TAREA 2

ROBIN EMMANUEL CARLOS GONZALEZ 21110427

1) Adquisición de conocimiento

1.1 Experto del dominio

- Qué: Fuente primaria del “saber-hacer”: reglas, atajos, excepciones, umbrales.
- Para qué: Asegurar que el sistema refleja prácticas reales y casos límite.
- Cómo (técnicas de elicitación):
    - Entrevista estructurada/semiestructurada (guiones, matrices de decisión).
    - Think-aloud (el experto resuelve casos y verbaliza criterios).
    - Card sorting / repertory grid (organizar conceptos y atributos).
    - Análisis de casos históricos (expedientes, reportes de incidentes).
    - Protocolos de validación: revisión de reglas por el experto, sesiones de “red team” con casos trampa.

1.2 Cognimático / Ingeniero de conocimiento

- Qué: Perfil puente entre experto y sistema (modela y formaliza).
- Para qué: Convertir conocimiento tácito → representaciones computables; evitar ambigüedades.
- Cómo:
    - Normalización de términos (glosario, sinónimos, ontología).
    - Desambiguación de reglas (precondiciones, contexto, excepciones).
    - Priorización (conflict set: especificidad > prioridad del experto > recencia).
    - Versionado (control de cambios, trazabilidad de quién/por qué cambió una regla).

1.3 Sensores

- Qué: Fuentes físicas (temperatura, vibración, imagen, ECG, etc.).
- Para qué: Inyectar hechos observables en tiempo real.
- Cómo:
    - Preprocesamiento (filtrado, calibración, umbrales, detección de outliers).
    - Transformación a hechos (p.ej., Temp>38.5 → FiebreAlta=Sí).
    - Gestión de incertidumbre: intervalos, etiquetas difusas (baja/media/alta).

1.4 Bases de datos

- Qué: Repositorios estructurados (SQL/NoSQL, historiales).
- Para qué: Proveer contexto y antecedentes.
- Cómo:
    - ETL a la Base de Hechos (BH): mapeo de columnas → hechos.
    - Integridad (uniqueness, referential integrity), calidad de datos (valores faltantes, duplicados).

1.5 Módulo de adquisición de conocimiento

- Qué: Herramientas para crear/editar/verificar reglas y conceptos.
- Para qué: Mantener la Base de Conocimiento (BC) viva.
- Cómo:
    - Editores de reglas con pruebas unitarias por regla.

    - Extractores (text mining de manuales/guías; aprendizaje de reglas propuestas para revisión humana).

    - Workflows de revisión: borrador → revisión experto → validación QA → producción.

2) Representación del conocimiento

2.1 Base de Conocimiento (BC)

- Qué: Reglas, ontologías, marcos, taxonomías, funciones de pertenencia difusa.

- Para qué: Capturar tanto declarativo (definiciones, relaciones) como procedimental (si-entonces, planes).

- Cómo (formatos comunes):

    - Reglas de producción: SI (Premisas) ENTONCES (Conclusión/Acción, Peso, Explicación).

    - Ontologías (OWL/RDF): clases, propiedades, relaciones “es-un”, “parte-de”.

    - Frames / objetos: atributos con valores por defecto y herencia.

    - Lógica difusa: conjuntos y reglas con conectores difusos (min, max, t-normas).

- Buenas prácticas: granularidad fina, nombres canónicos, justificación adjunta (“rationale”), pruebas de cobertura de reglas.

2.2 Base de Hechos (BH)

- Qué: Memoria de trabajo del caso (hechos actuales + derivados).

- Para qué: Contexto vivo para disparo de reglas.

- Cómo:
    - Estructura: pares atributo-valor, hechos temporales, sellos de tiempo.
    - Ciclo de vida: alta (entrada), actualización (por reglas), decaimiento/expiración (hechos obsoletos).
    - Gestión de conflicto: cuando varias reglas compiten por modificar el mismo hecho (prioridades, locks).

3) Tratamiento del conocimiento

3.1 Motor de Inferencia

- Qué: Núcleo de razonamiento.
- Para qué: Aplicar BC sobre BH para diagnosticar, clasificar, planificar o recomendar.
- Cómo (estrategias):
    - Encadenamiento hacia adelante (data-driven): sensores/entradas → disparo de reglas → conclusiones.
    - Encadenamiento hacia atrás (goal-driven): meta → sub-metas → pruebas de premisas → confirmación.
    - Resolución de conflictos:
        - Especificidad (regla más específica gana).
        - Recencia (hechos más recientes pesan más).
        - Prioridad (peso del experto o criticidad).
    - Incertidumbre:

        - Factores de certeza (MYCIN-like), probabilidad bayesiana, Dempster-Shafer, lógica difusa.

    - Eficiencia: indexado de premisas, Rete/Rete-OO para matching de patrones, memoria parcial de coincidencias.

3.2 Módulo de Explicaciones

- Qué: Trazabilidad del razonamiento.
- Para qué: Confianza, auditoría, cumplimiento normativo, docencia.
- Cómo:
    - ¿Por qué? (reglas y hechos que llevaron a la conclusión).
    - ¿Por qué no? (premisas que fallaron, umbrales no alcanzados).
    - ¿Qué pasaría si…? (análisis contrafactual modificando hechos).
    - Visualización: árboles de decisión activados, cronología de disparos, pesos/certezas.

4) Interfase y utilización

4.1 Interfase (UI/API)

- Qué: Punto de contacto con personas/sistemas.
- Para qué: Capturar entradas de forma guiada y mostrar salidas comprensibles.
- Cómo:
    - UI: formularios con validaciones, wizards por caso, paneles de explicación.
    - API: endpoints para integración (p.ej., ERP, MES, HIS).

- Diseño centrado en el usuario: vocabulario del dominio, estados claros, accesibilidad, logs de interacción.

4.2 Usuario

- Qué: Operador/analista/cliente final.
- Para qué: Tomar decisiones y retroalimentar el sistema.
- Cómo:
  - Proveer datos, revisar recomendaciones, solicitar explicación, aceptar/rechazar acciones, reportar falsos positivos/negativos.

5) Flujo end-to-end (operación)

1. Datos (usuario/sensores/BD) → BH.
2. Motor evalúa premisas con BC (Rete / matching).
3. Disparo de reglas → nuevos hechos/hipótesis/acciones con pesos.
4. Resolución de conflictos → se aplican acciones priorizadas.
5. Explicación registra trazas y responde “por qué/por qué no/qué-si”.
6. Salida en la interfaz + posible escritura en BD externas.
7. Retroalimentación del usuario (acepta/rechaza) → datos para mejora.
8. Mantenimiento: nuevas reglas, ajustes de umbrales, refactor de ontología.

6) Ejemplo concreto (mini-caso)

- Dominio: Mantenimiento predictivo de motores.
- Hechos iniciales (BH): Temp=92°C, Vibración_RMS=7.1 mm/s, Horas=4,900, Ruido_Espectral_Pico=1.6 kHz.
- Reglas (BC):
    - R1: *SI* Temp>90 Y Vibración>7 → Riesgo_Rodamiento=Alto (CF 0.7).
    - R2: *SI* Pico_espectral≈1.5–1.7 kHz → Falla_probable=Desgaste_Rodamiento (CF 0.6).
    - R3: *SI* Horas>4,500 Y Riesgo_Rodamiento=Alto → Plan=Paro_en_24h.
- Inferencia (adelante): R1 y R2 disparan; combinando CF → hipótesis fuerte; R3 agenda acción.
- Explicación: “Se recomendó Paro en 24h porque R1 y R2 se cumplieron con datos de hoy; R3 aplica por horas acumuladas”.

7) Validación, pruebas y operación continua

- Tipos de prueba:
    - Unitarias de reglas (cada regla con casos positivos/negativos).
    - Conjuntos de regresión (evitar que cambios rompan casos antes correctos).
    - Revisión por pares (otro experto valida la redacción y alcance).
    - A/B con operadores (comparar desempeño asistido vs. manual).
- Métricas:
    - Exactitud / Precisión / Recall / F1 (si hay verdad-terreno).

    - Tasa de aceptación del usuario, tiempo de caso, nº de explicaciones consultadas, nº de overrides.

    - Cobertura de reglas (porcentaje de casos que activan al menos 1 regla relevante).

- Mantenimiento:

    - Gestión de ciclo de vida (deprecar reglas, versionar ontología).

    - Monitoreo de drift (cambian equipos, procedimientos, perfiles de pacientes).

    - Panel de salud del sistema: top reglas activadas, conflictos frecuentes, hechos sin definición.

8) Errores comunes y cómo evitarlos

- Reglas demasiado genéricas → muchos falsos positivos. *Mitigar:* añadir condiciones específicas, umbrales y contexto.

- Solapamiento de reglas sin prioridad clara → decisiones inconsistentes. *Mitigar:* política de conflicto explícita.

- Glosarios inconsistentes (sinónimos, unidades). *Mitigar:* ontología central y validaciones de unidad.

- **Falta de trazabilidad** → auditoría imposible. *Mitigar:* módulo de explicaciones obligatorio y logs firmados.

- **No involucrar al usuario final** en el diseño → baja adopción. *Mitigar:* prototipado con operadores, métricas de usabilidad.