

Desarrollo de Sistemas Inteligentes

ALERTAS ECG: Informe 2

Convocatoria de Enero
Curso académico 2021/22



LosmonTapuercas

F. Aguilar Martínez
francisco.aguilarm@um.es
Grupo PCEO

J.A. Lorencio Abril
joseantonio.lorencioa@um.es
Grupo PCEO

R. Gaspar Marco
ruben.gasparm@um.es
Grupo PCEO

Profesor:
Javier Gomez Marín-Blázquez

Versión: 1.0
Fecha: 19 de noviembre de 2021

Índice general

Introducción	2
1 Creación de la Base de Hechos	2
2 Creación de la Base de Conocimiento	2
3 Inicialización de la Base de Hechos	2
4 Ejecución del SBR	3
5 Manual de Uso	3

Introducción

En este informe detallamos el desarrollo del Sistema Basado en Reglas que constituye parte del proyecto final de la asignatura “Desarrollo de Sistemas Inteligentes”. El objetivo de la práctica es desarrollar un prototipo de un sistema de alarmas que identifique, a partir de la interpretación de un electrocardiograma (ECG o EKG), posibles patrones de riesgo para la salud.

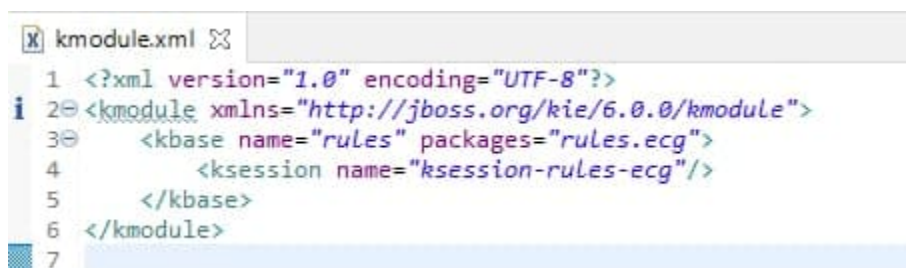
El programa tendrá como *input* un archivo de log de ECG a partir del cual se construye la Base de Hechos inicial. También contará con un motor de reglas que permitirá modificar la base de hechos hasta alcanzar un diagnóstico.

1. Creación de la Base de Hechos

Para poder trabajar con Drools sin complicaciones derivadas de integrar Protégé con el proyecto, hemos introducido la estructura de clases de la ontología en un proyecto Drools. La clase *MainClass.java* es la que contiene el método `main()`, y que detallaremos más adelante.

2. Creación de la Base de Conocimiento

Modificamos el archivo *kmodule.xml* para configurarlo conforme a nuestras necesidades.



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <kmodule xmlns="http://jboss.org/kie/6.0.0/kmodule">
3   <kbase name="rules" packages="rules.ecg">
4     <ksession name="ksession-rules-ecg"/>
5   </kbase>
6 </kmodule>
7
```

Figura 1: Fichero *kmodule.xml*

Además, creamos un paquete *rules.ecg* que contendrá el archivo de reglas *ReglasECG.drl*, en el que definimos las reglas con las que se crearán los distintos componentes del ECG.

3. Inicialización de la Base de Hechos

En la clase *MainClass.java* hemos desarrollado un *parser* para los archivos log de los ECG usando expresiones regulares (ER). Las ER utilizadas son sencillas, aunque podemos destacar la utilizada para capturar las ondas:

```
1 private static String onda = "([PQRST])\\((([0-9]+),([0-9]+),(-?[0-9]+.[0-9]+)\\))";
```

Listado 1: Regex para capturar ondas

Como vemos, la ER nos permite conocer el tipo de onda capturando la primera letra, además de las características que la definen. De esta forma, somos capaces de crear las ondas e introducirlas en la base de hechos.

4. Ejecución del SBR

Hemos dividido la agenda de reglas en cuatro grupos: *Inicialización*, *DeteccionIndicios*, *DeteccionEnfermedades* e *Impresión*. Una vez parseado el documento y creadas las instancias de las ondas detectadas en el fichero, lanzamos las reglas de inicialización, que construyen los demás componentes del ECG a partir de las ondas. Tras finalizar esta fase, se lanzan las reglas de detección de indicios, las cuales detectan y generan las instancias de indicios de diagnóstico. Finalmente se lanzan las reglas que detectarán las enfermedades a partir de los indicios de diagnóstico previamente generados. La relación entre indicios de diagnóstico y las distintas enfermedades puede verse en las reglas de detección de enfermedades, y la bibliografía usada para su desarrollo puede consultarse al final de esta memoria.

Cabe destacar la forma en la que detectamos la sanidad, esto lo hacemos con la regla:

```
1 rule "Sano"
2 agenda-group "DeteccionEnfermedades" salience -1
3 when
4     not (exists (Diagnostico()))
5 then
6     insert (new Sano(0));
7 end
```

Listado 2: Regla de detección de un paciente sano

Podemos observar que le asignamos prioridad -1, de esta forma nos aseguramos de que se ejecute en último lugar, asegurándonos así su correcto funcionamiento.

Se concluye con la regla de impresión, que obtiene la instancia de enfermedad más prontamente registrada de cada enfermedad encontrada en el ECG y muestra la información sobre su diagnóstico en pantalla.

```
1 rule "Impresion"
2 agenda-group "Impresion"
3 when
4     $d: Diagnostico()
5     not (Diagnostico(getClass() == $d.getClass() , ciclo < $d.getCiclo()))
6 then
7     System.out.println($d.toString());
8 end
```

Listado 3: Obtención de las enfermedades detectadas

5. Manual de Uso

El programa es muy sencillo de utilizar. Simplemente debemos ejecutar el programa, este nos preguntará por la ruta del fichero .ecg que deseamos analizar. Debemos introducir la ruta completa, incluyendo la extensión del archivo.

Tras introducir la ruta correctamente, el programa correrá y, cuando termine, nos mostrará por pantalla los resultados obtenidos, no es necesario hacer nada más.

Bibliografía

- [1] Life in the Fast Lane. *QT Interval*. URL: <https://litfl.com/qt-interval-ecg-library/>. (accessed: 12/11/2021).
- [2] Life in the Fast Lane. *T wave*. URL: <https://litfl.com/t-wave-ecg-library/>. (accessed: 12/11/2021).
- [3] Bioelectromagnetism Portal. *The Basis of ECG Diagnosis*. URL: <http://www.bem.fi/book/19/19.htm>. (accessed: 12/11/2021).
- [4] Wikipedia. *Some pathological patterns which can be seen on the ECG*. URL: https://en.wikipedia.org/w/index.php?title=Electrocardiography&oldid=513556137#Some_pathological_patterns_which_can_be_seen_on_the_ECG. (accessed: 12/11/2021).
- [5] Wikipedia. *T wave*. URL: https://en.wikipedia.org/wiki/T_wave. (accessed: 12/11/2021).