Catálogo de tareas y de inferencias

Francisco J. Martín Mateos

Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

Tareas

- Tareas analíticas: Se aplican sobre sistemas que existen antes de llevar a cabo la tarea.
  - La entrada a este tipo de tareas suele ser algún dato acerca el sistema, mientras que la salida es alguna caracterización del mismo.
  - Clasificación, Valoración, Diagnóstico, Monitorización, Predicción.
- Tareas sintéticas: El sistema no existe antes del desarrollo.
  - La entrada a este tipo de tareas son los requisitos del sistema a construir, mientras que la salida es una descripción del sistema construido.
  - Diseño, Configuración, Planificación, Temporalización, Asignación.

Inferencias

- Las inferencias son los elementos constitutivos de los procesos de razonamiento.
  - Los métodos de las tareas se describen como un pseudocódigo en el que los elementos básicos son las inferencias.
- CommonKADS proporciona un catálogo de 18 inferencias que cubre los procesos de razonamiento más habituales.
  - No existe ningún estándar
  - El catálogo puede ser extendido con nuevas inferencias

Catálogo de Inferencias

Estructura del catálogo

- **Nombre** de la inferencia.
- **Funcionamiento**.
- **Ejemplos** de uso.
- **Conocimiento estático**.
- **Tareas** en las que se suele usar.
- **Comportamiento de control**.
- **Métodos** computacionales.

Inferencia **Abstraer** (ABSTRACT)

- **Funcionamiento**: Dado un conjunto de información produce una forma abstracta del mismo (puede incluir el conjunto de entrada).
- **Ejemplo**: La información "Temperatura ≥ 38" se abstrae como "fiebre".
- **Conocimiento estático**: Reglas de abstracción, jerarquía de clases.
- **Tareas**: Analíticas (Asignación).
- **Comportamiento de control**: Puede tener éxito más de una vez.
- **Métodos**: Razonamiento hacia delante, generalización.
- Se pueden producir abstracciones encadenadas.

Inferencia **Agrupar** (GROUP)

- **Funcionamiento**: Dado un conjunto genera un objeto que contiene uno o más elementos del conjunto.
- **Ejemplo**: Agrupar los empleados para que la asignación de oficinas se pueda hacer por grupos.
- **Conocimiento estático**: Conocimiento específico del dominio sobre los requisitos, restricciones y preferencias del agrupamiento.
- **Tareas**: Sintéticas (Asignación)
- **Comportamiento de control**: Puede producir múltiples soluciones.
- **Métodos**: Algoritmos de satisfacción de restricciones.

Inferencia **Asignar** (ASSIGN)

- **Funcionamiento**: Trata de asignar un recurso a un objeto "activo".
- **Ejemplo**: Asignar un despacho a un empleado.
- **Conocimiento estático**: Un conjunto de restricciones y preferencias.
- **Tareas**: Sintéticas (Asignación, Temporalización).
- **Comportamiento de control**: Puede fallar y puede producir más de una salida para la misma entrada.
- **Métodos**: Sistemas basados en reglas, satisfacción de restricciones.
- Es diferente a la tarea de asignación.

Inferencia **Clasificar** (CLASSIFY)

- **Funcionamiento**: Asocia un objeto con la clase a la que pertenece.
- **Ejemplo**: Clasificar manifestaciones en anormales o normales.
- **Conocimiento estático**: Definiciones de clases indicando las características necesarias y las suficientes.
- **Tareas**: Analíticas (Monitorización) y Sintéticas.
- **Comportamiento de control**: Produce una solución precisa.
- **Métodos**: Comparación de patrones.
- En algunos casos hay que tratarla como una tarea: cuando el conocimiento no es completo o cuando se pueden dar varias posibilidades de clasificación.

Inferencia **Comparar** (COMPARE)

- **Funcionamiento**: Produce como salida verdadero, si los dos objetos de entrada son iguales, o falso en caso contrario.
- **Ejemplo**: Comparación de los hallazgos predichos por el sistemas y los que realmente aparecen.
- **Conocimiento estático**: Generalmente no hace falta, aunque puede aparecer (ej., comparación de intervalos).
- **Tareas**: Analíticas (Diagnóstico y Monitorización)
- **Comportamiento de control**: Produce una solución precisa.
- **Métodos**: Algoritmos muy simples.

Inferencia **Corresponder** (MATCH)

- **Funcionamiento**: Dado un conjunto de entradas determina si pueden llevar a un resultado conjunto.
- **Ejemplo**: Determinar si un conjunto de valoraciones nos llevan a una decisión conjunta.
- **Conocimiento estático**: Reglas que indiquen si una combinación de hallazgos llevan a una conclusión.
- **Tareas**: Analíticas (Valoración).
- **Comportamiento de control**: Puede fallar o producir una única solución.
- **Métodos**: Razonamiento hacia delante.

Inferencia **Criticar** (CRITIQUE)

- **Funcionamiento**: Dada una solución especifica los problemas que tiene o la forma en que puede ser mejorada.
- **Ejemplo**: Criticar un diseño.
- **Conocimiento estático**: Depende del dominio y es heurístico y dependiente del contexto.
- **Tareas**: Sintéticas (Configuración).
- **Comportamiento de control**: Puede producir más de una salida.
- **Métodos**: Depende del dominio.

Inferencia **Cubrir** (COVER)

- **Funcionamiento**: Dado un efecto determinar un estado del sistema que haya podido provocarlo.
- **Ejemplo**: Analizar los síntomas de un paciente para determinar la enfermedad que lo causa.
- **Conocimiento estático**: Algún modelo de comportamiento del sistema, por ejemplo una red bayesiana.
- **Tareas**: Analíticas (Diagnóstico).
- **Comportamiento de control**: Puede producir múltiples soluciones.
- **Métodos**: Métodos abductivos.

Inferencia **Especificar** (SPECIFY)

- **Funcionamiento**: Genera un nuevo objeto que está relacionado de alguna manera con el objeto de entrada.
- **Ejemplo**: Especificar una característica observable para una hipótesis.
- **Conocimiento estático**: Reglas específicas del dominio que hagan dicha asociación.
- **Tareas**: Analíticas y Sintéticas.
- **Comportamiento de control**: Puede fallar y puede producir múltiples soluciones.
- **Métodos**: Razonamiento hacia delante.

Inferencia **Evaluar** (EVALUE)

- **Funcionamiento**: Dado un conjunto de entrada y una norma devuelve verdadero si el conjunto cumple la norma. Si la norma no varía se puede considerar como rol estático.
- **Ejemplo**: Evaluar un formulario.
- **Conocimiento estático**: Debe indicar como se puede derivar el valor verdadero del conjunto de datos.
- **Tareas**: Analíticas y Sintéticas (Asignación, Temporalización).
- **Comportamiento de control**: Produce una solución precisa por cada conjunto de datos de entrada.
- **Métodos**: Razonamiento hacia atrás con la norma como objetivo.

Inferencia **Generar** (GENERATE)

- **Funcionamiento**: Dado algún dato del sistema (requisitos, características) produce una posible solución.
- **Ejemplo**: Generar las posibles clases a las que puede pertenecer una roca.
- **Conocimiento estático**:
  - En tareas analíticas: Conocimiento sobre todas las soluciones posibles.
  - En tareas sintéticas: Conocimiento sobre la composición del sistema.
- **Tareas**: Analíticas (Diagnóstico) y Sintéticas (Planificación).
- **Comportamiento de control**: Puede producir múltiples salidas.
- **Métodos**:
  - En tareas analíticas: Algoritmos de búsqueda.
  - En tareas sintéticas: Algoritmos para calcular todas las posibles combinaciones.

Inferencia **Modificar** (MODIFY)

- **Funcionamiento**: Tiene como entrada y salida una descripción del sistema a modificar y las modificaciones que hay que hacer.
- **Ejemplo**: Modificar el diseño de un dispositivo actualizando el modelo.
- **Conocimiento estático**: Restricciones de diseño.
- **Tareas**: Sintéticas (Configuración, Temporalización).
- **Comportamiento de control**: Produce una salida.
- **Métodos**: Una actualización.

Inferencia **Operacionalizar** (OPERATIONALIZE)

- **Funcionamiento**: Dados unos requerimientos para un sistema, transformarlos para que sean usados en el proceso de razonamiento.
- **Ejemplo**: Transformar el requisito "ordenador rápido" en "como mínimo 1 GHz".
- **Conocimiento estático**: Heurísitco y debe ser revisado durante el proceso de desarrollo.
- **Tareas**: Sintéticas (Configuración).
- **Comportamiento de control**: Preferible que aporte distintas alternativas de operacionalización.
- **Métodos**: Razonamiento hacia adelante.

Inferencia **Ordenar** (SORT)

- **Funcionamiento**: Ordena el conjunto de elementos de entrada (la salida es la misma lista ordenada).
- **Ejemplo**: Ordenar un conjunto de diseños de acuerdo con las preferencias.
- **Conocimiento estático**: Función de comparación.
- **Tareas**: Sintéticas.
- **Comportamiento de control**: Sólo da una salida.
- **Métodos**: Algoritmos de ordenación.

Inferencia **Predecir** (PREDICT)

- **Funcionamiento**: Dada un descripción de un sistema, predecir el comportamiento de dicho sistema en el futuro.
- **Ejemplo**: Predecir la presión sanguínea de un paciente.
- **Conocimiento estático**: Modelo del sistema (cualitativo o cuantitativo).
- **Tareas**: Analíticas (Diagnóstico).
- **Comportamiento de control**: Solución única.
- **Métodos**: Razonamiento cualitativo, cálculo matemático.

Inferencia **Proponer** (PROPOSE)

- **Funcionamiento**: Generar un nuevo elemento a ser incluido en el proceso de razonamiento.
- **Ejemplo**: Proponer un modelo de disco duro para el diseño de un ordenador.
- **Conocimiento estático**: Dependencias entre los posibles componentes y preferencias de componentes.
- **Tareas**: Sintéticas (Configuración).
- **Comportamiento de control**: Puede dar varias soluciones.
- **Métodos**: Razonamiento hacia delante, algoritmos que operen con las preferencias.

Inferencia **Seleccionar** (SELECT)

- **Funcionamiento**: Selecciona un elemento, una sublista o un subconjunto de una lista o conjunto.
- **Ejemplo**: Seleccionar una hipótesis de un diagnóstico diferencial.
- **Conocimiento estático**: Criterio de selección específico del dominio.
- **Tareas**: Analíticas y Sintéticas.
- **Comportamiento de control**: Puede dar múltiples soluciones.
- **Métodos**: Algoritmos de selección (puede ser selección aleatoria).

Inferencia **Verificar** (VERIFY)

- **Funcionamiento**: La entrada es una descripción del sistema a ser verificado. La salida es un valor booleano que indica si el sistema ha pasado la verificación. Otra salida puede ser el nombre de la violación en caso de fallo.
- **Ejemplo**: Verificar el diseño de un ordenador.
- **Conocimiento estático**:
  - En tareas analíticas: Conocimiento sobre la consistencia de las hipótesis con el conjunto de datos.
  - En tareas sintéticas: Restricciones internas y externas.
- **Tareas**: Analíticas (Diagnóstico) y Sintéticas (Configuración).
- **Comportamiento de control**: Solución única.
- **Métodos**: Razonamiento hacia adelante.

Catálogo de Tareas

Clasificación

- Identificar un objeto, un fenómeno, un patrón o cualquier cosa desconocida como miembro de una categoría conocida.
  - Identificación de especies de animales o plantas.
- Terminología.
  - Objeto: Aquello que se desconoce y se quiere clasificar.
  - Clase: Categoría en la que se agrupan los objetos que comparten características similares.
  - Atributo: Propiedad que se utiliza para identificar las clases.
  - Característica: Par atributo-valor que se verifica para cierto objeto.
- El proceso de clasificación consiste en equiparar un conjunto de datos con un conjunto de posibles soluciones conocidas de antemano. El resultado puede ser una única clase, un conjunto de clases candidatas o el conjunto vacío.

Clasificación

- Inferencias en la tarea de clasificación.
  - GENERAR: Genera las clases candidatas a ser solución (con o sin información sobre el objeto a clasificar).
  - ESPECIFICAR: Devuelve un atributo cuyo valor (desconocido) será de utilidad para distinguir entre las clases candidatas.
  - OBTENER: Obtiene el valor asociado a un atributo en el objeto a clasificar.
  - EQUIPARAR: Comprueba si una clase candidata tiene una característica.

Método de poda para la tarea de clasificación

while new-solution GENERAR(Objeto → Candidata) **do**
  Clases-candidatas := Candidata **union** Clases-candidatas;
end while
while new-solution ESPECIFICAR(Clases-candidatas → Atributo)
      and length Clases-candidatas > 1 **do**
  OBTENER(Atributo → Valor);
  for-each Candidata **in** Clases-candidatas **do**
    EQUIPARAR(Candidata + Atributo + Valor → Resultado);
    if Resultado = false **then**
      Clases-candidatas := Clases-candidatas **substract** candidata;
    end if
  end for-each
end while

Clasificación

- Observaciones:
  - Los valores de los atributos no siempre son observables.
- Variaciones sobre el algoritmo.
  - Reducir el conjunto de soluciones candidatas, modificando la inferencia GENERAR para que obtenga las clases candidatas teniendo en cuenta un conjunto inicial de datos.
  - Dar libertad al usuario a la hora de introducir los datos, sustituyendo la inferencia ESPECIFICAR por una función de transferencia adecuada.
  - Aprovechar la estructura jerárquica de las categorías, haciendo que la inferencia ESPECIFICAR seleccione atributos que permitan descartar superclases de la jerarquía.

Clasificación

- Elementos del esquema de conocimiento:
  - El conjunto de clases que pueden ser solución al problema.
  - El conjunto de características (pares atributo-valor) que identifica a cada clase.
  - Dependencias entre cada clase y su conjunto de características. Estas dependencias se pueden expresar mediante tipos de reglas.

Valoración

- Identificar un caso como miembro de una categoría de decisión.
  - Se puede considerar un tipo de clasificación en la que sólo hay dos clases candidatas: aceptar y rechazar.
  - Concesión de prestamos bancarios.
- Terminología.
  - Caso: Situación que se quiere valorar.
  - Abstracción: Especificación cualitativa de las características del caso.
  - Criterio: Factor que influye en la valoración.
  - Decisión: Resultado de la valoración, puede ser el valor aceptar o el valor rechazar.
- El proceso de valoración consiste en determinar todos los criterios aplicables a un caso y aplicarlos hasta que nos lleven a una decisión.

Valoración

- Inferencias en la tarea de valoración.
  - ABSTRAER: Obtiene una especificación cualitativa de las características de un caso.
  - ESPECIFICAR: Genera todos los criterios aplicables al caso.
  - SELECCIONAR: Selecciona un criterio de un conjunto.
  - EVALUAR: Evalúa un criterio sobre un caso.
  - EQUIPARAR: Determina una decisión (si es posible) a partir de los valores para un conjunto de criterios.

Valoración

Método para la tarea de valoración

while new-solution ABSTRAER(Caso → Característica) **do**
  Caso := Característica **union** Caso;
end while
ESPECIFICAR(Caso → Criterios);
repeat
  SELECCIONAR(Criterios → Criterio);
  EVALUAR(Caso + Criterio → Valor);
  Resultados := Valor **union** Resultados;
until has-solution EQUIPARAR(Resultados → Decisión);

Valoración

- Observaciones:
  - La tarea de valoración tiene que devolver una decisión.
  - Los criterios de valoración podrían ser inconsistentes.
- Variaciones:
  - La abstración del caso podría no ser necesaria, en este caso se prescinde de la inferencia ABSTRAER.
  - Los criterios pueden ser específicos para las características cualitativas de cada caso.
  - La selección de criterios puede ser aleatoria (no existe conocimiento sobre la selección), heurística (basada en la experiencia de los expertos de dominio) o estadística (se selecciona el elemento más predictivo).

Valoración

- Elementos del esquema de conocimiento:
  - La especificación de las características del caso.
  - El conocimiento de abstracción, que establece relaciones entre características cuantitativas del caso y características cualitativas.
  - El conocimiento de selección de los criterios.
  - El conocimiento de valoración, que establece relaciones entre valores de criterios y los tipos de decisión.

Diagnóstico

- Encontrar un defecto que provoca una disfunción en el sistema.
  - Identificación de problemas en dispositivos estropeados: electrodomésticos, vehículos.
  - Identificación de enfermedades a partir de los síntomas.
- Terminología.
  - Fallo: Características que reflejan la disfunción del sistema.
  - Hipótesis: Posible defecto que provoca una disfunción en el sistema.
  - Diferencial: Conjunto de hipótesis aplicables al fallo.
  - Evidencias: Características observadas en el sistema.
  - Defecto: Resultado del diagnóstico.
- El proceso de diagnóstico consiste en generar el conjunto diferencial formado por todas las hipótesis que pueden explicar la disfunción del sistema e ir eliminando hipótesis mediante la determinación de nuevas evidencias en el sistema.

Diagnóstico

- Inferencias en la tarea de diagnóstico.
  - CUBRIR: Obtiene una hipótesis que puede explicar el fallo.
  - SELECCIONAR: Selecciona una hipótesis del conjunto diferencial.
  - ESPECIFICAR: Determina una característica observable que puede confirmar o refutar la hipótesis.
  - OBTENER: Transferencia que obtiene el valor de una característica observable.
  - VERIFICAR: Comprueba si una hipótesis es válida a partir de un conjunto de valores de características observables.

Método de cobertura causal para la tarea de diagnóstico

while new-solution CUBRIR(Fallo → Hipótesis) **do**
  Diferencial := Hipótesis **union** Diferencial;
end while
repeat
  SELECCIONAR(Diferencial → Hipótesis);
  ESPECIFICAR(Hipótesis → Observable);
  OBTENER(Observable → Hallazgo);
  Evidencia := Hallazgo **union** Evidencia;
  foreach Hipótesis **in** Diferencial **do**
    VERIFICAR(Hipótesis + Evidencia → Resultado);
    if Resultado = false **then**
      Diferencial := Diferencial **substract** Hipótesis;
    end if
  end foreach
until length Diferencial ≤ 1 **or** "no hay más características observables";
Defectos := Hipótesis;

Diagnóstico

- Observaciones:
  - Si el modelo causal consiste en un conjunto de asociaciones directas entre fallos y defectos, entonces la tarea de diagnóstico se reduce a una tarea de clasificación.
- Variaciones:
  - La construcción del diferencial puede ser progresiva en función de las probabilidades de las hipótesis.
  - Se pueden añadir una nueva inferencia que realice una caracterización cualitativa de los fallos y hallazgos recopilados.
  - Se puede añadir un método de simulación que obtenga los hallazgos esperados para cada hipótesis de forma que estos se puedan utilizar para descartar hipótesis.
  - El grado de relación entre las evidencias y las hipótesis puede dar lugar a una reconsideración de hipótesis previamente eliminadas.

Diagnóstico

- Elementos del esquema de conocimiento.
  - Las características del sistema a diagnosticar: observables, internas, fallos.
  - El conocimiento causal que asocia los defectos del sistema con los valores de sus caraterísticas e indica el grado de asociación (probabilidad de que se den dichos valores).
  - Graduación de las hipótesis: frecuencia, gravedad.
  - Graduación de las características observables: tiempo de obtención de respuesta, fiabilidad de la respuesta.

Monitorización

- Analizar un proceso en funcionamiento para detectar si se comporta según las espectativas.
  - Seguimiento de un proyecto software.
  - Monitorización de una planta industrial.
- Terminología.
  - Parámetro: Dato relevante para el seguimiento del funcionamiento del sistema.
  - Norma: Valor esperado de un parámetro en el caso de buen funcionamiento.
  - Discrepancias: Características que indican que el sistema está funcionando mal.
  - Historial: Conjunto de datos recopilados en ciclos de monitorización previos.

Monitorización

- El proceso de monitorización consiste en un proceso cíclico que se repite para cada conjunto de valores de los parámetros. En este proceso se comparan los valores de los parámetros con la norma y se genera una descripción de la diferencia. Esta diferencia se clasifica o no como discrepancia teniendo en cuenta el historial de valores para los parámetros.

Monitorización

- Inferencias en la tarea de monitorización.
  - Recibir: Función de transferencia a través de la que se reciben los valores de los parámetros.
  - Seleccionar: Selecciona un parámetro para analizar.
  - Especificar: Obtiene el valor de la norma para un parámetro.
  - Comparar: Determina la diferencia entre el valor real de un parámetro y el valor de su norma.
  - Clasificar: Identifica una discrepancia a partir de los historiales de los parámetros.

Método para la tarea de monitorización

repeat
  RECIBIR(Hallazgo);
  while new-solution SELECCIONAR(Hallazgo → Parámetro) **do**
    ESPECIFICAR(Parámetro → Norma);
    COMPARAR(Norma + Hallazgo → Diferencia);
    Diferencias := Diferencia **add** Diferencias;
    Historial := Hallazgo **add** Historial;
  end while
  CLASIFICAR(Diferencias + Historial → Discrepancia);
forever

Monitorización

- Observaciones:
  - La salida del método es la discrepancia, sin justificación del fallo que la produce.
  - La salida de la tarea de monitorización se puede conectar con una tarea de diagnóstico para identificar el fallo.
- Variaciones:
  - Algunas veces la inferencia CLASIFICAR puede ser muy compleja y es necesario llevarla a cabo mediante una tarea de clasificación.
  - La transferencia RECIBIR se puede sustituir por OBTENER, en la que la iniciativa la toma el sistema.

Monitorización

- Elementos del esquema del dominio:
  - Descripción de las características del sistema.
  - Identificación de los parámetros y sus normas.
  - Relaciones entre las diferencias, el historial y las discrepancias.

Predicción

- Analizar el comportamiento de un sistema en un momento dado, para construir una descripción del estado que tendrá en un futuro.
  - Predicción meteorológica.
- Es una tarea analítica que presenta características de las tareas sintéticas.

Configuración

- Encontrar una combinación de componentes que se ajuste a un conjunto de requisitos y restricciones.
  - Configuración de sistemas hardware.
- Terminología.
  - Componente: Una de las partes a combinar.
  - Parámetro: Una característica de un componente o de una combinación de éstos.
  - Restriciones: Factores que limitan la elección de los componentes o el valor de los paramétros.
  - Preferencias: Factores que indican el diseño que se desea obtener.
  - Requisitos: Factores que expresan las necesidades y preferencias de los usuarios del sistema.

Configuración

- La tarea de configuración consiste en:
  - Convertir el conjunto de requisitos iniciales en un conjunto de restricciones y preferencias.
  - Especificar uno o varios formatos iniciales para el diseño acordes con los requisitos iniciales.
  - Proponer extensiones de los diseños considerados a partir de las opciones para las componentes de diseño y las preferencias.
  - Verificar la consistencia de cada configuración propuesta, teniendo en cuenta las restricciones dadas.
  - Si la configuración es inconsistente, generar un conjunto de acciones a llevar a cabo para corregir la inconsistencia y modificar el diseño de acuerdo con estas acciones.

Configuración

- Inferencias en la tarea de configuración.
  - TRASLADAR: Transforma los requisitos iniciales en un conjunto de restricciones y preferencias.
  - ESPECIFICAR: Genera un diseño inicial a partir de los requisitos.
  - PROPONER: Propone una extensión del diseño considerado a partir de las preferencias.
  - VERIFICAR: Valora la adecuación del diseño con respecto al conjunto de restricciones.
  - CRITICAR: Genera un conjunto de acciones para modificar el diseño.
  - SELECCIONAR: Selecciona una acción.
  - MODIFICAR: Modifica un diseño de acuerdo con una acción.

Método para la tarea de configuración

TRASLADAR(Requisitos → Restricciones + Preferencias);
ESPECIFICAR(Restricciones → DiseñoInicial);
while new-solution PROPONER(DiseñoInicial + Preferencias → Extension) **do**;
  Diseño := Extensión **union** DiseñoInicial;
  VERIFICAR(Diseño + Restricciones → Valor + Violación);
  if Valor = false **then**
    CRITICAR(Violación + Diseño → Acciones);
    repeat
      SELECCIONAR(Acciones → Acción);
      MODIFICAR(Diseño + Acción → Diseño);
      VERIFICAR(Diseño + Restricciones → Valor + Violación);
    until Valor = true;
  end if
end while

Configuración

- Observaciones:
  - El algoritmo puede utilizarse para obtener un conjunto de diseños posibles.
- Variaciones:
  - Llevar a cabo las fases de verificación y revisión sólo cuando se propone una componente para todos los elementos del diseño.
  - Evitar el conocimiento sobre modificaciones y usar vuelta atrás cronológica (backtracking).

Configuración

- Elementos del esquema de dominio.
  - Elementos del diseño: componentes y parámetros que los caracterizan.
  - Dependencias entre elementos del diseño.
  - Restricciones sobre componentes: incompatibilidades.
  - Preferencias sobre componentes.
  - Acciones para evitar violaciones de restricciones.

Asignación

- Crear una correspondencia entre dos conjuntos de objetos.
  - Asignación de puertas de embarque a aviones.
- Terminología.
  - Objeto: Elementos a los que se les va asignar algún recurso.
  - Recurso: Elementos que se van a asignar a algún objeto.
  - Reparto: Asociación entre un objeto y un recurso.
- El proceso de asignación consiste en seleccionar un subgrupo de objetos a los que se les asigna un recurso. Este proceso se repite hasta que todos los objetos tengan un recurso asignado.

Asignación

- Inferencias en la tarea de asignación.
  - AGRUPAR: Selecciona un grupo de objetos a los que asignar un recurso común.
  - ASIGNAR: Asigna un recurso disponible a un grupo de objetos.

Asignación

Método para la tarea de asignación

while not empty Objetos **do**
  AGRUPAR(Objetos $\rightarrow$ Grupo);
  ASIGNAR(Grupo + Recursos + Repartos $\rightarrow$ Recurso);
  Repartos := $\langle$ Grupo, Recurso $\rangle$ **union** Recursos;
  Objetos := Objetos **delete** Grupo;
  Recursos := Recursos **delete** Recurso;
end while

Asignación

- Variaciones:
  - Algunas veces existen asignaciones previas a la ejecución del método.
  - Puede ser necesario añadir restricciones y preferencias específicas en casos particulares.

Programación temporal

- Dado un conjunto de trabajos predefinidos consistentes en la realización secuencial de una serie de tareas, la programación temporal asigna recursos en intervalos de tiempo a todas las tareas.
  - Programación de la producción en fábricas.
- Terminología.
  - Trabajo: Secuencia temporal de actividades simples.
  - Unidad: Actividad simple que precisa de un recurso para desarrollarse.
  - Recurso: Elemento necesario para desarrollar actividades simples.
  - Restricciones: Condiciones a verificar cuando se asignan unidades a recursos.
  - Programa: Programación temporal de trabajos.

Programación temporal

- El proceso de programación temporal consiste en especificar un programa inicial e iterar el siguiente proceso.
  - Seleccionar una unidad para ser asignada.
  - Seleccionar un recurso para dicha unidad.
  - Asignar la unidad al recurso seleccionado.
  - Evaluar el programa actual.
  - Modificar el programa si fuera necesario.

Programación temporal

- Inferencias en la tarea de programación temporal.
  - ESPECIFICAR: Genera un programa inicial.
  - SELECCIONAR: Selecciona una unidad participante en un programa.
  - BUSCAR: Busca un recurso para asignar a una unidad.
  - ASIGNAR: Asigna un recurso a una unidad.
  - EVALUAR: Evalúa un programa.
  - MODIFICAR: Modifica un programa.

Programación temporal

Método para la tarea de programación temporal

ESPECIFICAR(Trabajos → Programa);
while new-solution SELECCIONAR(Programa → Unidad) **do**
  BUSCAR(Unidad + Programa → Recurso);
  ASIGNAR(Unidad + Recurso → Programa);
  EVALUAR(Programa → Resultado);
  if Resultado = false **then**
    MODIFICAR(Programa → Programa);
  end if
end while

Programación temporal

- Existen dos métodos complementarios para la programación temporal.
  - Métodos constructivos: van ampliando incrementalmente programas parciales hasta que se alcanza uno completo.
  - Métodos de reparación: comienzan con un programa completo y lo van modificando iterativamente.

Programación temporal

- Elementos del esquema de dominio:
  - Asociaciones entre un recurso y grupos de unidades.
  - Trabajos que contienen varias unidades temporalmente ordenadas.
  - Programaciones temporales compuestas de varios trabajos y recursos.

Sintesis

- Construir una descripción de un sistema que verifique un conjunto dado de requisitos.
- El método analiza y traslada los requisitos iniciales en dos tipos de requisitos: los necesarios y los deseables. En las siguientes etapas se generan todas las estructuras de sistema posibles (cumpliendo los requisitos necesarios) y selecciona un subconjunto de estructuras válidas (cumpliendo los requisitos deseables). Finalmente el método ordena las estructuras válidas de acuerdo con las preferencias.

Bibliografía

- Schreiber G., Akkermans H., ...
  "Knowledge Engineering and Management: The CommonKADS Methodology", The MIT Press, 1999.
  - Cap. 6: "Template Knowledge Models"
- Alonso A., Guijarro B., ...
  "Ingeniería del Conocimiento: Aspectos Metodológicos", Pearson Prentice Hall, 2004.
  - Cap. 6: "Librería de tareas y métodos de resolución de problemas"