SABI2020
22 CONGRESO DE BIOINGENIERÍA | 11 JORNADA DE INGENIERÍA CLÍNICA
4-6 MARZO | PIRIAPOLIS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY · nib · crmf - UdelaR · UNER

High-Level Language to Specify an Adaptive Heart Failure Follow up Strategy

Rodrigo Olivares[1], Gabriela Silvera[2], Gabriela Ormaechea[2], Pedro Álvarez-Rochay[2], Carla Taramasco[1] and Franco Simini[3]

[1]*Escuela de Ingeniería Civil Informática Universidad de Valparaíso Valparaíso, Chile*
[2]*Facultad de Medicina Universidad de la República Montevideo, Uruguay*
[3]*Núcleo de Ingeniería Biomédica (NIB), de las Facultades de Medicina e Ingeniería, Universidad de la República, Montevideo, Uruguay*

MAIN CONTRIBUTION

To implement high-level procedures using a modern programming language in order to specify adaptive heart failure follow up strategies on the Heart Failure Management Information System (SIMIC).

ABSTRACT

The **HEART FAILURE MANAGEMENT INFORMATION SYSTEM** (SIMIC for the Spanish acronym) includes an App which interacts with the patient, putting in practice the data capture and context-adapted feedback messages. During the following visit, **SIMIC-web** displays the information recorded since the previous visit, to be saved as **ELECTRONIC CLINICAL RECORD** by the physician. The specification of Clinical Guidelines and messaging to the patient need a formal language, which is described in this paper. We use Production Rules to describe unequivocally the actions to be taken autonomously by **SIMIC**, in a way similar to **EXPERT SYSTEMS**. The syntax described includes the use of standard logical operators that can be parsed by SIMIC and executed as a real time system. SIMIC operates according to a standard clinical seriousness scale. Four rules are given referring to increased and reduced body weight combined with no shows at medical visits, resulting in either a message to the patient and / or an alarm dispatched to **THE HEALTH CARE SYSTEM**.

EXPERT SYSTEMS

Following Crosan and Abraham [1], an **EXPERT SYSTEMS** produces a useful output based on formal rules, which are the representation of knowledge. Figure 1 shows a simplified architecture of an **ES**, and its elements are the following: [1][2][3]:

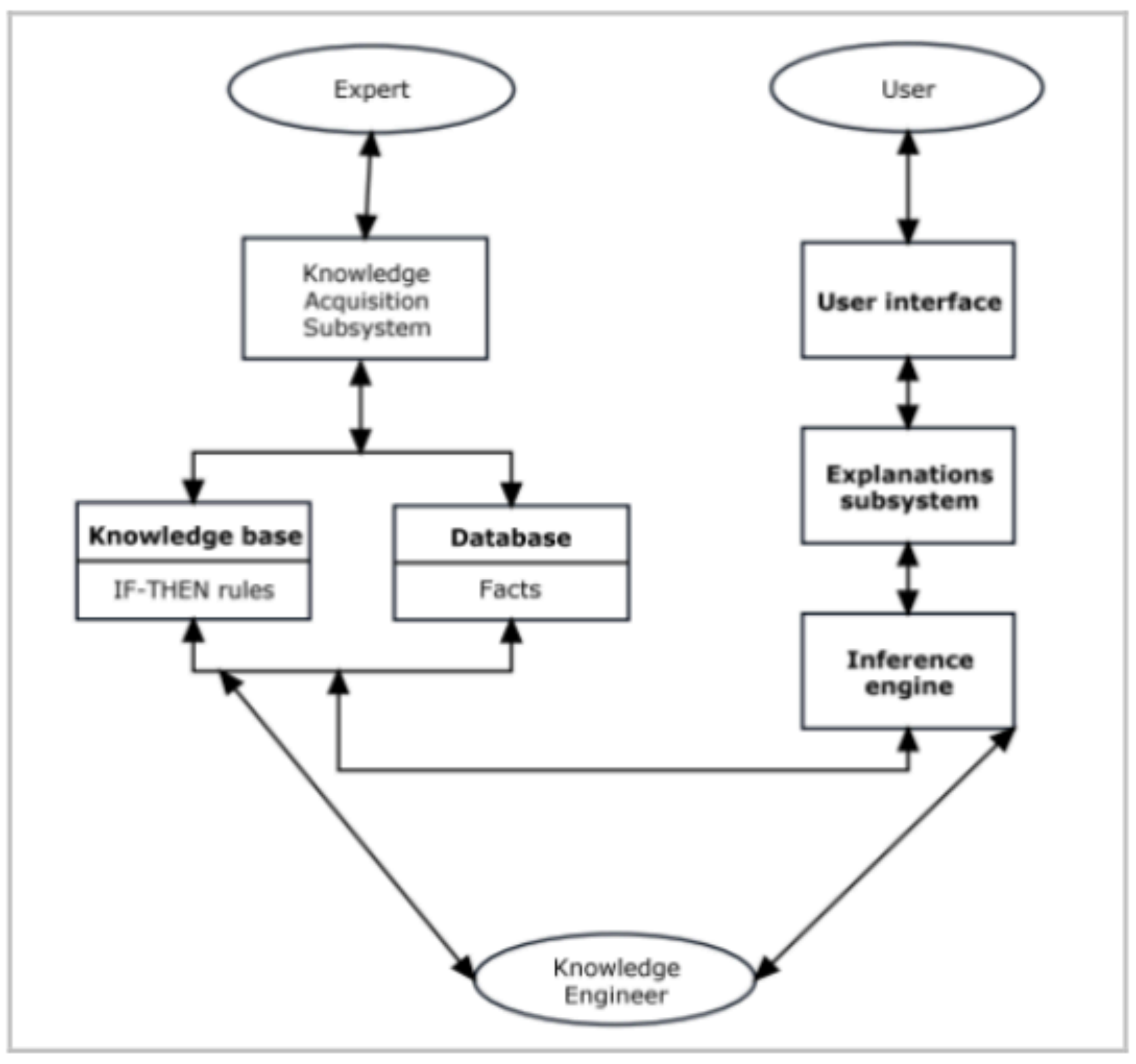


Figura 1. Generic architecture of an expert system based on formal rules, taken from Crosan et al [8].

HEART FAILURE patients are described by a set of variables such as age, clinical stage of HF (mild, intermediate, severe), suggested home and/or outpatient visit periodicity.

Let **GM** be the suffix of all rules producing an alert or a message for the Health Care Team, and let **GP** be the suffix for rules producing either an alert or a data request to the Patient. Let *Fi* denote events where *i* = *{1,...,n}* spans all possible events up to a total of *n*. Production rules are named *Rj* for *j* = *{1,...,m}* where *m* is the total number of rules.

GM.F2: (Weight out of range) Alerts the Health Care Team of a weight variation away from the ideal range. Let PP be the previous & and PA actual weight to be compared to the clinically set range PS*min* to PS*max*.

```
R1: M.Alert is FALSE
R2: IF (P.PA > P.PP) is TRUE
AND (P.PA > P.PSmax) is TRUE
THEN M.Alert is TRUE
R3: IF (P.PA < P.PP) is TRUE
AND (P.PA < P.PSmin) es TRUE
THEN M.Alert is TRUE
```

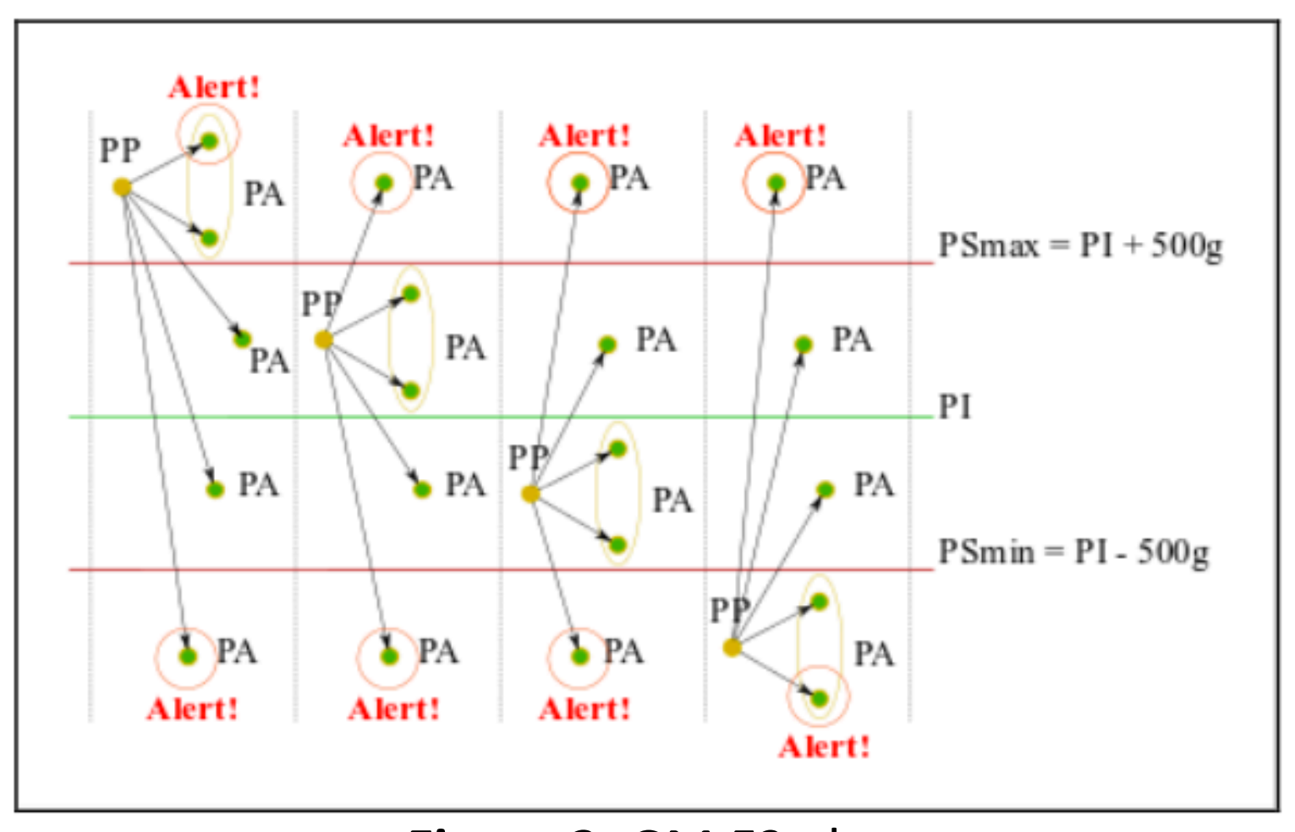
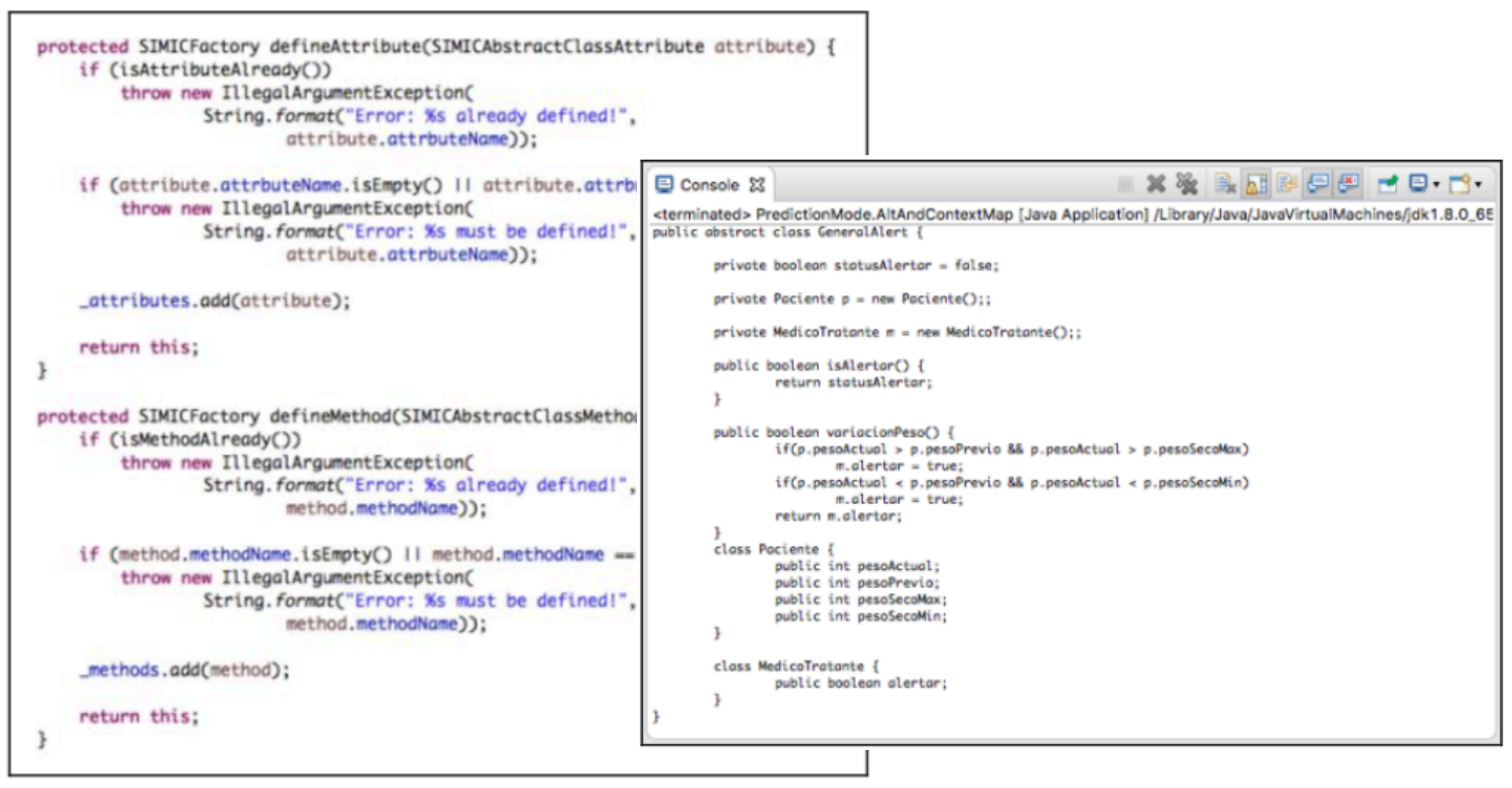


Figura 2. GM.F2 alerts.

IMPLEMENTATIONS

The rules were implemented with **ANTLR** 4 and Eclipse IDE Oxigen 2 © using **JAVA**. After defining the **ES** tokens, syntax and semantic rules were defined, before coding the source code generator.



```
protected SIMICFactory defineAttribute(SIMICAbstractClassAttribute attribute) {
    if (isAttributeAlready())
        throw new IllegalArgumentException(
                String.format("Error: %s already defined!",
                        attribute.attrbuteName));

    if (attribute.attrbuteName.isEmpty() || attribute.attrb
        throw new IllegalArgumentException(
                String.format("Error: %s must be defined!",
                        attribute.attrbuteName));

    _attributes.add(attribute);

    return this;
}

protected SIMICFactory defineMethod(SIMICAbstractClassMetho
    if (isMethodAlready())
        throw new IllegalArgumentException(
                String.format("Error: %s already defined!",
                        method.methodName));

    if (method.methodName.isEmpty() || method.methodName ==
        throw new IllegalArgumentException(
                String.format("Error: %s must be defined!",
                        method.methodName));

    _methods.add(method);

    return this;
}
```

```
<terminated> PredictionMode.AltAndContextMap [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_65
public abstract class GeneralAlert {

        private boolean statusAlertar = false;

        private Paciente p = new Paciente();;

        private MedicoTratante m = new MedicoTratante();;

        public boolean isAlertar() {
                return statusAlertar;
        }

        public boolean variacionPeso() {
                if(p.pesoActual > p.pesoPrevio && p.pesoActual > p.pesoSecoMax)
                        m.alertar = true;
                if(p.pesoActual < p.pesoPrevio && p.pesoActual < p.pesoSecoMin)
                        m.alertar = true;
                return m.alertar;
        }
        class Paciente {
                public int pesoActual;
                public int pesoPrevio;
                public int pesoSecoMax;
                public int pesoSecoMin;
        }

        class MedicoTratante {
                public boolean alertar;
        }
}
```

Figura 3. Previous validation before the code generation process and generated code for GM.F2.

The last step-in high-level language is automatic code generation. For the SIMIC App to include new clinical guidelines knowledge with no mobile App programming skills, code is created by our code generator. The source code we developed is compatible with **OBJECT ORIENTED PROGRAMMING** and consists of **JAVA CLASSES**.

CONCLUSIÓN

SIMIC behaves like an assistant issuing simple texts, at the same time collecting every day home data entered by the patient. It is expected that this intimate interaction will have a positive effect on the follow up eventually reducing the number of new hospitalizations and difficult moments for patient & family.

Referencias

[1] C. Grosan, A. Abraham, (2011), Rule-Based Expert Systems, Springer Berlin Heidelberg, p. 149-185, ISBN 978-3-642-21003-7
[2] B. Buchanan, R. Duda, (1983), Principles of Rule-Based Expert Systems, Advances in Computers, Elsevier, p. 163-216, DOI 10.1016/s0065-2458(08)60129-1.
[3] D. Dominik, D. Reidenbach, (2013), Inferring descriptive generalisations of formal languages, Journal of Computer and System Sciences, Elsevier, 79(5), p. 622-639, DOI 10.1016/j.jcss.2012.10.001.