el problema, el operador debe alimentar en el evento la causa identificada, qué acciones realizó en esta ocasión y cual es la calificación asignada a esta solución.

Un criterio útil de calificación puede ser el tiempo empleado en la solución del problema u otro parámetro de relevancia en la operación.

Una vez resuelto el problema A3 realiza las siguientes funciones:

1 Almacena el evento en el modulo de historia como un nuevo caso
2 Actualiza las probabilidades causa-efecto del problema
3 Actualiza la mejor práctica para la causa identificada en el evento.

El **Módulo de historia**

Es el repositorio de casos de A3.

Es administrado automáticamente por A3 tanto en la búsqueda de casos como agregando eventos resueltos.

Se pueden emplear diferentes bases de datos comerciales para aprovechar la infraestructura disponible en cada instalación de A3. Algunas opciones disponibles de Base de datos son ORACLE, Sybase, SQL Base.

**Como aprende A3**.

Basado en la más avanzada tecnología de Sistemas Inteligentes, A3 puede aprender cualquier cosa que puede aprender un operador usando los mismos esquemas de aprendizaje.

***Enseñanza*.**

A3 aprende cuando el usuario define nuevos problemas y alimenta conocimiento para el diagnóstico, análisis de causas o acciones a ejecutar.

El proceso de enseñanza puede ser en línea, esto es, se puede agregar o modificar conocimiento mientras A3 realiza su operación en el proceso.

***Experiencia*.**
Cuando un evento es resuelto A3 actualiza las causas del problema así como la mejor práctica asociada a dicha causa y las probabilidades causaefecto.

De esta forma, causas no identificadas anteriormente se agregan al conocimiento del problema para futuro análisis.

**Manejo de conocimiento en A3.**
Existe una gran variedad de formas de representar conocimiento en A3, lo que permite aplicarlo a cualquier situación real simple o compleja. Algunas de las más importantes son las siguientes:

**Objetos** – Para representar equipos, productos y procesos incluyendo sus atributos y variables.

**Reglas** – Para definir diagnóstico o análisis de causas

**Fórmulas** – Para definir expresiones matemáticas

**Modelos** – Para representar Sistemas de ecuaciones

**Algoritmos y Procedimientos** –
Para ejecutar acciones

**Redes Neurales** - Para sofisticadas técnicas de análisis no lineal o multivariable

**Integración avanzada**. A3 puede integrar sofisticadas herramientas de análisis como Neuronline y GDA de Gensym ya que opera sobre la misma base tecnológica.

**Modo de Simulación.**

Este modo de operación permite apoyar el entrenamiento de operadores simulando, a partir del módulo de historia, la ocurrencia de eventos y mostrando las condiciones del proceso para cada uno de ellos.

El operador, bajo entrenamiento, puede analizar las condiciones para identificar la posible causa y proponer las acciones de solución.

También puede consultar la mejor práctica conocida para una situación específica.

**Beneficios**

Entre los muchos beneficios que puede generar A3 se encuentran los siguientes:

- Institucionalización del conocimiento de operación
- La capacidad de tener el conocimiento del mejor operador los tres turnos
- La capacidad de que los operadores novatos se apoyen en la experiencia acumulada de los anteriores
- Operadores para identificar problemas y tomar mejores decisiones.
- El tener disponible lo anterior definitivamente incrementa el desempeño de la planta ó proceso.

Configuración requerida:

A3 se ofrece para estaciones de trabajo con plataforma UNIX, como las fabricadas por SUN, DEC,, IBM, HP y otros. Así como para sistemas DEC VAX/VMS y ALPHA AXP.
A3 requiere G2 ver 4.1 ó posterior.

# GLOSARIO

**Lógica Deductiva.**
Lógica tradicionalmente usada en sistemas expertos, definiendo un método para el razonamiento que parte de un concepto general para llegar a otros específicos.

**Método Heurístico.**
Resolución de problemas, probando diferentes métodos y comparando cual es el que ofrece la mejor solución.

**Base de Conocimiento**.
En inteligencia artificial, una recopilación del conocimiento y reglas sobre un tema específico.

**Lógica Inductiva.**
Lógica donde la agrupación de reglas que describen comportamientos particulares que conducen a una regla general.

**LISP.**
Un idioma de programación basado en lógica de predicado y uno de los más usados en aplicaciones de inteligencia artificiales.

**Fractales.**
Modelos matemáticos para describir la naturaleza irregular de líneas, planos o volúmenes. Se pueden aplicar para representar modelos de datos.

**Base de Conocimiento.**
En inteligencia artificial, una recopilación del conocimiento y reglas sobre un tema específico.

**Sistemas de Clasificación (Classifier Systems).**
En inteligencia artificial, estos sistemas ejecutan un tipo de aprendizaje que genera reglas a partir de ejemplos.

**Sistemas Determinísticos.**
Sistemas en los que el resultado es determinado por una ecuación, donde la causa y sus efectos son fácilmente reconocibles.

**Ada Lovelace**: (1811-1852).
Hija del escritor inglés Lord Byron, fue la primera programadora de la historia. Colaboró con el inventor Charles Babbage en el diseño de máquinas de computación en la primera mitad del siglo XIX. (Véase también Ada.)

**ADA**:
Lenguaje de programación derivado del Pascal. Recibió ese nombre en homenaje a Ada Lovelace, la primera programadora de la historia. Fue desarrollado en 1979 por un equipo

dirigido por Jean Ichbiah, por encargo del Pentágono para los proyectos de software del Departamento de Defensa de los Estados Unidos.

**ARPA:** Advanced Research Proyects Agency:

Agencia del Departamento de Defensa de los Estados Unidos que creó ARPAnet, la red que dio origen a Internet.

**ARPANet:** Advanced Research Proyects Agency Network:

Red de comunicación desarrollada por ARPA a fines de la década de los 60. Se la considera el origen de la actual Internet.

**Backbone:**

Columna vertebral. Infraestructura de la transmisión de datos en Internet.

**Back-end processor:**

Procesador que se utiliza para determinada función muy especializada, como por ejemplo, administrar una base de datos.

**BCC:** blind carbon copy.

Función que permite mandar un mensaje de e-mail a más de un destinatario. A diferencia de a función CC, el nombre de los destinatarios no aparece en el encabezado.

**Benchmark:**

Programa especialmente diseñado para evaluar el rendimiento de un sistema, de software o de hardware.

**Betatest:**

En el proceso de desarrollo de software, es la segunda fase de la verificación o testeo, previa al lanzamiento del producto.

**Buscador, motor de búsqueda:** search engine.

Es un programa, ubicado en un sitio de Internet, que recibe un pedido de búsqueda, lo compara con las entradas de su base de datos y devuelve el resultado. Algunos de los más conocidos: Yahoo, Altavista, Lycos, Infoseek.

**DSL**: Digital Subscriber Line:

Línea Digital de Suscripción. Tecnología que permite enviar mucha información a gran velocidad a través de líneas telefónicas.

**API**: Aplication Program Interface.

Es el conjunto de rutinas del sistema que se pueden usar en un programa para la gestión de entrada/salida, gestión de ficheros etc.

**ASN**: Autonomus System Number. Número de sistema autónomo.

Grupo de routers y redes controlados por una única autoridad administrativa.

## 2.3 Perspectivas.

Existen también sistemas de planificación, pudiéndose encontrar aplicaciones en este área, que establecen una secuencia de acciones a realizar encaminadas a la consecución de una serie de objetivos.

En las empresas, la Inteligencia Artificial, que se encontraba confinada en la "sala de computadoras", se va abriendo paso hacia la junta directiva. La razón de esto es simple: a medida que el mundo empresarial se complica y se llega a la competencia internacional, el conocimiento se convierte en el factor profesional más importante para un ejecutivo.

A la persona que esté planeando la estrategia a seguir por su empresa o que tome decisiones en producción, marketing, distribución o asignación de recursos, los Sistemas Expertos le pueden demostrar que se pueden tomar decisiones con más conocimiento, llevando a un aumento de ganancias así como a la obtención de beneficios importantes para la empresa, como el aumento de su capacidad.

La medida de la efectividad de las operaciones de planificación y control de una organización y su sensibilidad a los cambios, son elementos importantes en la buena dirección de la producción. Los planes y las decisiones en la producción se desarrollan y llevan a cabo en un mundo de representaciones simbólicas de hechos y conjeturas, muchas de las cuales no están informatizadas y representan la experiencia y el conocimiento de expertos.

En cada estadio de los procesos de planificación, decisión y control para las operaciones de producción, sea ésta automatizada o no, las personas expertas son las que asesoran, localizan los fallos y dirigen.

Ellas son las que ayudan a interpretar la multitud de datos procedentes de los departamentos de diseño, de la planta de producción y de los representantes de los clientes, observan modelos procesables en dichos datos, prueban mentalmente, y con computadoras, posibles líneas de acción, recomiendan las medidas que la gerencia debe tomar y ayudan luego a poner en marcha sistemas pensados para conseguir planificaciones mejores, operaciones más fluidas y una competencia más efectiva.

Otro sistema de este tipo, aunque en este caso orientado a medicina, es GUIDON, pensado para que lo utilicen las Facultades de Medicina para formar a los médicos en la realización de consultas. GUIDON viene a ser una reorganización de MYCIN con intenciones educativas, por esto, tiene la ventaja adicional de disponer de toda la base de conocimientos de MYCIN además de la experiencia acumulada, por consiguiente, puede recuperar como ejemplo cualquier caso que MYCIN haya tratado.

Además de las áreas de aplicación ya citadas, existen otras como las relativas a los sistemas de interpretación, que realizan tareas de inferencia a partir de una serie de datos observables (p. ej. análisis de imágenes, o bien interpretación de señal).

Para concluir con esta exposición de aplicaciones de Sistemas Expertos, solamente indicar que el futuro de dichas aplicaciones pasa por la imaginación de cada uno, siempre que el área elegida requiera la presencia de un experto para la obtención de cualquier tipo de beneficio.

Los sistemas expertos son una variedad comercial de una clase de programas computacionales llamados sistemas basados en conocimiento. El conocimiento en sistemas expertos es altamente in estructurado, esto es, el proceso de solucionar problemas de un dominio no es manifiesto. Y es establecido explícitamente en relaciones o deductivamente inferidos desde la cadena de proposiciones.

Dado que cada condición que puede ser encontrada puede ser descrita por una regla, los sistemas expertos basados en reglas no pueden manejar eventos no anticipados, pero pueden evolucionar con el uso, y permanece limitado a un estrecho dominio de problemas.

Otra variante de sistemas expertos, uno que no posee esta limitación, emplea una base de conocimiento que consiste de descripciones estructuradas de situaciones, de problemas del mundo real y de decisiones actualmente hechas por expertos humanos.

En medicina, por ejemplo, el registro de un paciente contiene descripciones de datos personales, exámenes físicos y de laboratorio, diagnóstico clínico, tratamiento propuesto, y los resultados de tales tratamientos.

Dada una gran base de datos con tales registros en una especialidad médica, el médico puede indagar acerca de eventos análogos a los relacionados con el paciente. Esto en contraste con el sistema que idealmente intenta reemplazar al ser humano, ya que en casos como estos sólo podría usarse este tipo de conocimiento como una herramienta que ayuda en la toma de decisiones.

El software requerido para este tipo de sistemas se ha ido complicando con el tiempo ya que su desarrollo demanda tiempo, un buen equipo de programadores y un buen producto final.

Se percibe este trabajo como una tarea para el ingeniero y consiste de:

Identificación y análisis de los requerimientos del usuario.

Desarrollo de las especificaciones del sistema (software y hardware).

Diseño del software.

Implementación

Testeo

Manutención

Tareas típicas para sistemas expertos involucran clasificación, diagnóstico, monitoreo, diseño, inventario y planificación para esfuerzos especializados.