



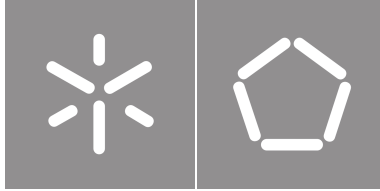
**Universidade do Minho**

Escola de Engenharia

Sarah Tifany da Silva

**Sistema de apoio à decisão clínica  
baseado em regras utilizando  
o modelo openEHR**





**Universidade do Minho**

Escola de Engenharia

Sarah Tifany da Silva

**Sistema de apoio à decisão clínica  
baseado em regras utilizando  
o modelo openEHR**

Dissertação de Mestrado

Mestrado Integrado em Engenharia Informática

Trabalho efetuado sob a orientação de:

**José Manuel Ferreira Machado**

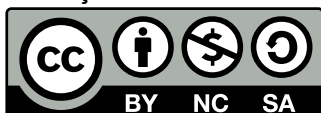
## **DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS**

Este é um trabalho académico que pode ser utilizado por terceiros desde que respeitadas as regras e boas práticas internacionalmente aceites, no que concerne aos direitos de autor e direitos conexos.

Assim, o presente trabalho pode ser utilizado nos termos previstos na licença abaixo indicada.

Caso o utilizador necessite de permissão para poder fazer um uso do trabalho em condições não previstas no licenciamento indicado, deverá contactar o autor, através do RepositoriUM da Universidade do Minho.

### ***Licença concedida aos utilizadores deste trabalho***



**Creative Commons Atribuição-NãoComercial-Compartilhalgal 4.0 Internacional**  
**CC BY-NC-SA 4.0**

<https://creativecommons.org/licenses/by-nc-sa/4.0/deed.pt>

# Agradecimentos

Gostaria de agradecer a todos as pessoas que estiveram presentes e que me acompanharam durante todo este percurso acadêmico.

À minha família que mesmo nos momentos mais difíceis sempre apoiou incondicionalmente as minhas decisões e me motivou a seguir os meus sonhos. Sem eles, nada disto seria possível.

Ao meu orientador e professor José Machado, que me acompanhou ao longo destes últimos dois anos e que me guiou na escolha do melhor tema, apresentando-me à equipa Algoritmi da Universidade do Minho, abrindo, assim, portas para um acompanhamento mais cuidado.

À Francini Hak um especial obrigado por estar sempre disposta a ajudar e por estar sempre presente ao longo deste ano letivo. Sem o seu conhecimento, ajuda e motivação este projeto não teria sido conseguido.

Aos meus amigos que me acompanharam e motivaram durante todo o percurso acadêmico.

Ao meu namorado, Guilherme Martins, quero agradecer por me apoiar ao longo deste percurso e por disponibilizar o seu tempo para a revisão deste relatório.

A todos que estiveram presentes quero agradecer do fundo do meu coração. Sem vocês não teria conseguido.

### DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho acadêmico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Braga, 29 Dezembro 2021  
(Local) (Data)

Sarah Tiffany Silva  
(Sarah Tiffany da Silva)

## **Sistema de apoio à decisão clínica baseado em regras utilizando o modelo openEHR**

Atualmente os Sistemas de Apoio à Decisão Clínica (SADC) são um componente essencial para qualquer sistema informático clínico. A sua correta integração nos sistemas existentes permite uma maior eficiência na aquisição e disponibilização de dados importantes e relativos a cada paciente. Deste modo, é imperativo que um SADC garanta a interoperabilidade semântica, isto é, a capacidade de comunicar com outras plataformas sem que hajam perdas de significado no seu conteúdo. É necessário também garantir que estes sistemas sejam de fácil integração, de modo a possibilitar a agregação da maior quantidade de dados clínicos possível dos diferentes sistemas existentes nas instituições.

Com isto, surge então a necessidade de optar por padrões que garantam a interoperabilidade semântica deste sistemas e que facultem meios para a sua rápida integração. Neste sentido, emerge o openEHR, um padrão *e-health* aberto que oferece os meios necessários para garantir as propriedades supracitadas anteriormente. Adicionalmente, mas não menos importante, este apresenta um módulo de apoio à decisão que permite codificar a lógica de decisão presente nos planos e *guidelines* clínicos.

São várias as definições existentes para o conceito de *guideline*, no entanto, pode-se assumir, para efeitos de simplificação, o *guideline* como sendo um conjunto de declarações que visam guiar o profissional de saúde na tomada de decisão no seu dia-a-dia. Este tipo de sistema é relevante pois visa minimizar possíveis erros de teor humano que podem facilmente ser cometidos por parte dos profissionais que estão sujeitos diariamente a vários factores como turnos extensos, cansaço, entre outros e que em última instância podem comprometer a tomada de uma decisão acertada.

Assim, a presente dissertação tem como finalidade desenvolver um Sistema de Apoio à Decisão Clínica, baseado em regras e no modelo openEHR, que permite a criação, gestão e execução de *guidelines* clínicos. Para além disso, foi-se mais além e desenvolveu-se uma interface gráfica que faculta ao utilizador um meio simples, intuitivo e de agradável visualização.

**Palavras-chave** Sistemas de Apoio à Decisão Clínica, openEHR, interoperabilidade semântica, motor de inferência, módulo de apoio à decisão, *guidelines*, interoperabilidade semântica, registos eletrónicos de saúde

# Abstract

## **Rule-based Clinical Decision Support System using openEHR model**

Nowadays, Clinical Decision Support System (CDSS) are an essential component for any clinical computer system. Correctly integrated into existing systems allows for a higher efficiency in the availability of important data related to each patient. Hence, it is imperative that a CDSS ensures a semantic interoperability, in other terms, the capacity to communicate with other platforms without losing the content's significance. It is also necessary to ensure that these systems are able to be easily integrated, in order to enable the aggregation of as much clinical data as possible from the different existing institutional systems. With this comes the need to rely on pattern based languages that guarantee a semantic interoperability of these systems and that provide crucial tools for fast integration.

And so, openEHR emerges, an open e-health pattern based language that offers the necessary means to cover the previously mentioned properties. Additionally, openEHR provides a decision support module that allows for the coding of the decision logic present in the clinical plans and guidelines.

The existing definitions for the concept of a guideline tends to vary significantly, we can assume, for simplification purposes, that a guideline is a group of declarations with the objective of guiding the health professional in the decision making process of their daily professional lives. This type of system proves itself relevant by trying to minimize the human error component of the decision making process, seen as every day health professionals are subjected to factors that can diminish their focus, such as long shifts, fatigue and many others that can sometimes compromise their decisions and decision making speed.

With all this in mind, the ultimate goal of this dissertation is to develop a clinical decision support system based on specific rules and on the openEHR module, which will enable the creation, management and execution of clinical guidelines. Furthermore, the choice was made to go beyond and develop a graphical interface that provides the user with simple, intuitive and pleasant means of visualization.

**Keywords:** Clinical Decision Support System, openEHR , semantic interoperability, inference engine, decision support module, guidelines, semantic interoperability, electronic health records .



# Índice

<b>Índice de Figuras</b>	<b>xi</b>
<b>Índice de Tabelas</b>	<b>xiii</b>
<b>Siglas</b>	<b>xiv</b>
<b>1 Introdução</b>	<b>1</b>
1.1 Enquadramento . . . . .	1
1.2 Finalidade e Objetivos . . . . .	2
1.3 Motivação . . . . .	3
1.4 Estrutura da Dissertação . . . . .	4
<b>2 Estado de Arte</b>	<b>5</b>
2.1 Evolução dos Sistemas de Apoio à Decisão Clínica . . . . .	5
2.2 Sistemas de Apoio à Decisão Clínica . . . . .	8
2.2.1 Categorização dos SADCs . . . . .	8
2.2.2 Funcionalidades dos SADCs . . . . .	9
2.2.3 Vantagens na adoção de SADCs . . . . .	10
2.2.4 Dificuldades e riscos no desenvolvimento e adoção de SADCs . . . . .	11
2.3 Modelo OpenEHR . . . . .	12
2.3.1 Definição . . . . .	12
2.3.2 Como surgiu openEHR . . . . .	12
2.3.3 Visão Geral da Arquitetura OpenEHR . . . . .	13
2.4 Linguagens de Decisão Padronizadas existentes no mercado . . . . .	14
2.4.1 Arden Syntax . . . . .	15
2.4.2 Guideline Interchange Format GLIF . . . . .	17
2.4.3 PROforma . . . . .	19

2.4.4	Decision Logic Module (DLM) - Suporte à decisão . . . . .	20
2.4.5	Visão Crítica das Linguagens Abordadas . . . . .	29
<b>3</b>	<b>Abordagens Metodológicas e Ferramentas Utilizadas</b>	<b>31</b>
3.1	Design Science Research . . . . .	31
3.2	Tecnologias . . . . .	33
3.2.1	MERN Stack . . . . .	33
3.2.2	Tecnologias Adotadas . . . . .	35
<b>4</b>	<b>Decision Maker</b>	<b>37</b>
4.1	Requisitos . . . . .	37
4.1.1	Requisitos Funcionais: . . . . .	37
4.1.2	Requisitos Não Funcionais: . . . . .	38
4.2	Arquitetura . . . . .	41
4.3	Representação Computacional do Guideline: . . . . .	44
4.4	Motor de Inferência . . . . .	47
4.5	Demonstração e Explicação de Conceitos . . . . .	49
4.5.1	Página Inicial . . . . .	50
4.5.2	Barra de Navegação . . . . .	50
4.5.3	Criação Guideline . . . . .	51
4.5.4	Guidelines . . . . .	62
4.5.5	Subject Proxy . . . . .	66
<b>5</b>	<b>Avaliação</b>	<b>72</b>
5.1	Análise SWOT . . . . .	72
<b>6</b>	<b>Conclusão</b>	<b>75</b>
	<b>Bibliografia</b>	<b>77</b>

## Índice de Figuras

1	Arquitetura OpenEHR (retirado de [24]) . . . . .	14
2	Organização do Medical Logic Module (retirado de [25]) . . . . .	16
3	Arquitetura Decision Logic Module do openEHR (adpatado de [68]) . . . . .	20
4	Estrutura openEHR do DLM (retirado de [59]) . . . . .	23
5	Secção de identificação (retirado de [59]) . . . . .	23
6	Secção de Linguagem e Descrição (retirado de [59]) . . . . .	24
7	Secção dos Identificadores (retirado de [59]) . . . . .	25
8	Secção do <i>Use_model</i> (retirado de [59]) . . . . .	25
9	Secção das Pré-Condições (retirado de [59]) . . . . .	26
10	Secção das Referências (retirado de [ <b>OpenEHRSpecificationProgram2</b> ]) . . . . .	26
11	Tabela de Riscos, presente na Secção das Referências (retirado de [59]) . . . . .	26
12	Atribuição da variável do Subject Proxy, presenta na Secção de Input (retirado de [59]) . . . . .	27
13	Secção de Input (retirado de [59]) . . . . .	27
14	<i>Tracked Variables</i> com o modificador <i>range</i> (retirado de [59]) . . . . .	28
15	Exemplo de uma Secção de Regras (retirado de [59]) . . . . .	28
16	Exemplo de uma Secção das Terminologias (retirado de [59]) . . . . .	29
17	Modelo da metodologia DSR (retirado de [33]) . . . . .	32
18	Mern Stack (retirado de [39]) . . . . .	33
19	Arquitetura Mern Stack (retirado de [39]) . . . . .	33
20	Diagrama da Arquitetura Funcional do Decision Maker . . . . .	41
21	Diagrama de Arquitetura . . . . .	42
22	Esquema que pretende mostrar um exemplo de um documento armazenado na base de dados . . . . .	44
23	Adpatação do diagrama de dataflow para a função <i>evaluatePreConditions</i> . . . . .	47
24	Adpatação do diagrama de dataflow para a função <i>run</i> que representa o motor de inferência . . . . .	49

25	Página Inicial do Guideline Creator . . . . .	50
26	Barra de navegação . . . . .	51
27	Criação do Guideline, secção de Descrição . . . . .	52
28	Criação do Guideline, secção de Configuração do Sujeito do Subject Proxy . . . . .	53
29	Demonstração da mensagem de erro mostrada caso o identificador inserido seja inválido.	53
30	Criação do Guideline, secção das Condições . . . . .	54
31	Demonstração da criação de uma condição do tipo Manual . . . . .	55
32	Demonstração da lista de tipos existente . . . . .	55
33	Demonstração da lista de operadores definida para o tipo Texto . . . . .	55
34	Demonstração da lista de operadores definida para o tipo Número . . . . .	56
35	Demonstração da lista de operadores definida para o tipo Lista . . . . .	56
36	Demonstração da criação de uma condição do tipo Variável do Subject Proxy . . . . .	57
37	Demonstração da lista de variáveis do Sujeito do Subject Proxy apresentadas ao utilizador	57
38	Criação do Guideline, secção das Pré-condições . . . . .	58
39	Demonstração da criação de um pré-condição do tipo Manual . . . . .	58
40	Demonstração da criação de um pré-condição do tipo Criar Condição . . . . .	59
41	Criação do Guideline, secção das Regras . . . . .	60
42	Demonstração da criação de eventos associada a uma regra específica . . . . .	61
43	Criação do Guideline, secção do Resumo . . . . .	62
44	Guidelines . . . . .	63
45	Guidelines, secção da Informação detalhada . . . . .	64
46	Guidelines, secção da Execução . . . . .	64
47	Guidelines, secção da Execução . . . . .	65
48	Guidelines, demonstração de um exemplo de output gerado . . . . .	65
49	Guidelines, demonstração de um exemplo de output gerado, caso as pré-condições não sejam cumpridas . . . . .	66
50	Guidelines, demonstração de um exemplo de output gerado, caso as regras não sejam cumpridas . . . . .	66
51	Subject Proxy, Adicionar novo Sujeito . . . . .	67
52	Subject Proxy, Remover Sujeito . . . . .	68
53	Subject Proxy, Adicionar nova variável ao Sujeito . . . . .	68
54	Subject Proxy, Adicionar novo dataframe . . . . .	69
55	Subject Proxy, Adicionar novo dataframe do tipo API . . . . .	70
56	Subject Proxy, Adicionar novo dataframe do tipo Query . . . . .	70
57	Subject Proxy, Adicionar novo databinding . . . . .	71

## Índice de Tabelas

1	Tabela Resumo de Tecnologias utilizadas . . . . .	36
2	Análise SWOT: Forças e Fraquezas . . . . .	73
3	Análise SWOT: Oportunidades e Ameaças . . . . .	74

# Siglas

<b>API</b>	Application Programming Interface <a href="#">7</a> , <a href="#">34</a> , <a href="#">40</a> , <a href="#">67</a> , <a href="#">69</a> , <a href="#">70</a>
<b>BD</b>	Base de Dados <a href="#">35</a>
<b>BNF</b>	Backus-Naur <a href="#">19</a>
<b>DLM</b>	Decision Logic Module <a href="#">x</a> , <a href="#">20</a> , <a href="#">21</a> , <a href="#">24</a> , <a href="#">25</a> , <a href="#">26</a> , <a href="#">28</a> , <a href="#">29</a> , <a href="#">37</a>
<b>DSR</b>	Design Science Research <a href="#">xi</a> , <a href="#">31</a> , <a href="#">32</a>
<b>EHR</b>	Eletronic Health Record <a href="#">3</a> , <a href="#">8</a> , <a href="#">12</a> , <a href="#">13</a> , <a href="#">14</a> , <a href="#">18</a>
<b>ERP</b>	Enterprise Resource Planning <a href="#">7</a>
<b>GEHR</b>	Good European Health Record <a href="#">12</a> , <a href="#">13</a>
<b>GLIF</b>	Guideline Interchange Format <a href="#">6</a> , <a href="#">17</a> , <a href="#">18</a> , <a href="#">29</a>
<b>HIMSS</b>	Health Information And Management Systems Society <a href="#">3</a>
<b>HL7</b>	Health Level 7 <a href="#">18</a>
<b>HTML</b>	HyperText Markup Language <a href="#">34</a>
<b>IA</b>	Inteligência Artificial <a href="#">1</a> , <a href="#">5</a>
<b>JSON</b>	Javascript Object Notation <a href="#">34</a> , <a href="#">35</a>
<b>JSX</b>	JavaScript XML <a href="#">34</a>
<b>MERN</b>	MongoDB ExpressJS ReactJS NodeJS <a href="#">33</a>
<b>ML</b>	Machine-Learning <a href="#">1</a>

<b>MLM</b>	Medical Logic Module <a href="#">15</a> , <a href="#">16</a>
<b>OLAP</b>	Online Analytical Processing <a href="#">7</a>
<b>RM</b>	Reference Model <a href="#">14</a>
<b>RMRS</b>	Regenstrief Medical Record System <a href="#">6</a>
<b>SADC</b>	Sistema de Apoio à Decisão Clínica <a href="#">1</a> , <a href="#">3</a> , <a href="#">4</a> , <a href="#">5</a> , <a href="#">6</a> , <a href="#">7</a> , <a href="#">8</a> , <a href="#">9</a> , <a href="#">10</a> , <a href="#">11</a> , <a href="#">12</a> , <a href="#">74</a>
<b>SAGE</b>	Shareable Active Guideline Environment <a href="#">7</a>
<b>SGDB</b>	Sistemas de Gestão de Base de Dados <a href="#">6</a>
<b>SQL</b>	Structured Query Language <a href="#">35</a>
<b>UCL</b>	Universidade Colleague London <a href="#">13</a>
<b>URL</b>	Uniform Resource Locator <a href="#">34</a>
<b>XML</b>	Extensible Markup Language <a href="#">34</a>





# Introdução

No presente capítulo introdutório, será abordado o enquadramento do tema de dissertação, assim como a sua finalidade e objetivos, motivação e a descrição da estrutura do presente documento.

## 1.1 Enquadramento

Os Sistemas de Apoio à Decisão Clínica (SADC) podem ser definidos como aplicações informáticas que relacionam os dados clínicos de cada paciente de modo a estabelecer uma base de conhecimento e, assim, auxiliar os profissionais de saúde na tomada de decisão [47, 71]. Estes podem apresentar diversas finalidades como: gestão de informação, gestão da complexidade, gestão de detalhes clínicos através de alertas, controle de custos, suporte à decisão através de recomendações específicas ao paciente, quer na fase de diagnóstico, quer na fase de terapia, entre muitas outras [47]. Além disto, os SADCs podem-se manifestar como alertas, lembretes, *guidelines* computadorizados, conjuntos de pedidos, relatórios de dados dos pacientes, *templates* de documentação e ainda como ferramentas de *workflow* clínico [62].

Contudo, de modo a clarificar o conceito de SADC é necessário salientar que estes são, frequentemente, classificados como sistemas baseados em conhecimento ou como sistemas não-baseados em conhecimento [62]. Os sistemas baseados em conhecimento tem como base de funcionamento a avaliação de um conjunto de regras (afirmações if-then-else) que incidem sobre os dados relativos a um determinado sujeito, produzindo em última instância uma acção ou um *output*. Por sua vez, os sistemas não-baseados em conhecimento utilizam inteligência artificial (IA), machine-learning (ML) ou o reconhecimento de padrões estatísticos para a tomada de decisão. No entanto, estes sistemas apresentam diversos problemas como a falta de transparência no que diz respeito à lógica assente na tomada de decisão e problemas com a disponibilidade dos dados[62].

Inseridos nos SADC baseados em conhecimento surgem os *guidelines* clínicos. Segundo a literatura, um *guideline* pode ser entendido como um conjunto de declarações, desenvolvidas com base em evidências literárias, que visam auxiliar os profissionais de saúde e os pacientes na tomada de decisões sobre tratamentos adequados para determinadas condições clínicas específicas [15]. A sua utilização no contexto clínico tem vindo a apresentar inúmeras vantagens sendo a mais relevante o melhoramento dos

cuidados prestados ao paciente [10]. No entanto a sua adoção por parte das organizações de saúde tem mostrado que os clínicos na sua maioria estão pouco à vontade a escrever *guidelines* e falham na sua correcta utilização aquando a prestação dos cuidados de saúde. Neste sentido, a implementação de *guidelines* computadorizados promete melhorar a sua aceitação e aplicabilidade no contexto da rotina clínica do profissional. Estes sistemas de apoio à decisão têm mostrado melhorar significativamente a qualidade dos cuidados prestados, especialmente quando estes utilizam sistemas de informações clínicas como os registos eletrónicos dos pacientes [10].

Portanto, conforme o título do relatório de dissertação indica o sistema de apoio à decisão clínica que foi desenvolvido é baseado em conhecimento e é orientado à criação de *guidelines* clínicos, sendo baseado também no modelo openEHR. Mas, afinal, o que é o modelo openEHR?

O openEHR é um padrão *e-health* que disponibiliza um conjunto de especificações abertas e modelos clínicos que possibilitam a criação de registos eletrónicos clínicos em módulos e de acordo com a necessidade, facultando uma solução interoperável para sistemas de cuidados de saúde [2, 19]. O seu principal objetivo assenta na possibilitação da construção de sistemas de registos eletrónicos de saúde que sejam capazes de comunicar entre si, sem que se perca o significado do seu conteúdo. Assim, o objetivo máximo é atingir a interoperabilidade semântica. Adicionalmente, o openEHR apresenta um conceito inovador que o distingue dos demais padrões *e-health* que é a modelação em dois níveis de conhecimento clínico

Conclusivamente, pretende-se tirar partido desta tecnologia com o intuito de criar um sistema de apoio à decisão clínica interoperável e de fácil integração. Para além disto, pretende-se ir mais longe e criar um sistema que permite o fácil desenvolvimento de *guidelines* que, posteriormente, podem ser utilizados para auxiliar o profissional de saúde no seu quotidiano de trabalho.

## 1.2 Finalidade e Objetivos

Como já referido neste capítulo, pretende-se implementar um sistema de apoio à decisão clínica baseado em regras e no modelo openEHR que visa a criação e gestão de *guidelines* clínicos. Deste modo, irá se explorar os conceitos principais que permitam alcançar uma solução tangível e correta, visando cumprir os seguintes objetivos principais:

1. criação de um módulo de decisão lógica padronizado;
2. representação do conhecimento através de regras e *guidelines* clínicos;
3. garantir a interoperabilidade semântica entre os sistemas;
4. construção de uma interface intuitiva, dinâmica e apelativa.

Assim, para se conseguir alcançar os objetivos acima delineados será necessário basear no módulo openEHR de modo a se alcançar um solução interoperável e de utilização intuitiva.

## 1.3 Motivação

A quantidade e qualidade de dados clínicos tem vindo a expandir-se rapidamente [22] o que nem sempre significa que a sua disponibilidade está assegurada quando é mais precisa. Assim, é empírico disponibilizar estes dados aos profissionais de saúde, de modo a que possam tomar decisões corretas e específicas a cada paciente. A tomada de decisão torna-se, portanto, imperativa para os cuidados de saúde, na medida em que é necessário auxiliar os profissionais de saúde com o máximo de informação possível acerca de cada paciente.

Deste modo, em colaboração com o Centro Universitário do Hospital do Porto e com centro ALGORITMI da Universidade do Minho surge a motivação da criação de um sistema de apoio à decisão clínica que permita a criação de *guidelines* e a sua posterior utilização. Este projecto visou integrar sistemas de *eletronic health records* - EHR - de modo a que a solução final fosse o mais madura e prestigiada possível. Neste sentido é importante referir que a fundação HIMSS [27] propõe um modelo de avaliação de sistemas de *eltronic health records* (EHR), onde já considera como requisito a integração de sistemas EHR nos SADC para que atinja o nível de prestígio e de maturidade necessários para a sua aplicabilidade no mundo real. Esta metodologia procura, assim, avaliar os sistemas, classificando-os de acordo com o seu nível de maturidade, estando relacionado com adoção e utilização de sistemas de EHR.

A necessidade da criação de *guidelines* clínicos computadorizados prende-se também no aumento da sua utilização em diversas áreas como desenvolvimento de políticas, gestão de utilização, educação, ensaios clínicos e auxílio no fluxo de trabalhos [10]. Para além disso, como já referido anteriormente, estes sistemas permitem ajudar o profissional de saúde na sua rotina de trabalho, possibilitando, assim, o melhoramento dos cuidados prestados ao paciente.

No entanto, a implementação de sistemas como estes apresentam diversas dificuldades que em última instância podem colocar em risco a sua adoção por parte das organizações e/ou profissionais de saúde. Por exemplo, a representação e formalização do conceito de *guideline* a nível informático apresenta complicações visto não existir um padrão consensual definido [10]. Para além disso, por vezes as soluções implementadas não utilizam formas automatizadas de aquisição de dados clínicos relativos ao sujeito alvo de cuidados ou provam ser de extrema dificuldade de utilização ou pouco intuitivos.

Tendo em consideração o que foi supracitado anteriormente, torna-se imperativo implementar um SADC que possibilite a fácil integração de EHR e facilite a criação de *guidelines* de modo a otimizar a tomada de decisões por parte dos profissionais de saúde no seu quotidiano. Assim sendo, justifica-se a criação de um sistema de apoio à decisão baseado em regras e no modelo openEHR que permite a criação e execução de *guidelines* e que em última instância ofereça uma interface de fácil entendimento e utilização.

## 1.4 Estrutura da Dissertação

Este documento é composto por cinco capítulos.

O primeiro capítulo corresponde à Introdução e neste procurou-se enquadrar o problema proposto, definir os objetivos, definir as finalidades, expor os motivos que levaram à escolha do tema e , por último, detalhou-se a estrutura que o presente documento contém.

O segundo capítulo é o Estado de Arte e corresponde à revisão de literatura que engloba os principais temas a abordar. Assim, neste capítulo começou-se por explicar a evolução dos sistemas de apoio à decisão clínica ao longo dos anos, seguindo-se de uma definição de [SADCs](#) e dos tipos existentes. Posteriormente, abordou-se o principal padrão a ser estudado ,o openEHR, prosseguindo-se para uma análise de mercado das linguagens de padrão de decisão existentes e que se consideram relevantes para o tema em questão. Para finalizar, analisou-se alguns artigos científicos e prosseguiu-se para a elaboração dos casos de estudo.

O terceiro capítulo refere-se às Abordagens Metodológicas e Ferramentas Utilizadas, pretendo-se delinear as metodologias de pesquisas utilizadas, bem como as ferramentas e tecnologias utilizadas.

O quarto capítulo diz respeito à fase de Desenvolvimento da aplicação *web* Decision Maker e neste pretendeu-se averiguar os requisitos necessários do sistema, delinear os conceitos relevantes, explicar o modo de funcionamento de certos componentes que se acharam importantes e ainda disponibilizar ao leitor uma explicação detelhada do modo de utilização da aplicação desenvolvida.

Por último, o capítulo cinco apresenta as considerações finais, bem como possíveis projetos futuros que podem dar continuidade ao projeto desenvolvido.

## Estado de Arte

### 2.1 Evolução dos Sistemas de Apoio à Decisão Clínica

Segundo a literatura [52], foi no anos 60 que investigadores começaram a estudar a utilização de modelos computacionais para auxiliar a tomada de decisão e o planeamento clínico. De facto, talvez a primeira referência data 1959 com a publicação do artigo científico de Ledley e Lusted [34], cujo principal propósito é entender o complexo processo de raciocínio assente no diagnóstico médico. Embora o foco central do trabalho realizado por Ledley e Lusted tenha sido a investigação da lógica do raciocínio presente no diagnóstico médico, estes também propuseram um modelo teórico que produzia um diagnóstico diferencial com o auxílio de um computador analógico. O modelo era trivial e baseava-se na utilização de cartas que continham o diagnóstico e uma série de furos que representavam os sintomas. Posteriormente, o computador "sorteavas" e produzia assim um diagnóstico diferencial.

Poucos anos depois, começaram a surgir mais modelos, como, por exemplo, o modelo matemático para o diagnóstico do defeito congénito do coração [66], o sistema baseado em cartas que continham perguntas ou sintomas e cujo a finalidade se focava no *Automated multiphasic screening and diagnosis* [7] e ainda modelo probabilístico que assentava no sistema computacional e que auxiliava no diagnóstico de dores abdominais agudas [13, 12]. Contudo, houve um sistema que na altura era considerado único [69], pois para além de realizar diagnósticos para os distúrbios ácido-base, também sugeria tratamentos. Adicionalmente, o sistema era capaz de produzir alertas, caso alguma informação faltasse [5].

Por volta da década de 70, a inteligência artificial começou a influenciar a informática na medicina [69], tendo surgido um sistema de prescrição de antibióticos, utilizando técnicas de IA [56]. Este sistema continha regras e dados que foram inseridos e demonstrou desenvolver uma terapia aceitável cerca de 75% da vezes, sendo que esta tinha tendência a melhorar com a inserção de novas regras.

Obviamente, que a lista de modelos acima inumerados representa apenas uma fatia daqueles que nas décadas de 60 e 70 foram publicados. No entanto, a maior parte dos modelos apresentavam traços característicos da altura. Este tipo de SADCs na sua maioria eram de uso muito específico e orientado a uma área particular da medicina. Para além disso, este tipo de sistema corria separadamente de qualquer outro tipo sistema informático, o que apresentava uma limitação à sua utilização pois exigia

que o utilizador inserisse manualmente os dados no sistema, que por vezes eram numa quantidade exorbitante. Este tipo de sistemas continham ainda mais falhas como a dificuldade que apresentavam em acompanhar as mudanças necessárias, a falta de robustez por parte das bases de dados para a realização de pesquisas típicas e, ainda, os custos elevados de implementação e manutenção que necessitavam [4]. Face ao problema da falta de integração dos SADCs nos sistemas informáticos clínicos, novas formas de implementação surgiram para culmar estas limitações.

O sistema HELP [31] foi o primeiro exemplo de tal integração [69]. Este foi desenvolvido para gerir dados clínicos e administrativos e, portanto, tanto auxiliava o profissional de saúde na tomada de decisão, como o profissional administrativo nas suas tarefas. Outro exemplo de tal integração foi o *Regenstrief Medical Record System - RMRS* - que é um sistema médico integrado de registo computacional para cuidados ambulatoriais [38]. Para além destes sistemas, outros foram desenvolvidos, maioritariamente em centros médicos académicos [69].

Apesar das vantagens que este tipo de sistema forneceu como a diminuição da inserção de dados por parte do utilizador ou a capacidade de gerar alertas, por exemplo, para interações perigosas de medicamentos, as desvantagens são de salientar. Provavelmente a mais relevante era a dificuldade em serem reutilizados, isto é, como são sistemas integrados nos sistemas clínicos estes não podem ser diretamente partilhados. Aditivamente, esta integração acrescia novos problemas, no que diz respeito à gestão do conhecimento, pois era difícil separar o código do conhecimento, o que tornava penoso o processo de atualização do conhecimento.

Assim, com o surgimento dos primeiros Sistemas de Gestão de Base de Dados - SGDB - na década de 80, foi possível melhorar o acesso aos dados disponíveis e, assim, reduzir custos [4]. No entanto, a nível de administração clínica, estes sistemas ainda eram bastante primordiais, pois apresentavam dificuldades no cruzamento de dados entre sistemas diferentes para a realização de relatórios. Este processo era dispendioso e exigia tempo pois era realizado na maior parte dos casos, manualmente.

No final dos anos 80 e inícios dos anos 90 começam a surgir linguagens que visam padronizar o conhecimento clínico e torná-lo partilhável. Exemplos dessas implementações são o Arden Syntax [65], que combinava as sintaxes utilizadas no sistema HELP e RMRS (sistemas mais utilizados na altura), e o Guideline Interchange Format - GLIF - que é modelo para representar *guidelines* clínicos [41].

O uso de padrões para a representação, codificação, armazenamento e partilha de conhecimento veio melhorar os sistemas integrados, na medida em que separou a lógica do conhecimento. No entanto, o surgimento de muitos padrões acarretou algumas limitações aos SADCs, precisamente por existir uma grande variedade destes. Outra limitação está presente na utilização de terminologias que tinham de ser comuns aos sistemas (clínico e de apoio à decisão), pois caso contrário, exigia um mapeamento demorado para resolver as diferenças.

Contudo, a década de 90 foi um marco importante pois as tecnologias informáticas sofreram um grande melhoramento. Segundo a literatura [67, 64], foi no início dos anos 90 que apareceram ferramentas CASE e as linguagens de Quarta Geração. O aparecimento destas ferramentas possibilitou um

melhoramento significativo no processo de desenvolvimento destes sistemas e também nas funcionalidades que estes apresentavam.

Contudo, segundo a literatura [67, 64], o surgimento das ferramentas Case e das linguagens de 4<sup>o</sup> geração impulsionaram não só o desenvolvimento dos SADCs, na medida em que tornou a sua implementação mais rápida e fácil, mas também melhorou o processo de emissão de relatórios e de realizações de consultas. Apesar destas ferramentas serem fortemente baseadas na manipulação e navegação dos dados, careciam de outros aspetos importante como a sua eficácia na análise de dados, sendo que se tornava limitadores [64]. Porém as folhas de cálculo tentaram culmar esta limitação ao produzir gráficos a partir de dados com extrema facilidade. No entanto, estas ferramentas não foram suficientes para satisfazer as necessidades cada vez mais crescentes do mercado comercial.

À medida que o desenvolvimento tecnológico avançava, o volume de dados armazenados também foi aumentado pelo que levou à necessidade de haver mais análise de dados, respostas mais rápidas e confiáveis, de modo a adaptar às necessidades da procura [64].

A queda dos custos de armazenamento de dados no mercado contribuiu bastante para a evolução das bases de dados. Com a globalização, a concorrência aumentou imenso a nível mundial, o que fez aumentar as necessidades administrativas, na medida em que passa a ser essencial novos tipos de consultas e de análise de dados [64]. Assim, foram desenvolvidos sistemas que permitiram culmar as necessidades requeridas pela procura. Por exemplo, o surgimento da Enterprise Resource Planning - ERP- que permite a gestão integrada da empresa, o datawarehouse, OLAP e data mining que corresponderam à nova geração de sistemas de apoio à decisão. Estas ferramentas permitiram a gestão do ambiente operacional e ambiente gerencial da empresa. Para além disto, o data warehouse e OLAP possibilitaram a elaboração de relatorios mais rápida, barata, confiável e acessível a qualquer utilizador, mesmo que não tivesse conhecimentos profundos sobre tecnologias informáticas [64]. Assim, a utilização destas ferramentas possibilitou a diminuição no tempo de gestão de dados.

A par com esta evolução tecnologica também começaram a surgir uma nova geração de SADCs que consumiam APIs. O Shareable Active Guideline Environment project - SAGE - foi alegadamente o primeiro projeto a utilizar uma API no sistema clínico [53]. O SEBASTIAN, por outro lado, utilizava uma interface *standard*. No entanto, estes sistemas apresentavam algumas desvantagens principalmente pela limitação presente na junção dos dois sistemas (clínico e de apoio à decisão) pois era necessário padronizar uma das interfaces. Adicionalmente, os sistemas não fazem cruzamento de informação entre sistemas o que não representa a forma como o conhecimento é adquirido no mundo real.

Como se pode verificar, os sistemas de apoio à decisão clinica foram evoluindo e tomando novas formas e arquiteturas. Com o decorrer dos anos, as várias arquiteturas aqui apresentadas foram sendo desenvolvidas, havendo possivelmente uma maior inclinação para os sistemas integrados, padronizados e orientados aos serviços.

## 2.2 Sistemas de Apoio à Decisão Clínica

Segundo Shortliffe e Edward H [57] "os sistemas de apoio à decisão clínica (SADC) são descritos como (...) qualquer programa informático desenhado para ajudar profissionais de saúde na tomada de decisão". Este tipo de programa é capaz de relacionar dados clínicos relativos a cada paciente, de forma a estabelecer bases de conhecimento que têm como finalidade assistir a tomada de decisão [47]. Contudo, é importante salientar que este tipo de sistema não é responsável, propriamente, por realizar decisões médicas. É, no entanto, responsável por fornecer dados devidamente analisados e fundamentados que são pertinentes na tomada de decisão.

### 2.2.1 Categorização dos SADCs

Os SADCs podem ser classificados de várias formas, dependendo da sua finalidade, da sua arquitetura, se são ou não integrados no sistemas informáticos clínicos, entre outras [1]. Uma forma de categorizar os SADCs é se estes são baseados em conhecimento ou não [35].

Os sistemas de apoio à decisão clínica não baseados em conhecimento utilizam mecanismos de inteligência artificial - machine learning, redes neuronais artificiais e algoritmos genéticos- que permitem que o programa adquira novo conhecimento com experiências passadas e/ou seja capaz de reconhecer padrões nos dados [37, 1]. Apesar de estudos mostrarem que este tipo de abordagem possa ser mais preciso no diagnóstico de doenças específicas [3, 35], a sua utilização em massa ficou limitada devido à falta de transparência na lógica de decisão que a inteligência artificial utiliza e também devido a problemas relacionados com a disponibilidade dos dados [11].

Relativamente aos SADCs baseados em conhecimento estes criam regras conhecidas como declarações if-then que são avaliadas pelo sistema através dos dados presentes, produzindo em última instância um *output* ou uma ação [55]. Assim, cada sistema baseado em conhecimento apresenta 3 componentes [55]:

1. **Base de Conhecimento** - consiste na coleção de dados, devidamente tratada, que habitualmente pode ser apresentada sob a forma de regras if-then;
2. **Motor de inferência** - também conhecido como mecanismo de raciocínio, este contém fórmulas que combinam as regras e/ou *guideline* (presentes na base de conhecimento) com os dados atuais do paciente;
3. **Mecanismo de Comunicação** - consiste no mecanismo de integração dos dados atuais relativos ao paciente no sistema. Para além disso, é responsável por apresentar o *output* resultante ao utilizador que, ultimamente, irá tomar a decisão. Em sistemas como os não integrados, os dados do paciente são inseridos manualmente pelo utilizador. Em sistemas integrados os dados são obtidos através de EHR. Assim, os dados encontram-se no formato eletrónico, permitindo assim obter informações de outros sistemas como os de laboratórios, farmácias, centros de saúde, entre



outros. No final, o *output* gerado para o utilizador pode se apresentar na forma de alerta ou recomendação, entre outras.

### 2.2.2 Funcionalidades dos SADCs

Primordialmente, este tipo de sistema tinha como única finalidade o auxílio na tomada de decisão, porém com o seu desenvolvimento ao longo dos anos, novas funcionalidades começaram a surgir, tornando estes sistemas mais completos e robustos, culminando, assim, algumas das necessidades que o mercado começara a exigir.

Na atualidade, as ferramentas de apoio à decisão podem incluir inúmeras áreas de aplicação, funcionalidades e finalidades, como por exemplo [50, 62]:

- Diagnóstico;
- Administrativa;
- Acompanhamento de doenças;
- Prescrições;
- Controlo de medicação;
- Fluxogramas de apoio ao diagnóstico e à terapêutica;
- Controlo de custos e iatrogenia - monitoração de medicação, prevenção de testes duplicados ou desnecessários;
- Reconhecimento e interpretação de padrões de imagem;
- Adaptação de orientações específicas à condição do paciente;

Os SADCs podem ainda se manifestar de diferentes formas, dependendo da sua finalidade e objetivos. Assim, temos:

- Alertas e lembretes computadorizados;
- *Guidelines* computadorizados;
- Conjuntos de pedidos;
- Relatórios de pacientes;
- *Templates* de documentação;
- Ferramentas de *workflow* clínico.

### 2.2.3 Vantagens na adoção de SADCs

Como já referido ao longo do presente documento, os SADCs visam melhorar os cuidados prestados ao paciente. No entanto, dependendo das diferentes finalidades que estes podem apresentar, outras vantagens emergem com a adoção deste tipo de sistemas.

Segundo estudos [62], alguns tipos de SADCs que apresentam como finalidade a gestão de prescrições/dosagens de medicamentos, têm se mostrado vantajosos e até mesmo benéficos na redução de erros relativos. Geralmente, este tipo de sistema apresenta-se sob a forma de alertas e lembretes, que permitem informar o profissional de saúde de, por exemplo, a presença de interações perigosas na prescrições ou de erros de dosagens. Assim, com o auxílio deste tipo de sistemas, é se capaz de proporcionar um melhoramento na segurança do paciente, promovendo a criação de prescrições seguras, evitanto-se, deste modo, possíveis efeitos secundários.

Outros estudos, mostram ainda, que os SADCs podem aumentar a adesão à utilização de *guidelines* clínicos [32], que na sua forma tradicional (formulários em papel, folhas de excel, entre outros) têm-se mostrado difíceis de utilizar, resultando numa fraca adesão por parte dos profissionais de saúde. A criação de *guidelines* computadorizados e a sua integração nos SADCs visa culmar este problema, na medida em que o profissional não necessita mais de ler as diretrizes, assimilá-las e implementá-las na sua prática quotidiana [62].

Os SADCs mostram-se ainda serem vantajosos no controlo de custos dos sistemas de saúde, na medida em que, alguns, possibilitam a diminuição do tempo de internamento, ou alternativas de medicamentos mais baratos ou ainda a com a redução da realização de testes duplicados [62].

Os SADCs podem ainda fornecer auxílio na codificação de diagnósticos, ordenação de procedimentos e testes e na triagem de pacientes [62]. Esta funcionalidade permite ajudar os profissionais de saúde na medida em que muitas vezes este utilizam documentação com códigos de diagnóstico ou protocolos clínicos padronizados. Por conseguinte, com a utilização de SADCs que melhoram diretamente a qualidade, precisão e atualidade desta documentação, resultando num melhor cuidado prestado.

Muitos SADCs são desenvolvidos com o objetivo final de auxiliar o profissional de saúde na tomada de decisão, através do forcimento de uma lista de possíveis diagnósticos. Estes podem abranger diferentes áreas de aplicação quando implementados correctamente, como imagiologia, laboratórios e patologias. Deste modo, podem fornecer auxílio a radiologistas na seleção do melhor pedido de imagem a realizar, por exemplo, ou até profissionais que realizam análises laboratoriais ajudando-os na análise e interpretação dos resultados.

Outros SADCs possibilitam a partilha de decisão entre paciente e o prestador de cuidados, isto é, a possibilidade do paciente puder participar ativamente nos cuidados prestados. Estes sistemas, normalmente apresentam-se sob a forma de *software*, aplicação *web* e/ ou *mobile*. Este tipo de sistemas apresenta inúmeras vantagens como a possibilidade de o paciente falar diretamente com o prestador de cuidados e inserir informação, que em última instância irá resultar num tratamento mais preciso,

auxiliar na deteção de doenças e oferecer possíveis tratamentos, monitorizar certos parâmetros (ex: monitorização da glicose através de aparelhos como *smartwatch*, em pacientes com diabetes) que serão posteriormente enviados para o profissional, podendo este acompanhar o estado do paciente entre visitas e até mesmo antecipar consultas, caso detete alguma irregularidade que necessita de ser averiguada o mais cedo possível.

Apesar do vasto leque de vantagens apresentados, também são grandes os riscos que a implementação destes sistemas apresentam e que precisam de ser tomados em conta, no momento de desenvolvimento de um [SADC](#).

### 2.2.4 Dificuldades e riscos no desenvolvimento e adoção de SADCs

Os [SADCs](#) podem constituir uma ferramenta de auxílio ao profissional de saúde quando devidamente desenvolvidos e implementados. No entanto, quando estes não são devidamente implementados podem apresentar riscos que, por último, podem comprometer o trabalho do profissional de saúde, resultando num declínio dos cuidados prestados.

Um dos riscos refere-se à interrupção do *workflow* por parte destes sistemas. Isto deve-se a [SADCs](#) que necessitem da inserção contínua de documentação ou informações. Por sua vez, a sua utilização traduzia-se num aumento de tempo dispendido a realizar as tarefas requeridas e, em oposição, no declínio de tempo utilizado para estar com o paciente. Estudos mostram que os profissionais de saúde com mais experiência têm menos probabilidade em utilizar [SADCs](#) e a seguir as indicações [14].

Outro tipo de problema que surge refere-se à geração de alertas excessivos, inapropriados e/ou irrelevantes. Estes factores podem induzir à desconfiança por parte do profissional, resultando na ignorância destes alertas e na desconfiança da sua veracidade, podendo ainda comprometer a eficiência do seu trabalho.

Um factor que se deve ter em conta é também o impacto da utilização dos [SADCs](#) ao longo do tempo. A sua utilização frequente pode levar à dependência por parte dos utilizadores no seu uso aquando a realização de uma determinada tarefa. Por consequência, esta dependência pode-se refletir no declínio da autonomia por parte do profissional de saúde que não se sente capaz em realizar uma determinada tarefa sem o auxílio destes sistemas. Este facto também levanta outro problema que corresponde à demasiada confiança depositada nos [SADCs](#), acarretando a falsa ideia que não é necessário verificar as indicações geradas por estes.

Alguns [SADCs](#) são demasiados técnicos no uso das terminologias e a sua utilização depende muito da literacia tecnológica do utilizador. Este por vezes, apresentam um curva de aprendizagem longa, resultando na sua fraca adoção por parte dos utilizadores.

A manutenção dos [SADCs](#) apresenta em si uma dificuldade, na medida em que é necessário manter estes sistemas o mais atuais e precisos possíveis. À medida que o conhecimento evolui, é necessário atualizar as base de conhecimento, o que dependendo do [SADC](#), pode representar um operação complexa.

Com o surgimento de novas práticas e guidelines clínicos, surge também a necessidade de atualizar o motor de inferência e de, assim, produzir e/ou alterar regras [62].

Um dos problemas assente nos SADCs é a presença de dados deficientes e ainda a presença de conteúdo incorreto. Como estes sistemas obtêm dados de diferentes fontes heterogéneas, nem sempre os dados obtidos são os mais precisos e/ou correctos. Consequentemente, a qualidade dos dados podem impactar diretamente a qualidade do SADC, comprometendo, deste modo, o componente de apoio à decisão.

Outro problemas emergem, como a falta de transportabilidade e interoperabilidade, isto é, a capacidade de serem integrados noutros sistemas sem que o significado do seu conteúdo se perca. Habitualmente, o cerne da questão está na ausência da utilização de padrões/ linguagens padronizadas reconhecidas pela comunidade científica.

Por fim, um dos principais entraves à adoção dos SADCs por parte dos sistemas de saúde diz respeito aos elevados custos que o seu desenvolvimento, manutenção e integração apresentam. Até 74% das entidades que possuem um SADC dizem que a viabilidade financeira permanece um desafio [28]. No entanto, ainda é difícil apurar até que ponto a viabilidade do sistema, em termos financeiros é colocada em questão, visto que as análises de custo de implementações de SADCs são díspares.

## 2.3 Modelo OpenEHR

### 2.3.1 Definição

O openEHR é uma padrão *e-health* aberto que, de modo sucinto, oferece um conjunto de especificações, modelos clínicos e *softwares* que visam o desenvolvimento de registos clínicos em módulos capazes de realizar operações entre si [2]. Além disso, este tipo de abordagem possibilita o acesso gratuito aos dados e especificações relativas a informações clínicas, sendo utilizado para a gestão, armazenamento e consulta de EHR [24].

As especificações são geridas pela Organização Internacional openEHR sem fins lucrativos. Esta também é responsável por disponibilizar as ferramentas necessárias para o uso da norma. O openEHR é composto também por uma comunidade virtual responsável por produzir os artefactos. O principal foco prende-se em possibilitar a construção de sistemas de registos eletrónicos de saúde que sejam capazes de comunicar entre si sem que haja perda de significado de conteúdo, garantindo assim a interoperabilidade semântica [2].

### 2.3.2 Como surgiu openEHR

O openEHR surgiu de um projeto de investigação internacional financiado pela União Europeia designado por *Good European Health Record* - GEHR. O seu desenvolvimento iniciou-se nos anos 90 e tinha como

principal objetivo desenvolver especificações para registos eletrónicos de saúde, baseando-se em detalhes clínicos e técnicos [2]. Este projeto envolveu vários países europeus, tendo produzido no total mais de 2000 páginas de documentação de domínio público [2]. Contudo, a sua adoção em massa foi condicionada pelas dificuldades de implementação que apresentava na altura e também por inúmeras razões técnicas.

Em 1997 um grupo de australianos liderado por dois antigos membros do GEHR (Sam Hear e Thomas Beals) uniram-se à Universidade Colleague London - UCL - com a finalidade de superar as adversidades presentes na implementação do GEHR [2]. Consequentemente, o grupo de australianos foi capaz de desenvolver uma abordagem inovadora conhecida pela modelação em dois níveis, que separava os dados clínicos dos dados demográficos. Esta abordagem baseava-se em arquétipos e apresentou uma solução para os problemas da implementação do GEHR.

Mais tarde o grupo de australianos fundou a Ocean Informatics e em 2002 o trabalho desenvolvido pela UCL paralelamente com o trabalho desenvolvido pela Ocean originou o novo modelo openEHR [2]. Posteriormente, foi criada a Fundação openEHR e a partir dela o modelo openEHR foi sofrendo consecutivos melhoramentos até aos dias de hoje. Atualmente, o openEHR é uma organização sem fins lucrativos que conta com a colaboração de uma vasta comunidade internacional, com o objetivo de criar EHRs clinicamente abrangentes e interoperáveis [42].

### **2.3.3 Visão Geral da Arquitetura OpenEHR**

A arquitetura openEHR é resultante de mais de uma década de pesquisa de varidíssimos projetos e *standards* por todo o mundo, tendo sido desenhada pelos requisitos recolhidos durante estes anos [18].

Seguindo o paradigma orientado ao arquétipo, esta arquitetura torna-se deste modo bastante genérica, o que faz com que possa ser utilizada em diversas situações. No entanto, a característica chave do openEHR e a que permitiu acompanhar a evolução do conhecimento de modo otimizado é a utilização da metodologia dual-model/multinível para descrever os dados dos registos eletrónicos de saúde [2].

A metodologia de modelagem de informação em dois níveis, onde o modelo de dados clínicos é desenvolvido numa camada diferente do modelo de referência, facilita a separação de questões clínicas e técnicas, permitindo, assim, um desenvolvimento mais independente de cada um desses domínios [43] (1). Por consequência, os sistemas que integrem este modelo são agora capazes de acomodar alterações em conceitos clínicos, sem que haja um trabalho acrescido destes. Para além disso, permite que os profissionais de saúde definam como pretendem capturar a realidade sem a necessidade de redefinir o modelo de informação, que é estável [46].

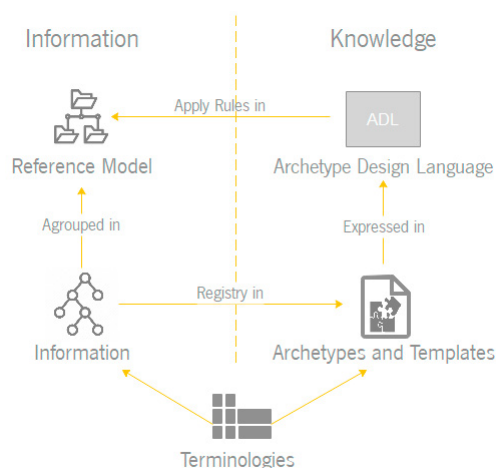


Figura 1: Arquitetura OpenEHR (retirado de [24])

O primeiro nível é composto pelo Reference Model -RM - e pelo modelo de serviços. O RM engloba e define as informações que são processadas para cada paciente no sistema e, para além disso, contém os tipos de dados que podem ser utilizados para representar a informação como dados de texto, dados quantitativos, datas, entre outros. A utilização de tipos de dados reduz significativamente a variação de definições de conteúdo dos sistemas de informação, possibilitando a construção de sistemas muito mais compactos e de fácil manutenção, quando se compara com os sistemas que utilizam um único nível de modelação [2, 24].

Por sua vez, o segundo nível é constituído pelas definições de conhecimento clínico, que se encontram sob a forma de *templates* e arquétipos, promovendo deste modo a interoperabilidade semântica destes sistemas.

## 2.4 Linguagens de Decisão Padronizadas existentes no mercado

Vários são os estudos que provam que a utilização de sistemas de apoio à decisão clínica melhoram o serviço de prestação de cuidados do paciente, principalmente quando estes utilizam sistemas de informações como os EHR. Neste sentido, a integração de *guidelines* computadorizados nestes sistemas visa culmar com muitos preconceitos que existem relativamente à utilização dos *guidelines* tradicionais, que mostravam-se ser difíceis de escrever e de aplicar no quotidiano de trabalho do profissional.

A utilização de *guidelines* para representar planos de cuidados de saúde, protocolos de configurações clínicas e *pathways* clínicos podem reduzir a variabilidade das práticas existentes e dos custos de tratamento dos pacientes, melhorando simultaneamente a qualidade dos cuidados prestados.

Assim, a necessidade de utilizar uma forma padronizada para representar estas entidades é crucial,

existindo inúmeras abordagens que permitem codificar o *guideline* sob a forma de artefacto computacional. Seguidamente, irá se abordar algumas linguagens de decisão padronizadas existentes no mercado, procurando-se apresentar uma explicação detalhada para cada uma e apresentando por fim, uma visão crítica das abordagens estudadas.

### 2.4.1 Arden Syntax

A Arden Syntax é uma linguagem padronizada utilizada para representar, processar e partilhar conhecimento científico e clínico entre instituições diferentes [26]. Atualmente esta linguagem é facultada pelo Health Level Seven International, no entanto, surgiu em 1992 com a aprovação por parte da American Society for Testing and Materials [29].

Segundo a literatura [17, 26, 29], a arquitetura Arden encontra-se organizado em ficheiros independentes, designados por Medical Logic Modules (MLMs) onde são integrados o suporte à decisão e à lógica. Deste modo, o código de processamento que representa conhecimento clínico e a lógica do programa (conteúdo) é separado de código mais técnico como declaração de variáveis e ligações com outros serviços ou fontes externas de dados (recursos), melhorando deste modo a transparência do MLM. Relativamente ao **conteúdo** do MLM este encontra-se dividido nas seguintes categorias, que por sua vez estão divididas em *slots*:

- **Manutenção:**

Contém *slots* que especificam informações gerais relativas aos MLMs como a versão do Arden Syntax que foi utilizada, o autor, a data de criação, entre outros. Esta categoria serve para controlar possíveis alterações e para a manutenção da base de conhecimento.

- **Biblioteca:**

Contém as *slots* que são pertinentes para a manutenção da base de conhecimento e que são relativas ao conhecimento do MLM. Esta *slots* fornecem aos profissionais de saúde informação pré-definida explicativa e *links* para literatura científica, bem como detalhes acerca do propósito do MLM.

- **Conhecimento:**

Possui o algoritmo para a tomada de decisão, definindo também mecanismos de acesso à base de dados e eventos. Neste são definidos, por exemplo, os termos usados, o contexto em que o MLM deve ser invocado, condições de teste e ações que devem ser despoletadas por condições que se verificaram ser verdadeiras.

Relativamente à secção dos **Recursos**, esta contém o conjunto de *slots* de linguagem que especificam o recurso textual de onde o operador localizado deve ser desenhado de modo a obter o conteúdo da mensagem de diferentes linguagens. Cada *slot* de linguagem define um conjunto de pares chave-valor que representam constantes textuais numa linguagem específica.



Figura 2: Organização do Medical Logic Module (retirado de [25])

Ainda, as MLMs podem assim, interagir com o *host-system* no qual está integrado através:

- **Input:** um parâmetro de entrada que pode ser inserido através da invocação de outra MLM
- **Curly Brace Expressions:** define cláusulas de mapeamento do *host-system* para as variáveis locais dos MLMs.
- **Escrever expressões:** textos podem ser escritos para destinos que são mantidos pelo *host-system*.
- **Output:** alguns dados podem ser submetidos ao *host-system* através das MLMs, quando a execução destes é terminada.

Segundo a literatura [70], uma das principais limitações do Arden Syntax prende-se precisamente no uso da Curly Brace Expression({}), pois a cada reutilização de um MLM é necessário fazer um adaptação dos mapeamentos locais. Deste modo, não consegue assegurar **semânticamente** que os dados sejam uma representação do conjunto de dados pois existem significados diferentes de conceitos clínicos. Por conseguinte, o resultado dos sistemas de apoio à decisão que utilizem esta linguagem podem produzir inferências fracas e assim, um mau resultado.



## 2.4.2 Guideline Interchange Format GLIF

O GuideLine Interchange Format (GLIF) corresponde a uma linguagem estruturada para representar *guidelines* desenvolvida pelo InterMed Colaboratory, um projeto que teve como colaboradores grupos informáticos de biomedicina das Universidades de Harvard, Columbia e Stanford [41]. Os *guidelines* são modelados na forma de fluxograma, sendo constituído por passos estruturados, representando ações e/ou decisões. Os seus principais objetivos são permitir a sua partilha entre instituições diferentes, facilitar o acesso aos *guidelines*, possível apenas com o recurso a ferramentas de *software*, permitir alterações que satisfaçam as necessidades locais e obter, no fundo uma representação computacional que pode ser facilmente entendida por humanos.

### 2.4.2.1 GLIF 2

Relativamente à versão 2 do GLIF, apesar de ser capaz de produzir *guidelines* mais complexos que o Arden, esta versão apresentava inúmeras deficiências [6, 41], como:

- a não especificação formal de uma representação estruturada para atributos e passos. A maioria dos atributos eram indicados como *strings*. Por consequência, estes *guidelines* não poderiam ser interpretados de forma confiável pelo computador.
- a falta de construtores que suportam conceitos importantes como descrições de interações, estado do paciente, condições excepcionais, decisões alternativas e eventos.
- a carência de construtores que permitissem o mapeamento de dados dos pacientes para *electronic medical records*. Consequentemente, a integração do GLIF2 em sistemas clínicos heterogêneos era penosa.

### 2.4.2.2 GLIF 3

O GLIF3 foi solução apresentada face aos problemas levantados com a versão 2 do GLIF e que foram acima abordados. Entre as demais modificações apresentadas no GLIF3 [6], salientam-se as seguintes:

- representação da entidade *guideline* mais estruturada;
- adição de expressões orientadas aos objetos e linguagens de *query*;
- melhoramento do controlo de fluxo, na medida em que novas classes foram adicionadas para especificar o estado do paciente, iterações, eventos e exceções, assim como, classes já existentes foram revistas e reformuladas.
- adição de um modelo de camada que facilitou a visualização dos dados dos pacientes e o conhecimento médico durante a execução de uma *guideline*

- adoção de *open standards*, como por exemplo a utilização do padrão Health Level 7 (HL7) e também a incorporação de terminologias médicas padrão.

Assim, sendo o GLIF3 é uma linguagem de decisão padronizada e orientado ao objecto, este é composto por classes que visam a descrição de características inerentes ao guideline, e ainda é composto por atributos e tipos de dados específicos. Deste modo, cada objecto representa um guideline e contém, por sua vez, atributos de carácter descritivo (ex: nome do autor do *guideline*, propósito, entre outros). Assim, cada guideline é composto por um conjunto de passos interligados num grafo direcional, passos estes que podem ser os seguintes:

- **Decisão:** é partir deste que é realizado o controlo do fluxo bem como modelado os pontos de decisão. Cada passo que é do tipo decisão contém expressões de lógica de decisão. Assim, com base no resultado destas expressões é realizado o controlo de fluxo, que permite definir, essencialmente, a direção escolhida de um passo para as suas demais alternativas. Adicionalmente, pode fornecer
- **Estado do paciente:** contém atributos que descrevem o estado atual do paciente.
- **Ramificações e sincronizações:** corresponde a um conjunto de passos concorrentes que direcionam o fluxo para múltiplos passos do guidelines paralelos e que são utilizados em conjunto com os passos da sincronização. Isto significa, que múltiplos passos que constituem um passo do *branch* eventualmente convergem no correspondente passo de sincronização.
- **Ações:** tarefas que têm ou devem ser executadas. Existem vários tipos de tarefas, como por exemplo: recomendações, devolução de dados proveniente de EHR ou de mensagens, entre outros.

Como se pode constatar foram várias as modificações inseridas no GLIF3 que permitiram a possibilidade de codificar *guidelines* que possam ser interpretados por qualquer programa informático.

De um modo geral, o GLIF specification engloba dois componentes: o GLIF model e o GLIF syntax [6, 41]. O GLIF model consiste num conjunto de classes. Deste modo, um determinado *guideline* codificado em GLIF é uma instância do GLIF model. Resumidamente o GLIF model apresenta todas as ferramentas necessárias para representar um entidade de um *guideline*. Por sua vez, o GLIF syntax define o formato dos ficheiros de texto que contém a codificação GLIF do *guideline* [6].

Segundo a literatura [6], o GLIF3 veio facilitar a partilha de *guidelines*, podendo ser agora interpretados informaticamente. Adicionalmente, este tipo de sistema é capaz de representar eficientemente *guidelines* complexos e extensos [49].

### 2.4.3 PROforma

O PROforma é uma linguagem formal utilizada para descrever *guidelines*, *care pathway*, protocolos e para automatizar os processos clínicos [61, 20, 21].

No contexto PROforma, um *guideline* é sempre modelado como um conjunto de tarefas e itens de dados. As tarefas por sua vez são organizadas hierarquicamente em planos e são divididas em classes

Em termos sintáticos, esta linguagem segue o formalismo de Backus-Naur (BNF) pelo que forma desde já uma limitação visto que *guidelines* que não estejam em conformidade com esta sintaxe não podem ser executados. Contudo, em termos técnicos esta sintaxe pode ser dividida em duas partes [61, 20, 21]:

- a parte que se refere às expressões e que define a forma como a linguagem permite utilização de condições lógicas e expressões matemáticas;
- o resto que define como as definições das tarefas e outros componentes do *guideline* podem ser organizados e separados.

Relativamente à semântica, o PROforma representa os *guidelines* como um conjunto de objectos denominados de componentes - *Proforma Components*. Este faculta uma série de classes que por sua vez têm um conjunto de propriedades. É importante referir que cada classe herda as propriedades da sua super classe. Para além das classes, cada componente possui um identificador único, o que significa que não existem dois componentes com o mesmo id que pertençam a classes diferentes.

Relativamente às propriedades estas devem corresponder a um PROforma value, ou seja [61, 20, 21]:

- número real ou inteiro;
- uma string textual;
- uma expressão PROforma
- identificador do componente
- uma das constantes definidas na especificação PROforma que pode incluir: unknown, true, false, dormant, entre outros.
- uma sequência finita de valores.

A abordagem PROforma originou o aparecimento de novas tecnologias e tem sido usado para desenvolver inúmeras aplicações clínicas, tais como a CAPSULE que auxilia os profissionais de saúde na prescrição de medicamentos, o RAGs que avalia o risco de cancro através da genética entre muitos outros [61, 20, 21].

### 2.4.4 Decision Logic Module (DLM) - Suporte à decisão

O **DLM**, proposto pelo openEHR [58], codifica toda a lógica de decisão assente nos planos, *guidelines* e outras entidades clínicas baseadas em regras, sendo esta expressa através de regras, tabelas de decisão entre outros artefactos.

O facto de arquitetura openEHR utilizar estes módulos para expressar todas as decisões lógicas ao invés de serem expressas nos planos garante que todas as condições de decisão são tratadas como artefactos de conhecimento de primeira ordem ao contrário de serem escondidas como expressões *ad hoc* contidas no próprio plano, dificultando a tarefa de as encontrar e as manter [58].

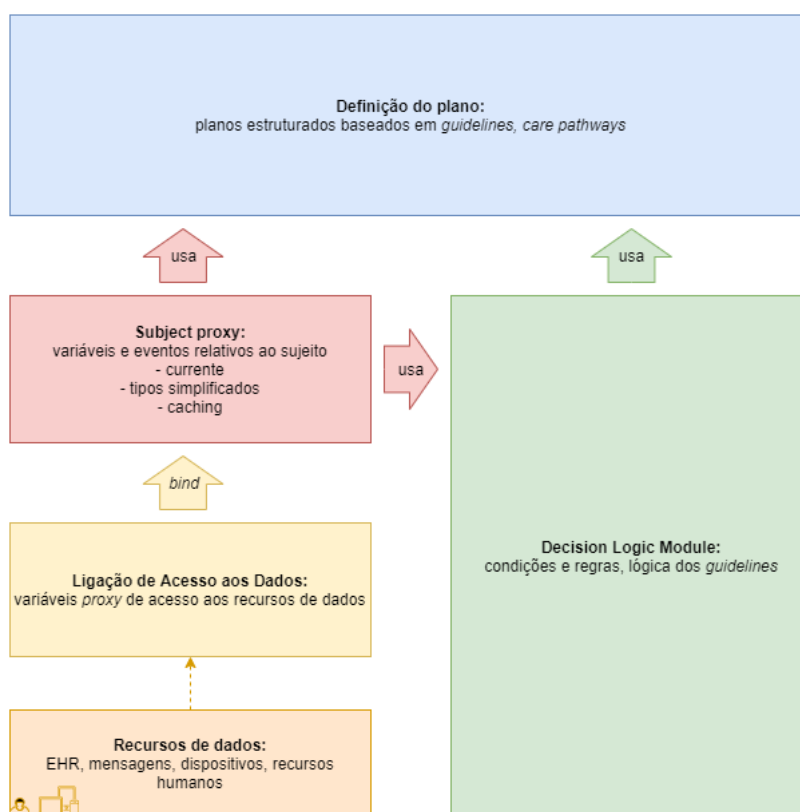


Figura 3: Arquitetura Decision Logic Module do openEHR (adaptado de [68])

A figura acima ilustrada 3 representa a arquitetura do Decision Logic Module no openEHR e a sua interação com os demais componentes que possibilitam assim a criação e execução de *guidelines*, entre outros artefactos clínicos.

Assim, a arquitetura 3 pode-se resumir nos seguintes componentes:

#### 2.4.4.1 Definição do Plano

Utilizado para expressar conjunto de tarefas estruturadas, designadas de *Work Plans*, segundo a especificação do openEHR Task Planning [60]. Resumidamente, este tipo de artefactos é executado com um

determinado objetivo aquando a indicação de um determinado agente, quer seja através da indicação por parte dos profissionais, quer seja através de softwares de carácter automatizado. Os planos ao serem executados apresentam **pontos de decisão** onde o plano se ramifica e pode seguir diferentes caminhos consoante condições e regras que em última instância dependem das **variáveis do sujeito**, facultadas pelo *Subject Proxy* [58]. Como já referido, todas as decisões lógicas são expressas e fornecidas pelo Decision Logic Module.

Segundo as especificações do openEHR [59], a definição do plano pode ser visto, do ponto de vista prático como a interface do utilizador. É nesta que devem estar contidos as definições dos *guidelines* estipulados pelo o utilizador da ferramenta. Por sua vez, a interface deve ser capaz de comunicar com o serviço do Subject Proxy de modo a obter as variáveis do sujeito desejadas. Adicionalmente, deve ser capaz de utilizar os serviços do DLM de modo a criar um artefacto estruturado e que represente o objecto criado em questão, sendo o mais interessante para o caso de estudo o *guideline*.

### 2.4.4.2 Decision Logic Module

Como já supracitado anteriormente, este componente é responsável por expressar regras, tabelas de decisão e não só, que em última instância codificam a lógica de decisão dos *guidelines* presentes no componente Definição do Plano [58]. Desta forma, os DLMs podem ser usados para expressar desde as condições booleanas mais simples, assim como entidades clínicas mais complexas. Contudo, para isto ser possível, os DLMs necessitam de declarar **variáveis do sujeito**, mais um vez facultadas pelo *Subject Proxy*, precisas para computar as regras. O Decision Logic Module inclui os seguintes elementos:

- **Variáveis de Input:** declarações de variáveis referentes ao sujeito que estão contidas nas condições e regras. Estas podem ser obtidas através do Serviço do Subject Proxy
- **Condições:** condições de valor booleano diretamente relacionadas com as variáveis de sujeito;
- **Regras:** conjunto de condições, complexas que geram *outputs* ou ações;
- **Conjunto de regras:** coleções de regras que operam num conjunto de variáveis de input que geram um conjunto de valores de output;
- **Variáveis de output:** resultados das regras intermediários que podem ser invocados por outros componentes.

### 2.4.4.3 Subject Proxy

Este representa conceptualmente a coleção de variáveis de input necessárias para o plano e/ou o DLM. É, portanto, um visão próxima do sujeito no seu estado atual. Assim, neste serão definidas e expressas as variáveis de input, que representam uma visão ontica do sujeito, isto é, o mais próxima possível do seu estado real atual. Adicionalmente, possibilita o encapsulamento da linguagem de domínio para as

variáveis do sujeito o que torna a lógica de decisão compreensível para os especialistas [58]. O Subject Proxy precisa, no entanto, de extrair os dados necessários para representar o estado atual do sujeito e para tal necessita de recorrer ao componente de Ligação de Acesso aos Dados para tal.

#### **2.4.4.4 Ligação de Acesso aos Dados**

Também designado por Subject Proxy Service, este componente trata de realizar e gerir a extração do estado do sujeito das várias fontes disponíveis, tais como, registos de saúde, sistemas laboratoriais, dispositivos de monitorização, entre outros.

### 2.4.4.5 Sintaxe do Decision Logic Module

```

dlm <form> <identifier>

definitions -- Descriptive
  |
  | lang and translations meta-data, as for archetypes
  |
  language = {
    original_language: [ISO_639-1::en],
    translations: {...}
  }
  ;

  |
  | description meta-data, as for archetypes
  |
  description = {...}
  ;

[use_model
  <reference model ids; defaults to openEHR Foundation & Base types>]

[use
  <local identifier>: <dlm identifier>
  <local identifier>: <dlm identifier>]

[preconditions
  <conditions that act as pre-conditions for this module>]

[definitions -- Reference
  <constant definitions>]

input -- Administrative State
  <administrative subject variables>

input -- Historical State
  <historical subject variables>

input -- Tracked State
  <real-time subject variable>

rules -- Conditions
  <conditions expressed as EL functions>

rules -- Main
  <computational rules expressed as EL functions>

rules -- Output
  <output rules expressed as EL functions>

definitions -- Terminology
  terminology = {...};

```

Figura 4: Estrutura openEHR do DLM (retirado de [59])

Na figura acima representada pode-se observar a estrutura geral de um Decision Logic Module que é utilizado para representar o *guideline* (e outros artefactos) no contexto computacional. Como se observa são várias as secções presentes neste, pelo que torna-se imperativo estudar cada uma.

Assim, segundo o openEHR specification program [59] temos as seguintes secções:

### 2.4.4.6 Secção de Identificação

A secção de identificação corresponde à figura de seguida apresentada:

```

dlm <form> <identifier>

```

Figura 5: Secção de identificação (retirado de [59])

Nesta associam-se parâmetros descritivos que permitem identificar o tipo de DLM implementado e também parâmetros de carácter identificativo, mencionando aditivamente a versão do DLM. É a partir desta secção que é possível utilizar um determinado DLM noutro contexto.

#### 2.4.4.7 Secções de Linguagem e Descrição:

```
definitions -- Descriptive TS
  language = {
    original_language: [ISO_639-1::en]
  }
  ;

  description = {
    lifecycle_state: "unmanaged",
    original_author: {
      name: "Thomas Beale",
      email: "thomas.beale@openEHR.org",
      organisation: "openEHR Foundation <http://www.openEHR.org>",
      date: "2021-01-10"
    },
    details: {
      "en" : {
        language: [ISO_639-1::en],
        purpose: "To record an individual's QRISK3 score."
      }
    },
    copyright: "@ 2021 openEHR Foundation",
    licence: "Creative Commons CC-BY <https://creativecommons.org/licenses/by/3.0/>",
    ip_acknowledgements: {
      "ClinRisk" : "This content developed from original publication of
        © 2017 ClinRisk Ltd., see https://qrisk.org",
      "QRISK" : "QRISK® is a registered trademark of the University of Nottingham and
        EMIS"
    }
  }
  ;
```

Figura 6: Secção de Linguagem e Descrição (retirado de [59])

A secção de Linguagem e Descrição diz respeito a parâmetros como a linguagem utilizada no DLM, aspectos descritivos referentes ao autor em questão como o nome, o email, licenças associadas, entre outros fatores.



#### 2.4.4.8 Secção dos Identificadores

```

definitions -- Terminology
{
  terminology = {
    term_definitions: {
      "en": {
        "resting_heart_rate": {
          text: "heart rate at rest",
          description: "...",
        }
        ...
        "SpO2": {
          text: "Oxygen saturation";
          description: "...",
        }
      }
    }
  }
}

```

Figura 7: Secção dos Identificadores (retirado de [59])

A secção dos identificadores é opcional, sendo que inclui definições linguísticas referentes aos identificadores que são caracterizados por serem uma *string* sem espaços em branco. Como podemos ver através da análise da figura 7, cada identificador possui um campo de texto - **text** - que permite definir por extenso o identificador e ainda um campo de descrição - **description** - que possibilita acrescentar uma breve descrição relativa ao identificador. Caso esta secção seja definida, estes identificadores podem ser depois utilizados nas regras.

#### 2.4.4.9 Secção do Use\_model

```

use
  BSA: Body_surface_area.v1

```

Figura 8: Secção do Use\_model (retirado de [59])

Esta secção permite que o DLM presente especifique um modelo que define o tipo do sistema para o DLM corrente. Para tal utiliza-se o identificador do modelo que deve seguir normas específicas. Caso não se defina nenhum modelo, o DLM irá assumir por defeito o openEHR Base and Foundation types [44].

#### 2.4.4.10 Secção das Pré-Condições

Esta secção é utilizada para estabelecer condições de carácter lógico que têm de ser avaliadas como verdadeiras de modo a que o DLM definido possa ser utilizado para o sujeito em questão [58]. As condições de carácter lógico podem ser condições ou regras previamente criadas. A figura seguinte, apresenta um exemplo de uma pré-condição definida nesta secção.

```
preconditions
  is_pregnant
```

Figura 9: Secção das Pré-Condições (retirado de [59])

#### 2.4.4.11 Secção das Referências

Nesta, são estabelecidos constantes, que são no fundo identificadores com um valor fixo associado. Aditivamente, nesta secção podem ser delineados identificadores simples como o que consta na figura 10, mas também podem ser delineados identificadores complexos, como o que consta na figura 12.

```
definitions -- Reference

paracetamol_dose: Quantity = 1g;
chlorphenamine_dose: Quantity = 10mg;
prednisolone_dose_per_m2: Quantity = 40mg;
```

Figura 10: Secção das Referências (retirado de [OpenEHRSpecificationProgram2])

```
definitions -- Reference TS

Smoking_interaction_scales = {
  [female]: {
    [age_1]: {
      [non_smoker]: 0,
      [ex_smoker]: -4.70571617858518910,
      [light_smoker]: -2.74303834035733370,
      [moderate_smoker]: -0.866080888293921820,
      [heavy_smoker]: 0.902415623697106480
    },
    [age_2]: {
      [non_smoker]: 0,
      [etc]: -0.07558924464319302600000000
    }
  },
  [male]: {
    [age_1]: {
      [non_smoker]: 0,
      [etc]: -0.21011133933516346000000000
    },
    [age_2]: {
      [non_smoker]: 0,
      [etc]: -0.00049854870275326121000000
    }
  }
}
;
```

Figura 11: Tabela de Riscos, presente na Secção das Referências (retirado de [59])

#### 2.4.4.12 Secção de Input

Aglomera todas as variáveis do Subject Proxy, relativas a um único sujeito, utilizadas pelo DLM. Um exemplo de uma definição de uma variável do sujeito pode ser observada na seguinte figura.

```
input
  heart_rate: Quantity
```

Figura 12: Atribuição da variável do Subject Proxy, presente na Secção de Input (retirado de [59])

#### 2.4.4.13 Rastreamento de Variáveis

Inserida na Secção de *Input*, as *Tracked Variables*, ou em português, o Rastreamento de variáveis, possibilita a obtenção automática do valor de uma dada variável do *Subject Proxy*, por parte dos sistemas de *back-end*. Cada variável, pode possuir informação relativamente à sua atualidade, isto é, o quão recente deve ser para que o DLM possa ser válido. Para tal define-se no campo, denominado de **currency**, este valor. O uso deste modificador permite delinear que uma determinada variável do sujeito está relacionado com o tempo, que por conseguinte está relacionada com um indicador que representa uma representação do estado atual no sujeito no mundo real (ex: colesterol) que diz respeito a um determinado sujeito [59].

```
input -- Administrative State
|
| untracked variable:
| DOB never changes, no currency needed
|
| date_of_birth: Date
|
| ;

input -- Historical State
|
| tracked variable:
| weight changes over a period of days
|
| weight: Quantity
|   currency = 3 days
|   ;
|
| untracked variable:
| assuming an adult subject, height constant
|
| height: Quantity
|
| ;

input -- Tracked State
|
| tracked variable:
| blood glucose changes within minutes in response to food
|
| blood_glucose: Quantity
|   currency = 15 min
|   ;
|
| tracked variable:
| Heart-rate may change quickly
|
| heart_rate: Quantity
|   currency = 5 sec
|   ;
```

Figura 13: Secção de Input (retirado de [59])

Como se consta através da análise da figura 13, são várias a subsecções apresentadas na Secção

de *Input*. Adicionalmente, consegue-se identificar claramente as variáveis de rastreamento, que contêm o modificador *currency* definido, em oposição das variáveis do sujeito que não são rastreadas e cujo modificador *currency* não se encontra definido.

Para além do modificador *currency*, estas variáveis podem definir intervalos - **ranges** -, apresentando desta forma uma estrutura mais complexa e que pode ser observada na seguinte imagem:

```
input -- Tracked State KOTLIN

PaO2_FiO2_ratio: Quantity
  currency = 1 min,
  ranges["mm[Hg]"] =
    -----
    | 400 |:      [normal],
    | 300 .. 399 |: [low],
    | 200 .. 299 |: [very_low],
    | 100 .. 199 |: [extremely_low],
    | <100 |:     [critical_low]
    -----
;
ranges["kPa"] =
    -----
    | 53 |:      [normal],
    | 39.9 .. 53 |: [low],
    | 26.6 .. 39.8 |: [very_low],
    | 13.3 .. 26.5 |: [extremely_low],
    | <13.3 |:     [critical_low]
    -----
;
```

Figura 14: *Tracked Variables* com o modificador *range* (retirado de [59])

#### 2.4.4.14 Secção das Regras

Esta é sem dúvida a secção mais importante, visto que contém as regras definidas no *DLM*, e pode ser dividida em vários grupos, dependendo da necessidade. Assim, as regras podem variar desde expressões mais simples que retornam unicamente um *output*, como podem ser apresentadas na forma de expressões complexas, utilizando condições e operadores lógicos ( disjunção ou conjunção). Na figura seguinte pode-se observar as diversas formas de definir regras.

```
rules -- Conditions

her2_positive:
  Result := her2_expression = [positive]
;

non_class_I_heart_failure:
  Result := has_heart_failure_class_II or
            has_heart_failure_class_III or
            has_heart_failure_class_IV
;

anthracyclines_contraindicated:
  Result := has_transmural_MI or
            ejection_fraction.in_range ([low]) or
            non_class_I_heart_failure
;
```

Figura 15: Exemplo de uma Secção de Regras (retirado de [59])

#### 2.4.4.15 Secção das Terminologias

```

definitions -- Terminology
terminology = {
  term_definitions: {
    "en": {
      "date_of_birth": {
        text: "Date of birth"
      },
      "age_in_years": {
        text: "Age (years)"
      },
      "qRisk_score": {
        text: "QRISK2 score"
      },
      "diabetes_status": {
        text: "Diabetic status of subject"
      },
      "no_diabetes": {
        text: "Non-diabetic"
      },
      "type1_diabetes": {
        text: "Has type 1 diabetes"
      },
      "type2_diabetes": {
        text: "Has type 2 diabetes"
      }
    }
  }
}

value_sets: {
  "diabetes_status": {
    id: "diabetes_status",
    members: ["no_diabetes", "type1_diabetes", "type2_diabetes"]
  }
}
;

```

Figura 16: Exemplo de uma Secção das Terminologias (retirado de [59])

A figura acima apresentada corresponde a um exemplo do que a secção das Terminologias pode conter. Sucintamente, esta secção contém definições dos termos utilizadas no DLM.

#### 2.4.5 Visão Crítica das Linguagens Abordadas

Anteriormente, foram abordadas linguagens de decisão padronizadas que se acharam relevantes para o caso em questão. No entanto, é necessário fazer uma revisão crítica das mesmas, de modo a justificar a decisão de utilizar o modelo openEHR.

Assim, anteriormente foram abordadas as linguagens Arden Syntax, GLIF, PROforma e ainda o módulo de Decision Language do openEHR. Algumas destas acarretam dificuldades ou carecem de certas funcionalidades que em última instância foram um fator decisivo na sua não adoção.

Linguagens como o Arden, GLIF e PROforma ao invés de serem declarativas, como o Decision Language do openEHR, são de carácter procedimental [58]. Ainda, certas linguagens como o GLIF3 foram especificadas para um modelo particular desenhado para representar mensagens. Muitas destas, como o Arden, o GLIF e o ProForma carecem da integração de formalismos relativos aos planos e workflows [58]. Para além disso, nenhuma destas suporta uma representação integrada dos diferentes módulos

referentes à decisão lógica, aos dados proxy, ao acesso a dados e ainda à representação dos processo. Aditivamente, carecem de uma comunidade de apoio vasta, sendo o seu suporte limitado ou não especificado [58].

Contudo, o openEHR procurou estudar bem estas tecnologias e face aos problemas apresentadas pelas mesmas desenvolveu uma linguagem de decisão que em última instância culmatou com as limitações acima inumeradas.

# Abordagens Metodológicas e Ferramentas Utilizadas

Neste capítulo irá se abordar as metodologias literárias que suportaram o desenvolvimento deste projeto, bem como as ferramentas e tecnologias utilizadas para o desenvolvimento do presente trabalho.

## 3.1 Design Science Research

De acordo com a literatura "para garantir que uma pesquisa seja reconhecida como sólida e potencialmente relevante, tanto pelo campo acadêmico como pela sociedade em geral , esta deve demonstrar que foi desenvolvida com rigor e que é passível de debate e verificação"[33],. Tendo isto em conta é necessário adotar uma metodologia de pesquisa robusta de modo a alcançar o sucesso na realização do estudo.

Assim, a metodologia Design Science Research (DSR) permite estabelecer um raciocínio para a realização das pesquisas científicas, possibilitando a sua inserção no desenvolvimento de sistemas de informação [51]. Esta inserção é fundamental pois permite abordar questões e casos práticos já realizados no ramo e devidamente justificados.

Deste modo, a abordagem DSR pode ser representada graficamente do seguinte modo:

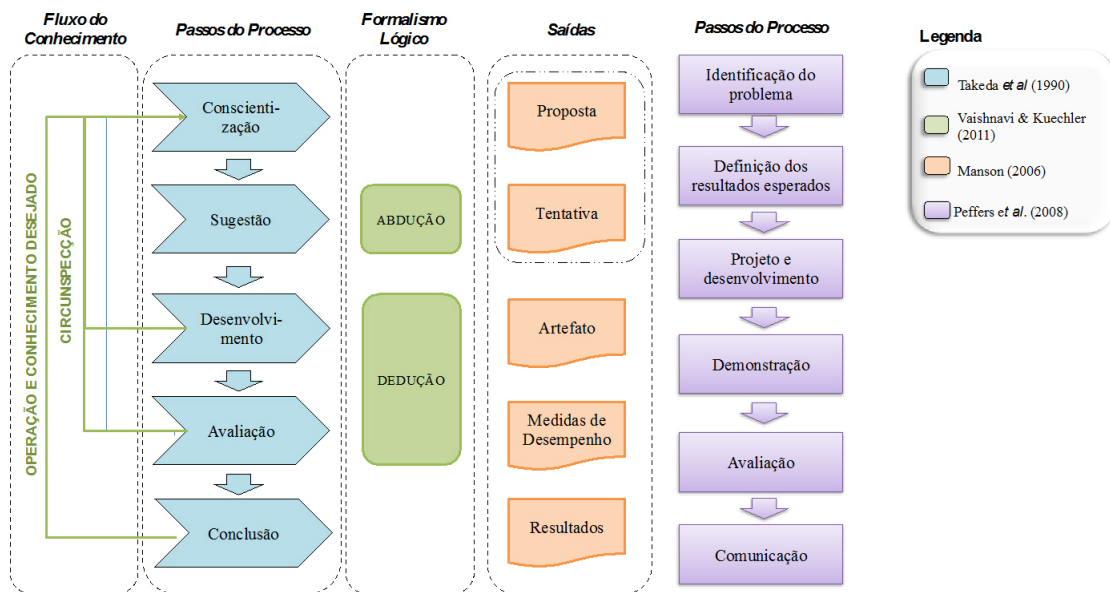


Figura 17: Modelo da metodologia DSR (retirado de [33])

A figura acima apresentada resulta da articulação das contribuições [63, 30, 36, 48], e pode ser descrita de uma forma sucinta, da seguinte maneira:

- **Identificação do problema/Conscientização** - corresponde à fase de compreensão da problemática presente. Nesta procura-se identificar o problema, bem como averiguar a relevância da sua solução proposta.
- **Definição dos resultados esperados/Sugestão** - baseando-se na descrição do problema bem como no conhecimento adquirido, pretende-se desenvolver uma ou mais alternativas de artefacto que resultem num resultado melhor aos já existentes
- **Projeto e desenvolvimento/Desenvolvimento** - corresponde à fase de produção do(s) artefacto(s), tendo como base o conhecimento teórico adquirido ao longo das etapas anteriores. No caso particular desta dissertação, corresponde à fase de desenvolvimento do sistema proposto.
- **Demonstração** - corresponde à fase de publicação do(s) artefacto(s) desenvolvidos, incentivando a sua utilização em contextos apropriados como casos de estudos, simulações, entre outros.
- **Avaliação** - consiste num processo rigoroso de verificação do comportamento do(s) artefacto(s) no ambiente para o qual foi projetado. Portanto, compara-se os objetivos iniciais delineados com o resultado final. Consequentemente, pretende-se apurar a sustentabilidade da solução, bem como, o desempenho do sistema, ou até mesmo o nível de satisfação do utilizador final.
- **Comunicação/Conclusão** - uma vez finalizadas as etapas apresentadas anteriormente, é necessário formalizar o processo e proceder para a sua comunicação às comunidades académicas e profissionais.



## 3.2 Tecnologias

Nesta secção irá se discutir as tecnologias mais revelantes que foram utilizadas para o desenvolvimento do sistema em questão. Por conseguinte, procurar-se-á enumerar as tecnologias e para cada uma indicar uma breve descrição da mesma e justificar a sua utilização.

### 3.2.1 MERN Stack

MERN Stack é uma "stack" Javascript utilizada no desenvolvimento rápido e fácil de aplicações web [23]. A descrição para o seu acrónimo é MongoDB, ExpressJS, ReactJS e NodeJS, um conjunto de tecnologias *open source* que oferecem uma vasta comunidade de suporte.

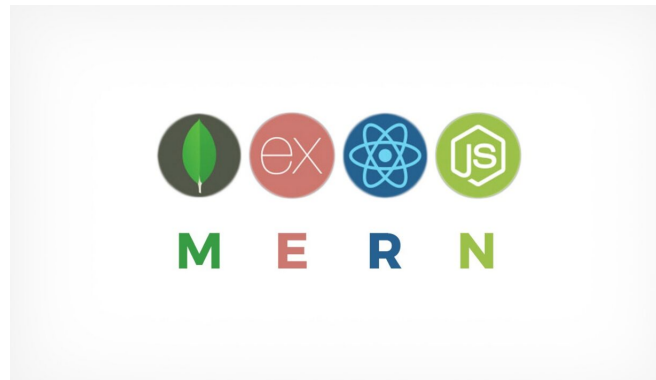


Figura 18: Mern Stack (retirado de [39])

A forma como estas tecnologias se organizam possibilita a construção de uma arquitetura *Three-Tier*, que será abordado no capítulo seguinte e, cuja a sua organização pode ser visualizada na seguinte figura 19

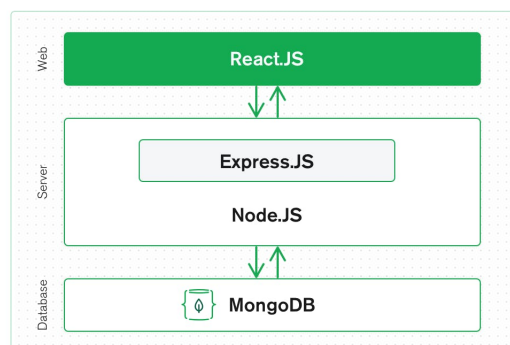


Figura 19: Arquitetura Mern Stack (retirado de [39])

Como podemos observar através da análise da figura 19, as tecnologias organizam-se em diferentes "camadas" fisicamente e computacionalmente independentes.

### 3.2.1.1 React.JS para o front-end

Sendo React uma framework javascript utilizada para criar interfaces para o utilizador, a sua forma declarativa de escrever o código permite que este seja mais previsível e mais fácil de detetar bugs [54, 39].

Adicionalmente, é orientada aos componentes o que significa que permite o encapsulamento dos próprios, gerindo eles próprios o seu estado. Assim, a criação de componentes é simplificada com o auxílio do *JSX*, uma extensão *HTML / XML* utilizada para escrever de forma fácil componentes React [23]. A utilização destes componentes permite criar aplicações *web* de rápido entendimento e promove, ainda, a reutilização de código. Para além disto, é relativamente simples criar aplicações Android/ios através do React Native, bastando apenas ter conhecimentos de JavaScript e React. Este fator, possibilita a fácil migração da aplicação *web* para *mobile*.

React é uma *framework* que proporciona uma elevada performance, pois tira partido do *Virtual DOM*, componentes e *JSX*, tornando-o mais rápido que a maioria das aplicações existentes [23]. Este facilita meios para construir interfaces complexas através de componentes simples, conectá-los aos dados no servidor *back-end* e renderizá-los em HTML [54].

### 3.2.1.2 Express.JS e Node.JS para o back-end

O Node.JS caracteriza-se por sendo um ambiente de execução Javascript assíncrono e orientado ao evento, que foi desenhado para criar aplicações escaláveis [40]. Este possibilita a criação de aplicações do lado do servidor *back-end* apresentando uma performance notória, na medida em que otimiza as taxas de transferências e a escalabilidade [8]. Como utiliza a linguagem de programação JavaScript, permite o decréscimo do tempo gasto para lidar com alterações de código entre o servidor e o navegador, visto que a linguagem é a mesma. Para além disso, oferece uma infinidade de pacotes que visam auxiliar o desenvolvimento.

O Express.JS, por sua vez, é uma *framework server-side*, que se encontra a correr dentro do servidor Node.JS [39]. Este contém modelos de roteamento URL bastante poderosos, sendo o responsável pelo tratamento de pedidos e respostas HTTP [39]. O ExpressJS facilita, ainda, a criação de APIs robustas, proporcionando eficiência, escalabilidade e respostas rápidas. As sua forma minimalista concede aos desenvolvedores liberdade para criar pacotes de *middleware*, conforme a necessidade. Adicionalmente oferece bibliotecas que facilitam o trabalho com *cookies*, sessões, login de utilizadores, parâmetros de URL, dados de requisição em pedidos POST, cabeçalhos de segurança, entre muitos outros [8].

### 3.2.1.3 Base de Dados MongoDB

O MongoDB é uma base de dados NoSQL que armazena documentos com uma estrutura semelhante a objectos JSON [23]. Este oferece um elevado grau de flexibilidade na medida em que permite a criação

de esquemas, tabelas, base de dados e ainda a utilização de *queries* em [SQL](#), sendo uma funcionalidade inovadora e única em bases de dados NoSQL.

A escolha desta tecnologia prendeu-se, sobretudo, nas vantagens que esta oferece. MongoDB permite que grandes quantidades de dados sejam armazenados, oferecendo, deste modo, a fácil escalabilidade da [BD](#), pois os dados são distribuídos por diferentes máquinas [9], resultando numa maior capacidade de armazenamento e desempenho. Para além disso, oferece uma resposta rápida aos pedidos, uma consequência de armazenar documentos indexados universalmente. O facto de uma consulta retornar o documento na íntegra facilita imenso a análise/processamento de dados, visto que não existem transações e *joins*. A utilização de JavaScript e ainda o modo de armazenamento de dados sob a forma de objectos [JSON](#), facilita a sua utilização e integração em aplicações *full-stack*. Esta particular característica torna-se muito útil no contexto desta aplicação pois permite simplificar o processamento de dados, pois os documentos JSON criados no *front-end* são facilmente enviados para o *back-end* e este pode diretamente armazená-los na base de dados, sem acrescentar complexidade ou meios de processamento de dados.

O MongoDB, sendo uma base de dados não relacional, oferece uma melhor performance no que toca ao processamento de um grande volume de dados, na medida em que suporta uma baixa latência e um alto desempenho quando comparado com o modelo tradicional de base de dados relacionais.

### **3.2.2 Tecnologias Adotadas**

Adicionalmente às tecnologias explicadas anteriormente, foi necessário recorrer ao auxílio de outras, que permitir não só otimizar o processo de desenvolvimento, como também acrescentaram novas funcionalidades úteis como, por exemplo, tornar mais apelativo o *design* da aplicação. Deste modo, segue-se a tabela com as tecnologias utilizadas e uma breve descrição das mesmas.

Tecnologia	Descrição	Utilidade
ReactJS	Biblioteca Javascript	Auxilio na criação da interface do utilizador
TailwindCSS	Framework CSS	Oferece classes como flex, text-center, entre outros que permite compor facilmente o design desejado.
Aida-UI	Framework de frontend	Oferece componentes como TextInput, Button, entre outros, enriquecendo deste modo o design do website.
Axios	Cliente HTTP promise-based HTTP.	Permite ao Frontend estabelecer comunicação com o Backend e o servidor do Subject Proxy
React Icons	Biblioteca de icons	Disponibiliza icons que permitem enriquecer, a nível de design o website.
NodeJS	Runtime JS	Servidor do backend
ExpressJS	Framework NodeJS	Responsável pelo tratamento dos pedidos/respostas HTTP do servidor Frontend, sendo utilizado tambem para o roteamento das rotas no backend.
Mongoose	Ferramenta de modelação de objectos MongoDB	Oferece métodos (ex: criação de queries) que permitem a manipulação dos dados contidos na base de dados MongoDB
Cors	Cross-Origin Resource Sharing Middleware	Oferece cabeçalhos adicionais HTTP, permitindo à aplicação Web aceder a dados de um determinado dominio, sendo neste caso o servidor Backend.
Body-parser	Middleware NodeJs de processamento do body dos pedidos.	Processa o corpo dos pedidos que chegam num camada de middleware, disponibilizando a informação na propriedade req.body.

Tabela 1: Tabela Resumo de Tecnologias utilizadas

# Decision Maker

## 4.1 Requisitos

Um passo importante no desenvolvimento de uma aplicação web passa por fazer o levantamento de requisitos de forma minuciosa e objectiva. Este processo passou por várias fases como a identificação do objetivo para a solução proposta, pesquisa literária e a realização de reuniões para a angariação de requisitos.

Antes demais, é importante delinear o objetivo principal deste projeto como sendo a criação de guidelines clínicos utilizando como base o modelo openEHR. Com isto em mente, pretende-se implementar uma aplicação *web* - Decision Maker - que se foque essencialmente no requisito acima estipulado, mas também que seja flexível e que permita aplicar os demais requisitos que serão explicados ao longo desta secção. Para tornar este processo de levantamento de requisitos o mais claro e sucinto possível dividiu-se os requisitos em dois tipos: **funcionais** que descrevem as funcionalidades que o sistema final deve apresentar e os **não funcionais** que dizem respeito a aspetos não funcionais do sistema como, por exemplo, restrições ao sistema ou propriedades relevantes do mesmo.

### 4.1.1 Requisitos Funcionais:

- **Criação de guidelines:** a aplicação permite ao usuário criar guidelines que, neste caso, seguem as especificações estipuladas pelo [DLM](#) do openEHR .
- **Execução de guidelines:** a aplicação deve possibilitar ao usuário correr o guideline pretendido e devolver a resposta adequada consoante as regras definidas que devem ser testadas com os valores que foram inseridos pelo utilizador ou que foram adquiridos através das variáveis do sujeito que são facultadas pelo Serviço do Subject Proxy.
- **Adicionar novo sujeito ao Subject Proxy:** a aplicação deve facultar meios para o utilizador conseguir adicionar um novo sujeito ao Serviço Subject Proxy.

- **Eliminar sujeito do Subject Proxy:** a aplicação deve permitir ao usuário eliminar um sujeito da base de dados do Subject Proxy.
- **Adicionar nova variável do sujeito ao Subject Proxy:** facultar ao utilizador formas de adicionar novas variáveis do sujeito ao Subject Proxy.
- **Adicionar novo dataframe ao Subject Proxy:** permitir que utilizador crie novos dataframes e os adicione ao Serviço do Subject Proxy.
- **Adicionar novo binding:** permitir ao utilizador criar novos bindings e armazená-los no Subject Proxy

## 4.1.2 Requisitos Não Funcionais:

Para efeitos de simplicidade e objetividade, decidiu-se dividir os requisitos não funcionais em categorias que dizem respeito ao sistema: Guidelines e Subject Proxy.

### 4.1.2.1 Requisitos do Guideline:

Os Requisitos do Guideline dizem respeito às propriedades que o objecto Guideline deve conter, no contexto da aplicação em questão. O delineamento deste foi conseguido graças à documentação do openEHR, mais especificamente, o Decision Language Specification e ainda com a elaboração de reuniões periódicas que permitiram validarem os requisitos. Através da sua análise apurou-se os campos relevantes e necessários que cada guideline deveria conter.

- **Guideline deve ser um objecto universal no contexto da aplicação:** o que significa que o servidor front-end deve ter conhecimento do mesmo objecto Guideline que o servidor back-end e a base de dados.
- **Guideline deve ter uma secção Descrição:** esta secção deve conter parâmetros como o nome do *guideline* e a categoria e também dados referentes ao autor criador do *guideline* como: nome, email, instituição que pertence e por fim o propósito do mesmo.
- **Guideline deve conter um secção para as Pré-condições:** esta secção destina-se a definir condições lógicas que devem ser avaliadas como verdadeiras de modo a que o Guideline possa ser executado. Um exemplo de uma pré-condição pode ser estar grávida ou ser fumador, sendo que obviamente depende do propósito do guideline desenvolvido.
- **Cada pré-condição deve conter um campo que indique se é manual ou não:** este requisito é preciso pois as pré-condições tanto podem ser criadas manualmente, como podem ser condições previamente criadas. Assim, no caso de se pretender criar uma pré-condição este campo deve ser definido como verdadeiro, caso contrário deve ser definido como falso.

- **Cada pré-condição deve conter um campo que armazena a pré-condição:** quer seja de criação manual ou de associação com a condição, cada pré-condição deve alojar a condição criada/associada.
- **Guideline deve conter um conjunto de condições:** cada objecto *guideline* deve conter um conjunto de condições que permitem criar o conjunto de regras e ainda serem utilizadas como pré-condições.
- **Cada condição deve conter um nome associado:** a definição do nome de cada condição irá variar consoante a necessidade. Assim, se o utilizador pretender associar a condição a uma variável do Sujeito do Subject Proxy, este campo corresponderá ao nome da variável escolhida. Caso contrário terá de ser definido manualmente pelo utilizador.
- **Cada condição deve conter um tipo associado:** a cada condição está associado um tipo que pode ser: Numero, Texto ou um *Array*.
- **Cada condição deve conter um operador associado:** o operador da condição representa um operador de comparação que varia consoante o tipo definido. Deste modo, no caso de o tipo definido ser Número o conjunto de operadores é: igual, diferente, maior, maior/igual, menor e menor/igual. No caso do tipo definido ser Texto, os operadores definidos são: igual ou diferente. E, por último, no caso do tipo definido ser Array, os operadores disponíveis são: contido e não contido.
- **Cada condição deve conter um valor associado:** este valor, definido pelo utilizador, será depois utilizado para testar a condição com o valor do facto atribuído.
- **Cada condição deve conter um valor de facto associado:** o valor do facto pode ser obtido através da inserção manual por parte do utilizador, aquando a sua execução ou através do valor obtido a partir da variável do sujeito previamente definida.
- **Cada condição deve conter uma indicação se é manual ou não:** uma condição pode utilizar uma variável do sujeito do Subject Proxy ou não. No caso de utilizar deve ser definido a falso este campo, caso contrário deve ser definido como verdadeiro.
- **Guideline deve conter um conjunto de regras:** o *guideline* deve conter um conjunto de regras criadas pelo utilizador
- **Cada regra deve conter um conjunto de condições associado:** cada regra têm de possuir um conjunto de condições que foram previamente criadas.
- **Cada regra deve conter um operador lógico associado:** para cada regra deve ser definido um operador lógico que pode ser de conjunção ou disjunção. Este operador define como o conjunto de condições da respetiva regra serão testadas.

- **Cada regra deve conter um conjunto de eventos/outputs associados:** cada regra deve conter um conjunto de eventos/outputs definidos que serão gerados caso as condições se verificarem como verdadeiras.
- **Cada evento deve conter um campo que defina o tipo de evento:** este tipo é definido pelo utilizador e pode assumir qualquer valor.
- **Cada evento deve conter um campo que defina a mensagem do evento:** o utilizador define a mensagem que pretende ser gerada caso o conjunto de condições, associado à regra específica se verifique.
- **Cada guideline pode ter associado um e um só identificador do sujeito:** o identificador do sujeito é associado aquando a atribuição da variável do sujeito à regra.

#### 4.1.2.2 Requisitos do Subject Proxy

A aquisição dos requisitos para esta parte foi conseguida graças ao fornecimento da [API](#) por parte do colaborador que implementou este componente e também com a realização de reuniões por videochamada de forma a discutir dúvidas e delinear requisitos.

- **De modo a ser possível realizar pedidos à [API](#) do Subject Proxy é necessário enviar no header de Authorization o bearer token gerado pelo mesmo:** todos os pedidos realizados à [API](#) necessitam de enviar no cabeçalho do pedido um bearer token.
- **Para criação do sujeito é necessário atribuir um identificador e a categoria:** a [API](#) requer que se envie no corpo do pedido o identificador e categoria de modo a se criar um novo sujeito no Subject Proxy.
- **Para a criação de novas variáveis é necessário fornecer os seguintes campos:** identificador do sujeito para o qual se pretende associar a variável, nome da mesma, se é manual (true) ou não (false) e o identificador do *binding* relacionado com a variável.
- **Para a criação de um novo binding é necessário facultar:** o `rm_type` e o identificador.
- **Para a criação dos *dataframes* é preciso indicar primeiro o tipo:**
  - se for do tipo [API](#) é preciso fornecer o url, uma lista de parâmetros contendo cada um o nome e o conteúdo associado e ainda uma lista dos parâmetros de ligação cada um contendo o nome e o valor associados.
  - se for do tipo QUERY é necessário facultar o texto da query, o tipo de base de dados que pode ser `sil`, ou `tplan`, ou `opengestão`. É ainda necessário fornecer uma lista de parâmetros cada qual contendo o nome e conteúdo atribuídos.



## 4.2 Arquitetura

De modo a perceber o funcionamento e a estrutura deste projecto recorreu-se à construção de diversos diagramas que serão mostrados e explicados ao longo desta secção.

Sendo assim, com a finalidade de apresentar uma visão geral das funcionalidades que o website oferece ao utilizador desenvolveu-se o seguinte diagrama da arquitetura funcional do mesmo.

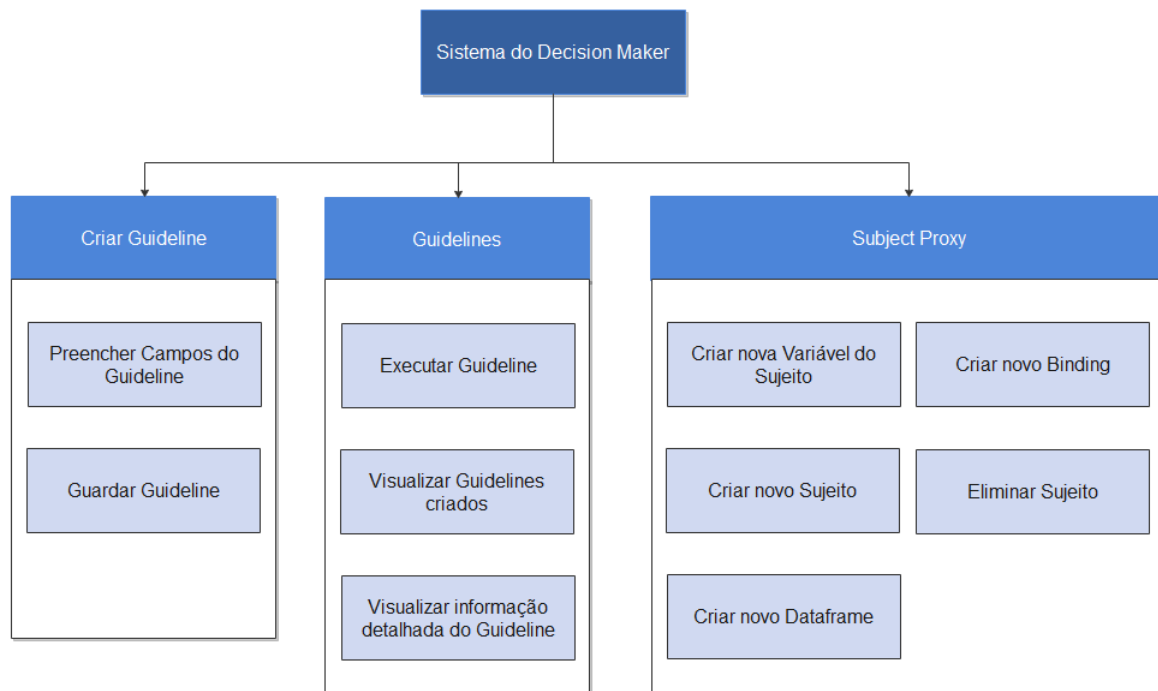


Figura 20: Diagrama da Arquitetura Funcional do Decision Maker

O diagrama acima apresentado mostra de forma clara todas as funcionalidades que o *website* oferece ao utilizador. Deste modo, identificou-se três funcionalidades principais:

- **Criar Guideline:** sendo a funcionalidade principal, o *website* disponibiliza ao utilizar a possibilidade de criar guidelines através do preenchimento do formulário apresentado e, posteriormente, armazená-los.
- **Guidelines:** esta funcionalidade permite ao utilizador para além de visualizar todos os *guidelines* criados, executar e/ou visualizar os detalhes referentes a um *guideline*.
- **Subject Proxy:** nesta área o utilizador pode gerenciar as funcionalidades referentes ao *Subject Proxy* como: criar e/ou remover sujeito, criar *data bindings*, *dataframes* e variáveis do sujeito

Uma vez explicitadas e devidamente explicadas as funcionalidades existentes da aplicação *web* em questão torna-se fucral entender a organização estrutural da mesma. Deste modo, recorreu-se ao auxílio do diagrama de arquitetura de seguida apresentado.

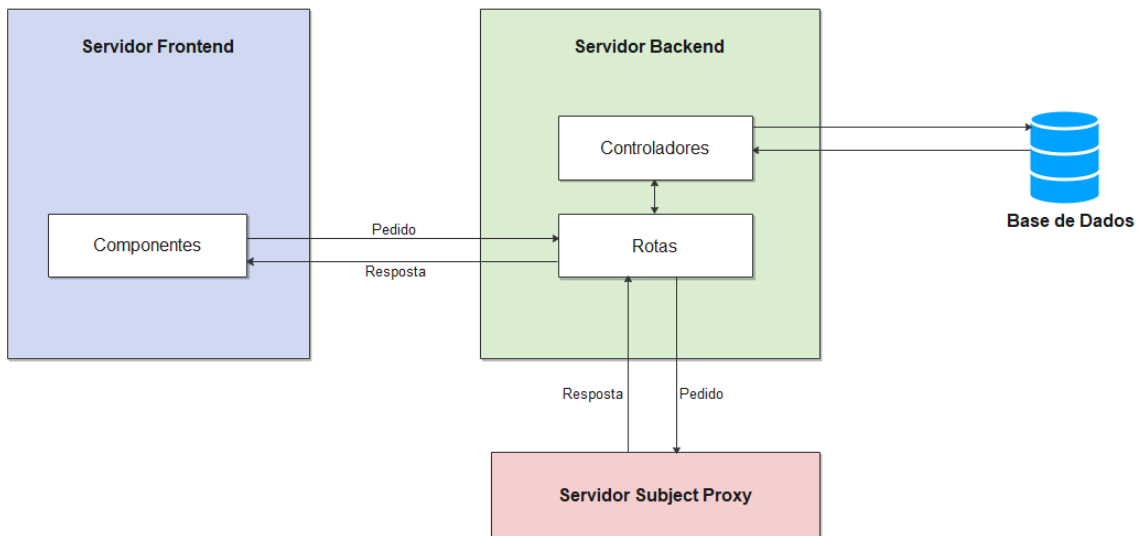


Figura 21: Diagrama de Arquitetura

Através do diagrama acima apresentado é fácil concluir que se seguiu a arquitetura Three-Tier [16]. Esta arquitetura divide essencialmente a aplicação em três "camadas"/ *tiers* lógicas que são fisicamente e computacionalmente independentes. Embora camadas não seja a tradução mais precisa para o termo tier, esta arquitetura divide-se nas seguintes:

- **Tier de Apresentação:** neste caso corresponde ao servidor front-end, sendo responsável por apresentar ao utilizador a interface na qual pode interagir e tirar vantagem das funcionalidades oferecidas pelo sistema.
- **Tier da Aplicação:** corresponde ao servidor back-end sendo o cerne do sistema. É responsável por processar a informação recebida pela *Tier de Apresentação* e por manipular e/ou adicionar dados referentes à base de dados.
- **Tier dos Dados:** corresponde à Base de Dados, sendo o local onde a informação é armazenada e, posteriormente gerida.

A escolha de seguir a arquitetura *Three-Tier* ao invés de outros tipos prendeu-se nas numerosas vantagens que esta apresenta como [16]:

- **Escalabilidade melhorada:** visto que cada "camada" é independente, é possível otimizar o processo de escalabilidade sem acarretar trabalho de adaptação desnecessário.
- **Maior confiabilidade:** caso uma camada por algum motivo fique comprometida, dificilmente afetará a disponibilidade ou desempenho das outras.

- **Maior segurança:** visto que a camada de apresentação e camada de dados não comunicam entre si é possível evitar possíveis ataques maliciosos por parte de terceiros.

Tendo em mente a explicação que foi anteriormente exposta, é necessário perceber ao promenor o funcionamento do sistema em si. Assim, graças ao diagrama anterior [22](#) temos as seguintes Tiers/"camadas":

- **Servidor front-end:** é responsável por renderizar a aplicação *web* e disponibilizar os meios que permitem ao utilizador usufruir de todas as funcionalidades oferecidas pelo mesmo. O servidor realiza pedidos ao back-end que tanto podem ser para obter dados como para armazená-los.
- **Servidor back-end:** trata-se da camada de negócio da aplicação, sendo portanto responsável por responder aos pedidos provenientes do front-end, processá-los e fornecer uma resposta adequada e útil. No que toca aos *guidelines*, implementa o motor de inferência que trata de processá-los e produzir os outputs conforme as regras e condições especificadas. Este quando necessita realiza pedidos ao Subject Proxy de modo a obter as variáveis do sujeito que precisa aquando a execução do guideline. O servidor do back-end comunica através de controladores com a Base de Dados de modo a obter dados/guidelines e armazená-los se requerido e através das rotas com o servidor front-end de modo a responder aos pedidos.
- **Base de Dados:** encarregue por logicamente armazenar os guidelines criados.
- **Servidor Subject Proxy:** este componente não foi implementado neste projecto, sendo apenas utilizado quando necessário. Ocasões em que este pode ser utilizado incluem obtenção de variáveis do sujeito, criação de sujeitos, *dataframes*, variáveis e *data bindings*.

### 4.3 Representação Computacional do Guideline:

Com base nos conceitos acima abordados e com a delineação dos requisitos acima apresentados, implementou-se a classe Guideline que pretende representar o objeto Guideline no contexto da aplicação.

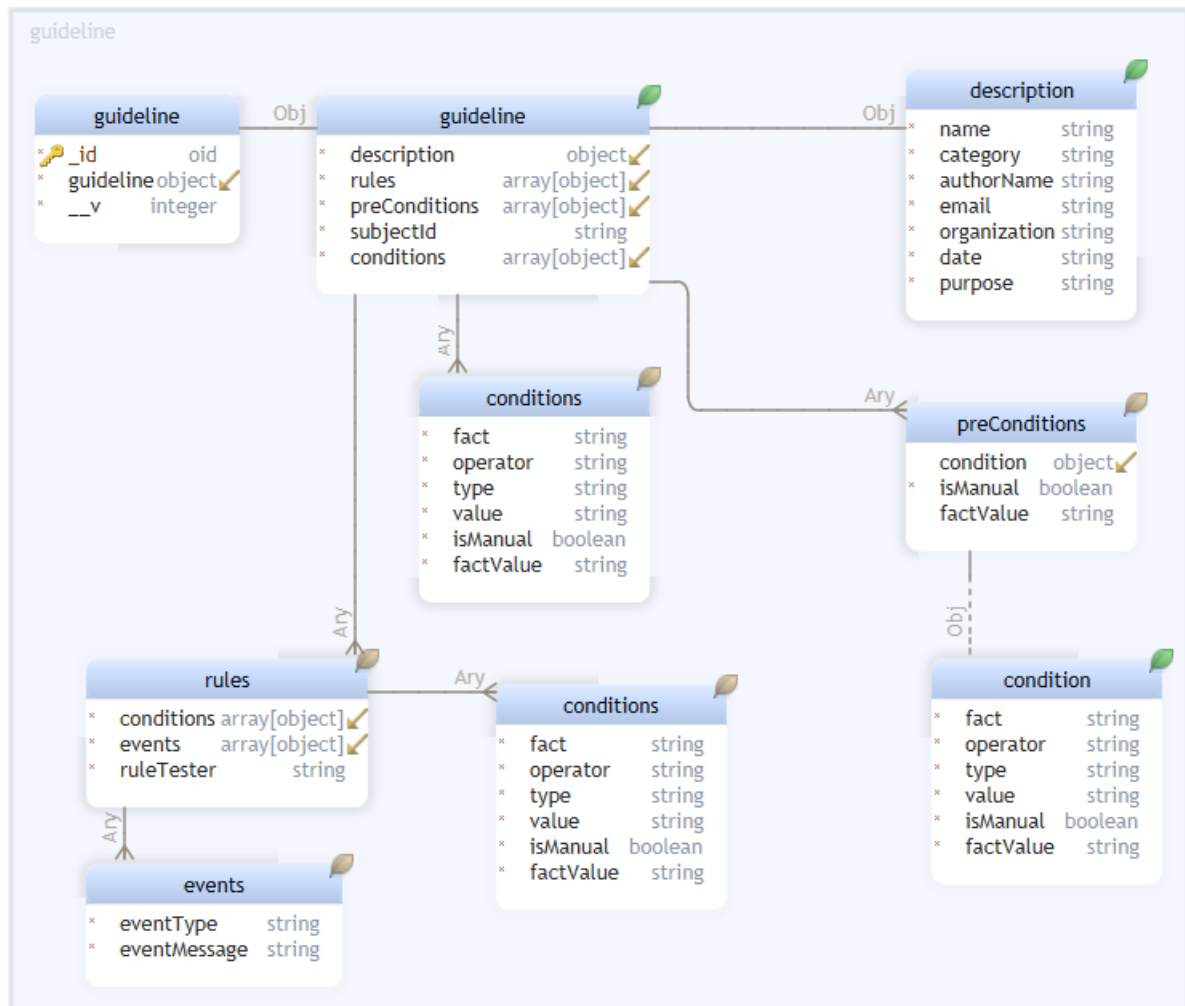


Figura 22: Esquema que pretende mostrar um exemplo de um documento armazenado na base de dados

A figura acima apresentada permite mostrar o que cada documento da base de dados possui. Assim, cada documento tem os seguintes campo:

- **\_id:** identifiador único gerado automaticamente aquando o armazenamento na base de dados
- **guideline:** objecto guideline, criado a partir da interface do utilizador.

Para simplificar o processo de envio de informação entre as diversas "camadas" do sistema, definiu-se uma classe Guideline que pretende representar computacionalmente todos os campos inseridos no mesmo. Desta forma, é importante salientar que tanto o servidor back-end, como o front-end e a base de dados têm o mesmo conhecimento de Guideline o que torna bastante simples a sua manipulação e posterior armazenamento. Salienta-se que a base de dados armazena no campo guideline, o objecto criado a partir da classe Guideline.

Este promenor permite facilitar bastante a criação dos guidelines, o seu posterior processamento no back-end e o seu armazenamento na base de dados, pois todas as entidades do sistema possuem o mesmo conhecimento do que representa o Guideline no universo da aplicação web implementada.

Assim, a classe Guideline apresenta os seguintes campos:

- **description:** armazena parâmetros descritivos acerca do guideline, como o nome do autor (**author Name**), o **email**, a organização (**organization**) à qual pertence, a data (**date**) associada ao momento de criação e ainda o propósito (**purpose**) visando esclarecer o objetivo último do guideline.
- **conditions:** array que armazena as condições criadas pelo utilizador. Para cada condição é atribuído os seguintes campos:
  - *fact*: nome da condição.
  - *type*: indica o tipo da condição, podendo assumir unicamente os seguintes valores: Number, String ou Array.
  - *operator*: consoante o tipo de condição podem-se definir diferentes operadores como Igual, Maior/Igual, Contido, Diferente, entre outros.
  - *isManual*: assume dois valores lógicos: verdadeiro ou falso. Se o valor for igual a falso, significa que a condição criada utiliza como valor a variável do sujeito inserida. Se o valor for verdadeiro, significa que o valor inserido será posteriormente solicitado ao utilizador, que o terá de inserir.
  - *value*: indica o valor da condição para o qual o valor do facto terá de ser testado
  - *factValue*: corresponde ao valor do facto que tanto pode ser inserido manualmente, como pode ser obtido através do Subject Proxy.
- **preConditions:** corresponde ao um array que armazena as pré-condições preenchidas pelo utilizador. Estas pré-condições podem ser criadas pelo utilizador, ou podem ser associadas a uma condição, previamente definida. Assim tem-se os seguintes campos:
  - *isManual*: pode assumir dois valores lógicos: verdadeiro ou falso. Caso seja verdadeiro indica que o utilizador posteriormente terá que indicar se a condição é verdadeira ou não. Caso seja falso, significa que o utilizador usou uma das condições previamente criadas.

- **condition**: este campo pode ser apresentado de duas formas diferentes, dependendo se a pré-condição utiliza ou não condições. Deste modo, caso utilize condições previamente criadas, este campo corresponderá à condição, contendo todos os campos explicados anteriormente. Caso não utilize condições, este campo indica apenas o nome da pré-condição criada.
- **factValue**: corresponde ao campo do valor da pré-condição. Este tanto pode ser obtido pela inserção manual por parte do utilizador, como pode ser obtido através das variáveis do Subject Proxy.
- **subjectId**: é o identificador do sujeito contido no serviço do SubjectProxy.
- **rules**: corresponde às regras criadas pelo utilizador. Cada regra, contém os seguintes campos:
  - **rulesTester**: que pode assumir o valor de "Tudo" ou "Qualquer um". No caso de ser "Tudo", as condições definidas serão avaliadas, utilizando o operador lógico de disjunção. No caso de estar definido "Qualquer um", significa que o operador lógico utilizado para avaliar as condições será o operador de conjunção.
  - **condition**: reflete o array de condições, definido pelo o utilizador. Cada índice do array contém um objecto do tipo *condition*, previamente abordado.
  - **events**: corresponde ao *array* de eventos associado a uma regra. Cada evento contém os seguintes campos:
    - \* **eventType**: diz respeito ao tipo de evento criado.
    - \* **eventMessage**: diz respeito à mensagem que deve ser devolvida, caso o conjunto de condições definido se avalia como verdadeiro.

## 4.4 Motor de Inferência

Como referido anteriormente, todo o sistema de criação de *guidelines* utiliza como representação computacional, a classe *Guideline*. Consequentemente, e a nível de back-end definiu-se um método que implementa o motor de inferência. Este motor de inferência permite validar as pré-condições e regras definidas, devolvendo o output para que possa ser apresentado ao utilizador na interface.

Assim, com o intuito de explicar o modo de funcionamento do motor que é responsável por avaliar o *guideline* elaborou-se o diagrama 24, que é no fundo uma adaptação do diagrama de data flow.

Como se pode constar na figura 24, o motor de inferência corresponde ao método **run** contido na classe do *Guideline*. O método **run** retorna uma promessa que segue as seguintes etapas:

- **Avaliação das Pré-condições:** nesta fase, o motor procura avaliar as pré-condições definidas através da invocação do método **evaluatePreConditions**. Resumidamente, este método percorre o array de pré-condições e para cada pré-condição avalia o valor do facto com o valor da pré-condição. No final é retornado um valor booleano. Se retornar verdadeiro significa que todas as pré-condições foram avaliadas como verdadeiras e pode-se, então prosseguir para a análise das regras. Se retornar falso, significa que alguma pré-condição falhou, senão foram todas a falhar e é retornado uma mensagem, parando assim a execução.

Na figura 23 pode-se observar o diagrama de data flow para o método **evaluatePreConditions** que foi abordado anteriormente.

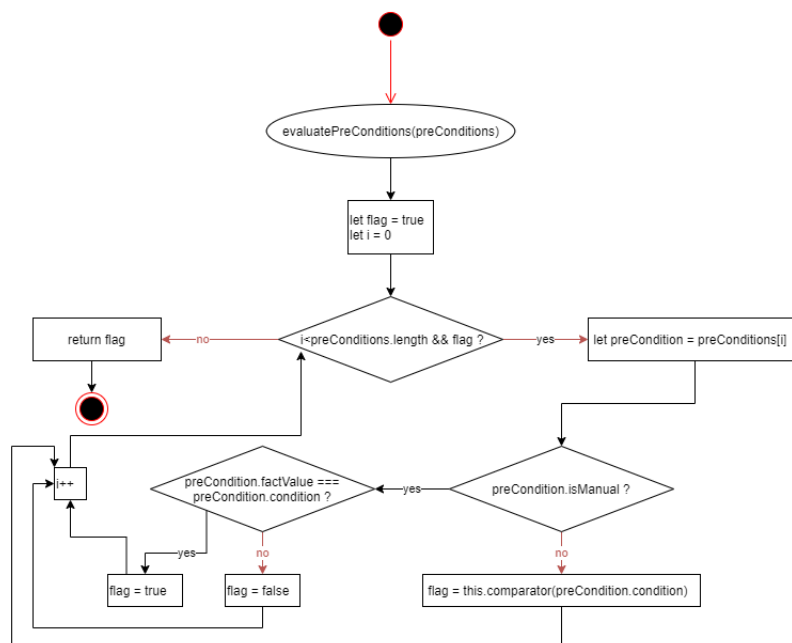


Figura 23: Adaptação do diagrama de dataflow para a função `evaluatePreConditions`

- **Avaliação das regras:** nesta etapa, o motor é capaz de testar as regras se e só se as pré-condições foram avaliadas como verdadeiras previamente. Deste modo, é percorrido o array de regras contido no *guideline* e para cada regra é percorrido o *array* das condições.

No *array* de condições, para cada condição invoca-se o método **comparator** que é responsável por avaliar a condição. No fundo testa o valor do facto com o valor definido na condição, utilizando o operador definido na mesma e retorna um valor logico verdadeiro ou falso. Este valor lógico é armazenado no *array* de *flags* previamente inicializado a verdadeiro e cujo tamanho corresponde ao tamanho do *array* de condições atual. A cada iteração do array é ainda avaliado o array das flags. Esta avaliação é feita consoante o operador lógico definido no campo **ruleTester** da regra atual. Assim, se o *ruleTester* for igual a Tudo, as *flags* serão avaliada utilizando o operador lógico de disjunção. Caso contrário as *flags* serão testadas utilizando o operador lógico de conjunção. Quando a iteração do array das condições acabar, verifica-se se a regra foi avaliada como verdadeira ou falsa. No caso de ser verdadeiro (flag == true), percorre-se o array de eventos da regra e adiciona-se cada evento ao array de eventos auxiliar que será retornado no final da execução do método *run*. No final da iteração das regras presentes no *array*, verifica-se se o *array* auxiliar de eventos contém ou não elementos inseridos neste. Se conter devolve o *array* com a invocação do *resolve* da Promise. Caso contrário, invoca o *reject* com uma mensagem de erro que será apresentada ao utilizador a nível de frontend.





a aplicação web desenvolvida. Adicionalmente e a par com cada imagem será exposto uma breve explicação acerca do funcionamento da mesma e ainda uma definição para os conceitos apresentados.

### 4.5.1 Página Inicial

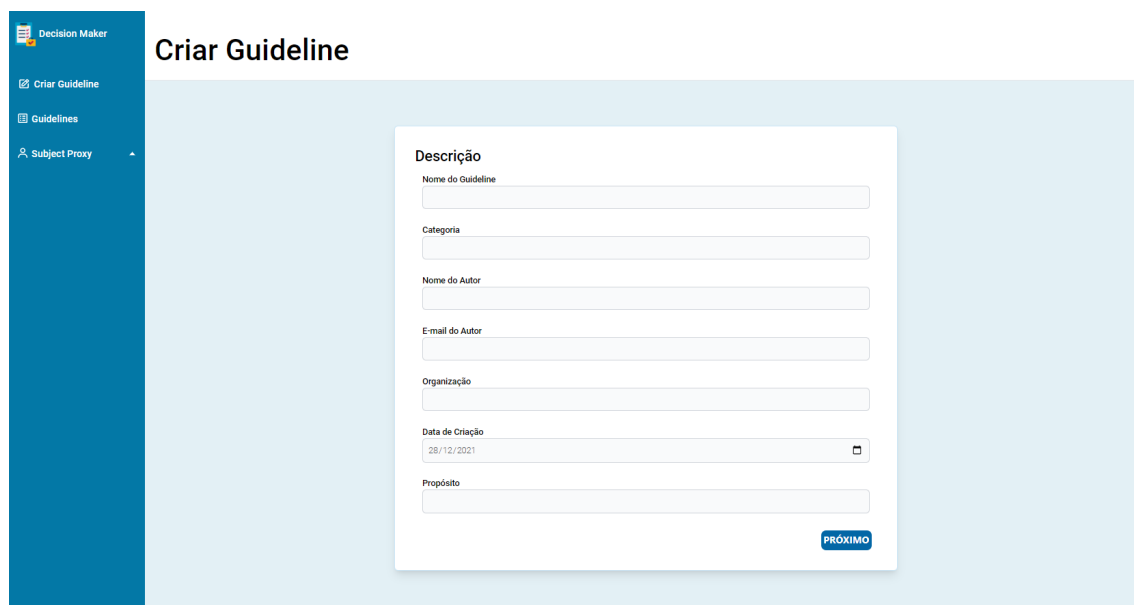


Figura 25: Página Inicial do Guideline Creator

Quando o utilizador inicia a sua experiência na aplicação é confrontado com a página inicial que se pode observar na figura 25.

Como se consta, a primeira página diz respeito à secção de criação de *guidelines*. O utilizador pode alternar de página consoante a sua vontade, ou seja, se pretender pode continuar nesta secção e assim criar o *guideline* pretendido, ou pode escolher outras funcionalidades que o sistema oferece e que são enumeradas na barra de navegação fixada do lado esquerdo do ecrã.

### 4.5.2 Barra de Navegação

Para facilitar a navegação no *website* implementou-se uma barra de navegação que disponibiliza as seguintes secções:

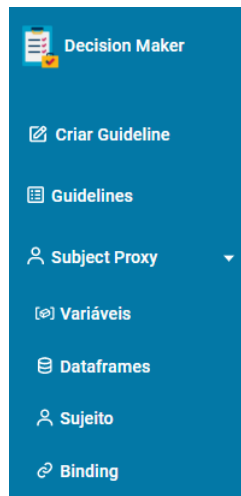


Figura 26: Barra de navegação

Como se pode observar, o utilizador pode navegar no site e escolher as seguintes páginas:

- **Criar Guideline:** nesta secção pode criar os *guidelines* que desejar.
- **Guidelines:** aqui pode visualizar os *guidelines* criados, correr um determinado *guideline* e ainda visualizar as informações detelhadas relativas a um *guideline*.
- **Subject Proxy:** nesta secção o utilizador pode criar novas variáveis, criar e/ou remover sujeitos e ainda criar novos *dataframes*. Portanto, esta secção é a que permite manipular dados do serviço do Subject Proxy.

### 4.5.3 Criação Guideline

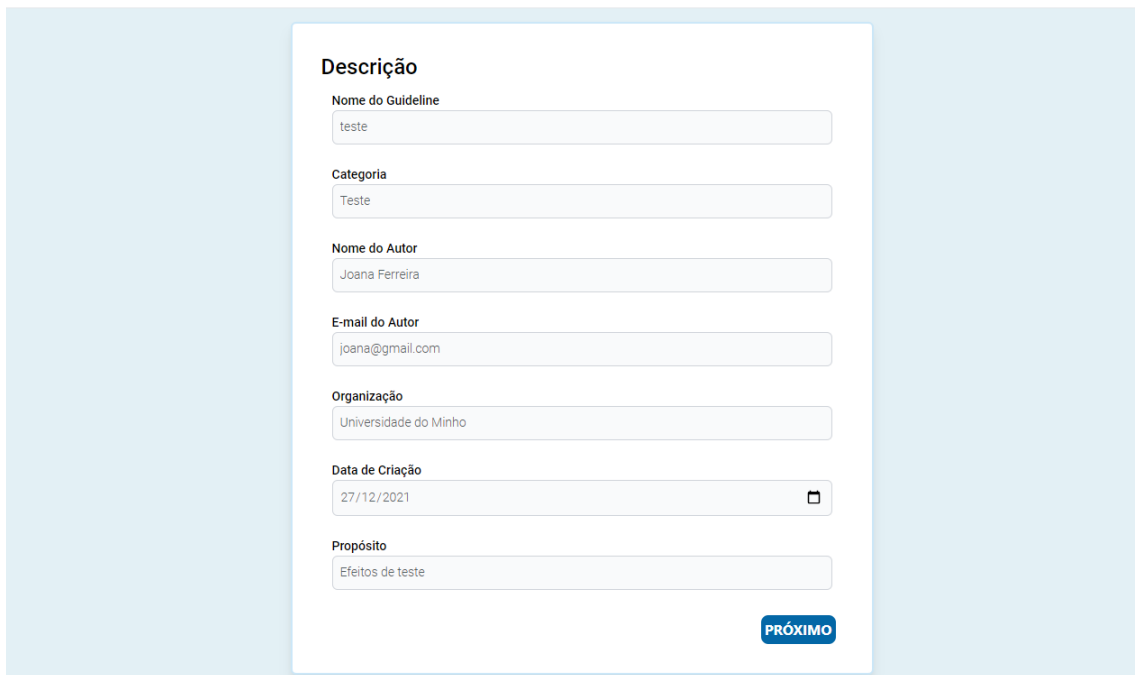
Antes de proceder à explicação desta funcionalidade é necessário ter em mente uma definição clara do que é um *guideline*.

Segundo as especificações do openEHR, o *guideline* pode ser interpretado como um conjunto de declarações desenvolvidas sistematicamente, visando ajudar os profissionais e pacientes a tomar decisões informadas para determinadas circunstâncias clínicas [58]. Assim, os *guidelines* podem ser categorizados nos seguintes:

- **terapêuticos:** descrevem procedimentos como por exemplo: administração de medicamentos, dosagens, gestão de uma condição específica como diabetes, entre outros.
- **diagnóstico:** descrevem diagnósticos como, determinação do tipo de diabetes, entre outros,
- **baseados em pontuação:** descrevem *guidelines* cujo objetivo é estratificar riscos associados a um determinado tipo de intervenção.

Uma vez esclarecido o conceito de *guideline* pode-se, assim, prosseguir para a explicação do processo de criação de *guidelines*. De modo a simplificar este processo achou-se vantajoso dividir em diferentes passos, diminuindo assim a quantidade de informação a que o utilizador é confrontando.

## Criar Guideline



The image shows a web form titled "Descrição" for creating a guideline. It includes the following fields and values:

- Nome do Guideline:** teste
- Categoria:** Teste
- Nome do Autor:** Joana Ferreira
- E-mail do Autor:** joana@gmail.com
- Organização:** Universidade do Minho
- Data de Criação:** 27/12/2021
- Propósito:** Efeitos de teste

A blue button labeled "PRÓXIMO" is positioned at the bottom right of the form.

Figura 27: Criação do Guideline, secção de Descrição

Como se pode observar na figura 27, o primeiro passo de criação corresponde à **Descrição**. Aqui, o utilizador preenche parâmetros como o nome do *guideline*, o nome e e-mail do autor, a instituição à qual está associado, a data de criação e o propósito.

Uma vez preenchidos estes campos, o utilizador pode passar ao próximo passo clicando no botão próximo.

## Criar Guideline

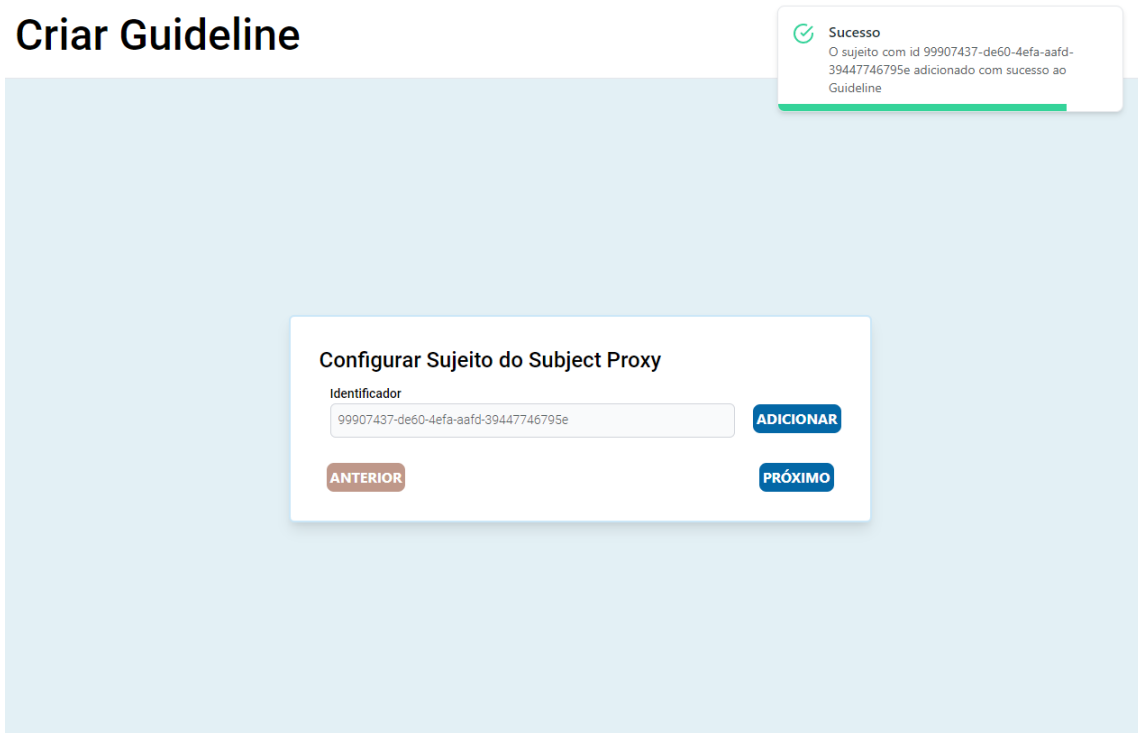


Figura 28: Criação do Guideline, secção de Configuração do Sujeito do Subject Proxy

O próximo passo corresponde à Configuração do Sujeito do Subject Proxy e pode ser visualizada na figura 28. Antes de prosseguir, é necessário esclarecer o conceito de Sujeito no contexto do problema.

No contexto da problemática em questão, um sujeito refere-se à entidade alvo de cuidados, ou seja, o paciente. Neste caso, o sujeito encontra-se identificado no serviço do Subject Proxy e é através deste serviço que é possível obter as variáveis que contêm os dados provenientes de fontes heterogêneas. Entende-se como variáveis do sujeito valores que são obtidos através do Serviço do Subject Proxy. Estas variáveis dizem respeito ao sujeito alvo de cuidados, neste caso o paciente, podendo ser, por exemplo, a idade, data de nascimento, valores de colesterol, entre outros.

Portanto, voltando à explicação da Configuração do Subject Proxy 28, o utilizador pode associar um sujeito contido no serviço do Subject Proxy ao *guideline* que está a criar. Para tal necessita de inserir um identificador válido deste. Caso não insira um identificador válido uma mensagem de erro aparecerá no ecrã a informar o utilizador.

No caso da figura 28 o identificador inserido encontra-se no Serviço do Subject Proxy, mas caso não esteja inserido a seguinte mensagem será mostrada.

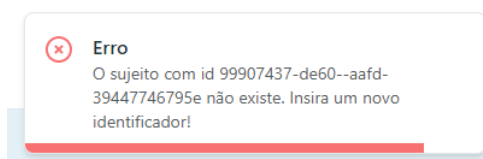
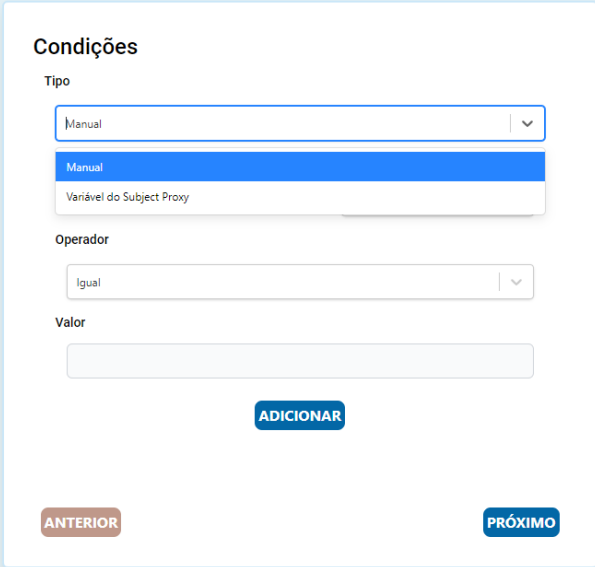


Figura 29: Demonstração da mensagem de erro mostrada caso o identificador inserido seja inválido.

Uma vez definido o sujeito, o utilizador pode seguir para o próximo passo correspondendo à criação das condições.

## Criar Guideline



The image shows a web form titled "Condições" (Conditions) for creating a guideline. It features three main sections: "Tipo" (Type) with a dropdown menu showing "Manual" selected and a list of options including "Manual" and "Variável do Subject Proxy"; "Operador" (Operator) with a dropdown menu showing "Igual" (Equal) selected; and "Valor" (Value) with an empty text input field. At the bottom of the form, there are three buttons: "ADICIONAR" (Add) in blue, "ANTERIOR" (Previous) in brown, and "PRÓXIMO" (Next) in blue.

Figura 30: Criação do Guideline, secção das Condições

Ao criar as condições, o utilizador tem de escolher primeiro o tipo de condição que pretende criar:

- **Manual:** significa que o utilizador deseja que o facto seja posteriormente inserido manualmente através do teclado.
- **Variável do Subject Proxy:** significa que o utilizador deseja que o valor do facto seja obtido através do serviço do Subject Proxy.

Caso o tipo de condição escolhida seja **Manual** a seguinte janela é despoletada.

**Condições**

Tipo  
Manual

Nome  Tipo  
Texto

Operador  
Igual

Valor

**ADICIONAR**

Lista de Condições

Facto	Operador	Valor
condiçao	==	sim

**ANTERIOR** **PRÓXIMO**

Figura 31: Demonstração da criação de uma condição do tipo Manual

Aqui o utilizador pode definir o nome da condição, o tipo que pode ser : Texto, Numero ou Lista (32), o operador (pode ser observado nas figuras 33, 34 e 35) que varia consoante o tipo definido e ainda o valor da condição para o qual o facto será avaliado. No final de preencher estes campos o utilizador pressiona o botão de adicionar e é apresentada uma lista das condições que criou.

Tipo  
Texto

Operador  
Igual

Valor

Figura 32: Demonstração da lista de tipos existente

Tipo  
Texto

Operador  
Igual

Valor  
Igual

Figura 33: Demonstração da lista de operadores definida para o tipo Texto

The image shows a user interface for configuring a decision rule. It consists of three main components: a 'Tipo' dropdown menu with 'Número' selected, an 'Operador' dropdown menu with 'Igual' selected, and a 'Variável do Subject Proxy' dropdown menu. The 'Variável do Subject Proxy' dropdown is open, displaying a list of operators: 'Igual', 'Diferente', 'Maior', 'Menor', 'Maior ou igual', and 'Menor ou igual'. The 'Igual' option is highlighted in blue.

Figura 34: Demonstração da lista de operadores definida para o tipo Número

The image shows a user interface for configuring a decision rule. It consists of three main components: a 'Tipo' dropdown menu with 'Lista' selected, an 'Operador' dropdown menu with 'Contido' selected, and a 'Variável do Subject Proxy' dropdown menu. The 'Variável do Subject Proxy' dropdown is open, displaying a list of operators: 'Contido' and 'Não Contido'. The 'Contido' option is highlighted in blue.

Figura 35: Demonstração da lista de operadores definida para o tipo Lista

Caso o tipo de condição escolhida seja **Variável do Subject Proxy** a seguinte janela é despoletada:



**Condições**

Tipo

Facto

Tipo

Operador

Valor

**ADICIONAR**

Lista de Condições

Facto	Operador	Valor
condição	===	sim

**ANTERIOR** **PRÓXIMO**

Figura 36: Demonstração da criação de uma condição do tipo Variável do Subject Proxy

Nesta secção, o utilizador pode preencher os mesmo campos explicados anteriormente com a exceção do nome da condição que ficou definida como sendo o nome da variável escolhida para condição que está a criar. Salienta-se que é fornecida uma lista ao utilizador onde aparecem todas as variáveis existentes do sujeito e onde o utilizador pode escolher uma destas [37](#).

Facto

- id\_ped
- numpedido
- teamuid
- team
- sequencial
- processo
- nome
- sexo
- datanascimento

Figura 37: Demonstração da lista de variáveis do Sujeito do Subject Proxy apresentadas ao utilizador

O utilizador pode, assim, adicionar o número de condições que bem entender como pode eliminar as que desejar. Quando este finalizar a criação de condições, pode então prosseguir para o próximo passo relativo às **Pré-Condições** ([38](#)).

As pré-condições descrevem condições booleanas que têm de se avaliar como verdadeiras de modo a que o *guideline* possa ser executado [58]. São, portanto, uma espécie de filtro às regras definidas no *guideline*.

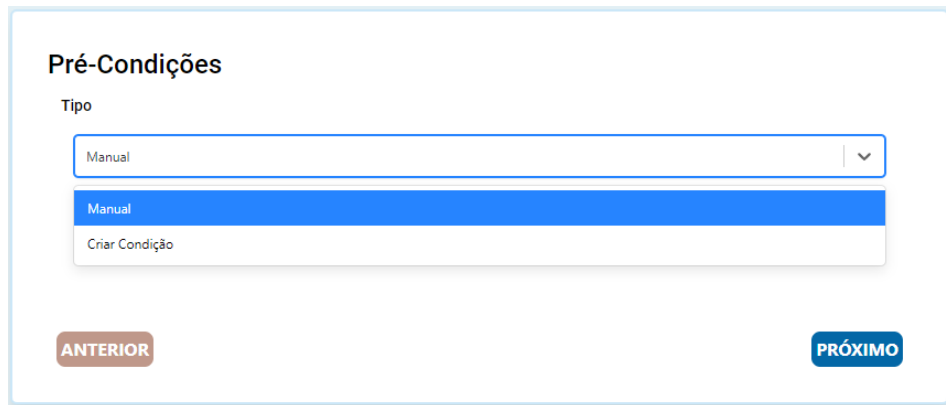
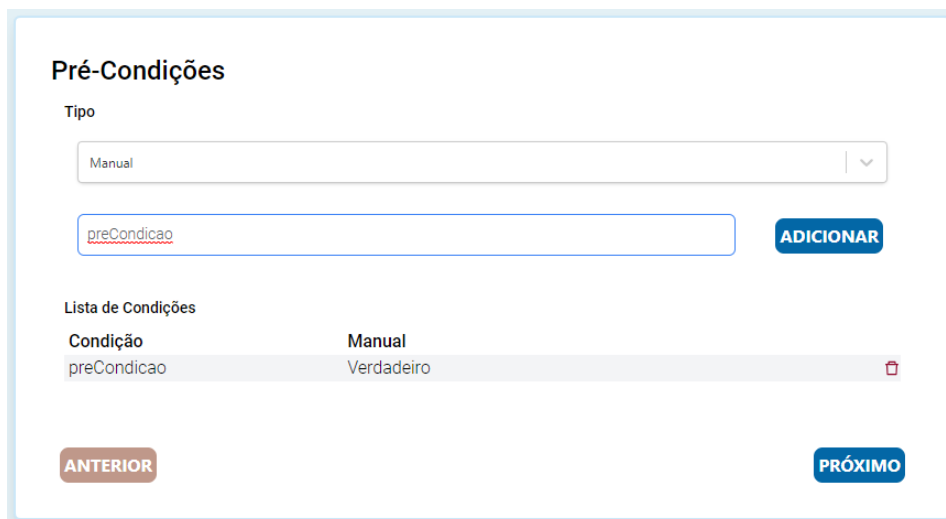


Figura 38: Criação do Guideline, secção das Pré-condições

Aqui o utilizador pode criar dois tipos de condição:

- **Manual:** significa que o utilizador pretende que posteriormente o valor seja definido através do teclado,
- **Criar Condição:** significa que o utilizador pretende associar à pré-condição uma condição já existente.

No caso do tipo escolhido ser **Manual** a seguinte janela é ativada:



Lista de Condições	
Condição	Manual
preCondicao	Verdadeiro

Figura 39: Demonstração da criação de um pré-condição do tipo Manual

Como se pode observar na figura 39, basta inserir na caixa de texto o nome da condição e clicar no botão de adicionar para adicionar à lista de condições existentes.

No caso do tipo escolhido ser **Criar Condição** a seguinte janela é ativada:

**Pré-Condições**

Tipo

Criar Condição

condicao

ADICIONAR

condicao

new\_mwpid

Condição	manual
preCondicao	Verdadeiro

ANTERIOR

PRÓXIMO

Figura 40: Demonstração da criação de um pré-condição do tipo Criar Condição

Aqui, o utilizador apenas tem de seleccionar uma das condições presentes na lista e adicionar à lista de pré-condições existente. Como se consta, em qualquer dos casos é possível eliminar condições já criadas.

Quando o utilizador não precisar de criar mais pré-condições pode passar para o próximo passo que corresponde à criação das **Regras**.

As regras geram valores que não são de carácter booleano. Estas utilizam as condições e quando são avaliadas correctamente produzem o output previamente delineado.

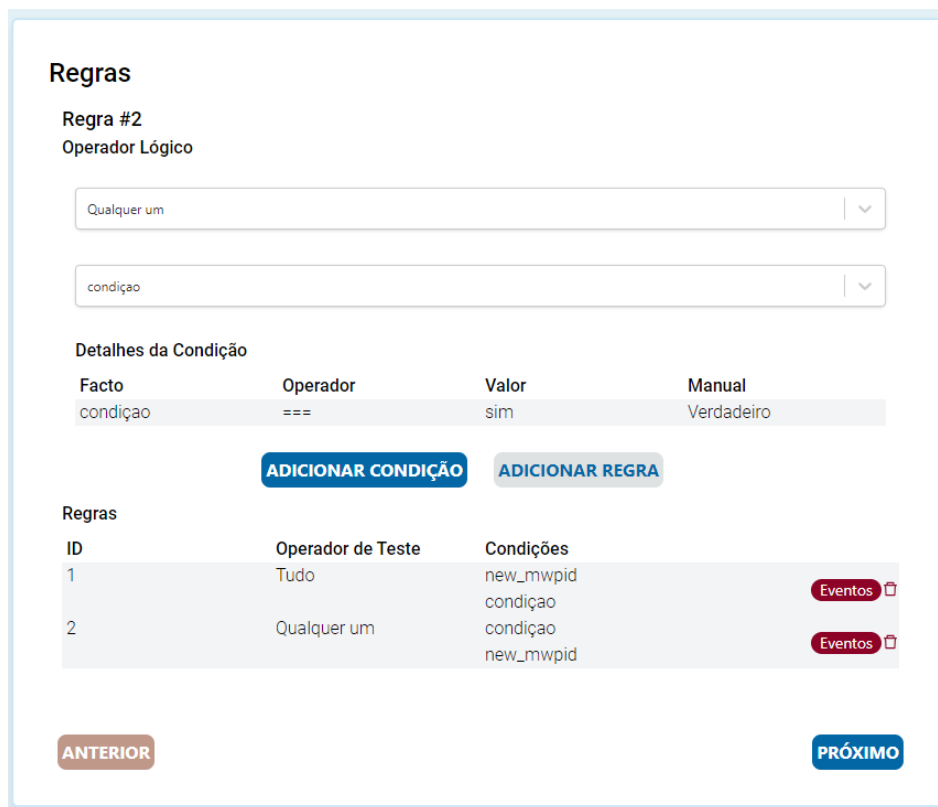


Figura 41: Criação do Guideline, secção das Regras

Nesta fase, o utilizador define as regras através das condições criadas. Assim, pode adicionar as condições que desejar à regra atual a partir das lista de condições apresentadas. Para além disso tem de definir o operador lógico para avaliar as condições. Este pode ser Tudo, que corresponde à disjunção das condições, ou pode ser Qualquer um que corresponde ao operador lógico de conjunção. Note que para cada condição selecionada é apresentado uma breve descrição da mesma (Detalhes da Condição)

Ainda, é possível adicionar um conjunto de eventos para cada regra. Na figura 42 é possível observar o modo de criação de eventos:

**Eventos**

Tipo de Evento

aviso

Mensagem

uma mensagem

**ADICIONAR**

Lista de Eventos

Tipo	Mensagem	
aviso	uma mensagem	

**FECHAR** **GUARDAR**

Figura 42: Demonstração da criação de eventos associada a uma regra específica

Como se consta através da análise da figura, o utilizador apenas tem que preencher os campos de texto relativos ao tipo de evento e mensagem e quando finalizar basta clicar no botão adicionar para adicionar o novo evento criado. Note que é apresentado ao utilizador a lista de eventos, podendo eliminar as que desejar.

Uma vez criadas as regras o utilizador pode prosseguir à próxima fase em que lhe é apresentado um **Resumo** do guideline criado.

### Resumo

#### Descrição

Nome do Guideline: teste  
 Categoria: Teste  
 Nome do Autor: Joana Ferreira  
 E-mail do Autor: joana@gmail.com  
 Organização: Universidade do Minho  
 Data de Criação: 2021-12-27  
 Propósito: Efeitos de Teste

#### Sujeito do Subject Proxy

Identificador do Sujeito: 99907437-de60-4efa-aafd-39447746795e

#### Pré-Condições

preCondicao  
 new\_mwpid

#### Condições

Facto	Operador	Valor
condicao	===	sim
new_mwpid	===	d097a4a1-5869-4acf-ab3f-935a78850f...

#### Regras

ID	Operador de Teste	Condições	Eventos	Mensagem
1	Tudo	condicao new_mwpid	Tipo aviso	Mensagem condições verificadas sucesso
2	Qualquer um	condicao new_mwpid	Tipo aviso	Mensagem condições verificadas

ANTERIOR

GUARDAR

Figura 43: Criação do Guideline, secção do Resumo

Nesta etapa, é apresentado ao utilizador um resumo do guideline criado (43) onde este pode observar todas as etapas criadas previamente. Uma vez revisto, o utilizador decide se prossegue com o armazenamento do *guideline* ou se pretende ir para trás e editar os campos que achar necessário.

Caso clique no botão de guardar o *guideline* é enviado para o back-end, que trata de armazená-lo na base de dados.

### 4.5.4 Guidelines

Ao clicar na secção de Guidelines na barra de navegação, é apresentado uma lista dos guidelines que foram criados previamente e que podem ser observados na seguinte figura:

## Guidelines

The image displays three guideline cards arranged horizontally. Each card contains the following information:

- Guideline 1:** ID: 61c8b6e7585c9f060c7859a0, Nome: name, Categoria: categoria, Nome do Autor: Sarah Silva, E-mail do Autor: a76867@alunos.uminho.pt, Organização: Universidade do Minho, Data de Criação: 2021-12-26, Propósito: testar binding.
- Guideline 2:** ID: 61c999d840265e19b8fe9227, Nome: Nome do Guideline, Categoria: Categoria, Nome do Autor: Guilherme Pereira Martins, E-mail do Autor: gui@hotmail.com, Organização: Universidade do Minho, Data de Criação: 2021-12-27, Propósito: testar binding.
- Guideline 3:** ID: 61c98eca40265e19b8fe9228, Nome: nome, Categoria: categoria, Nome do Autor: Sarah Silva, E-mail do Autor: a76867@alunos.uminho.pt, Organização: Universidade do Minho, Data de Criação: 2021-12-27, Propósito: um teste.

At the bottom of each card, there are two buttons: a yellow button with a magnifying glass icon labeled 'INFORMAÇÃO' and a red button labeled 'EXECUTAR'.

Figura 44: Guidelines

Nesta secção o utilizador pode consultar informações detelhadas referentes a um *guideline* ou executar

No caso de desejar consultar o *guideline*, ao clicar no botão de informação é despoletada a seguinte janela (45) que expõe todos os dados relativos ao *guideline*

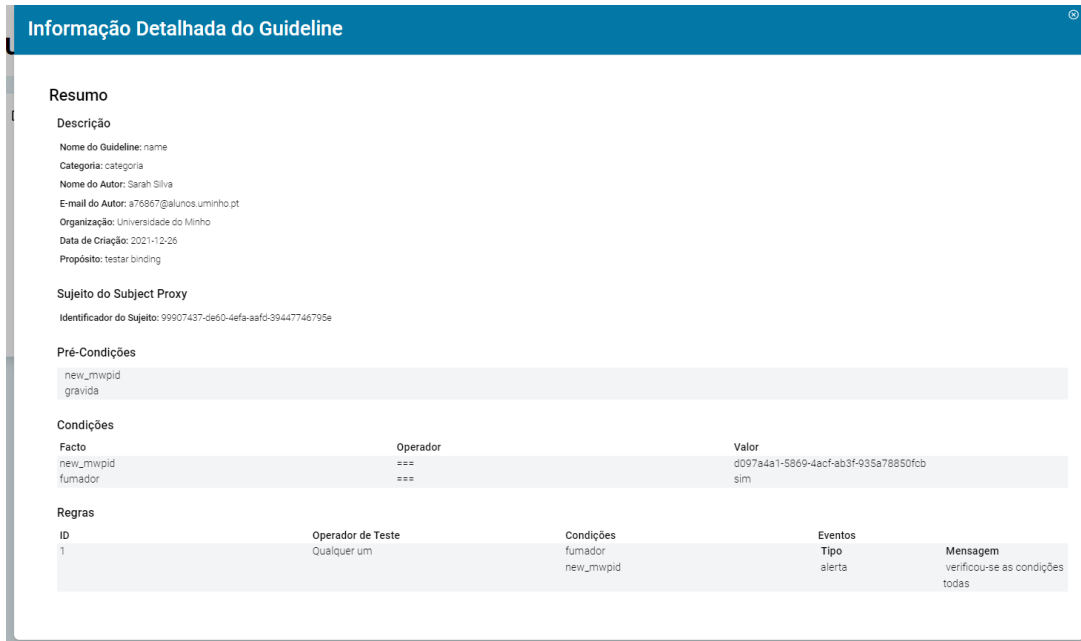


Figura 45: Guidelines, secção da Informação detalhada

No caso de desejar executar o guideline, é assim iniciado o processo de execução exposto na seguinte janela inicial:

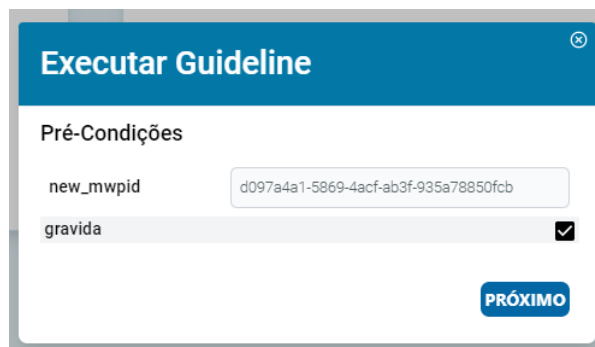
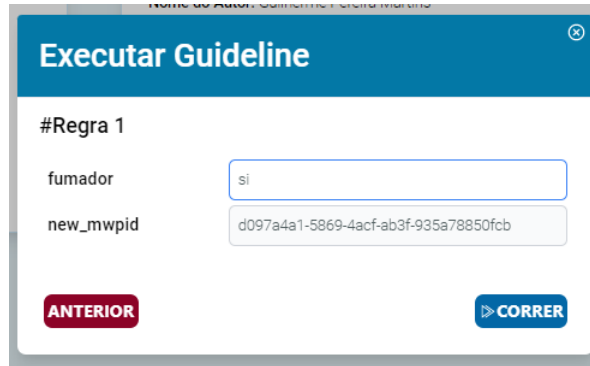


Figura 46: Guidelines, secção da Execução



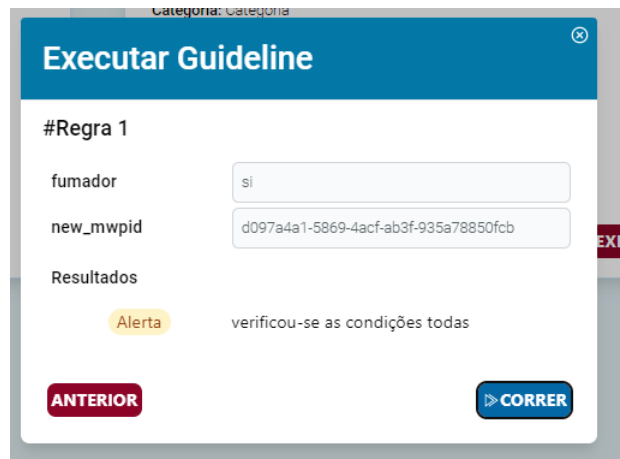
Sendo assim, a primeira etapa relativa ao processo de execução corresponde à validação das pré-condições. Aqui, o utilizador deve gerir as pré-condições e indicar se estas se verificam ou não. Uma vez configurado esta etapa, o passo seguinte corresponde à validação das regras e pode ser observado na seguinte figura:



The screenshot shows a web interface titled "Executar Guideline". Under the heading "#Regra 1", there are two input fields: "fumador" with the value "si" and "new\_mwpid" with the value "d097a4a1-5869-4acf-ab3f-935a78850fcb". At the bottom, there are two buttons: "ANTERIOR" (red) and "CORRER" (blue with a right-pointing arrow).

Figura 47: Guidelines, secção da Execução

Nesta secção, o utilizador preenche os campos necessários, sendo que alguns já se encontram preenchidos pois são valores obtidos através do Subject Proxy. Quando se encontrar pronto, basta clicar no botão correr e será mostrado o resultado final da avaliação do *guideline*.



The screenshot shows the same "Executar Guideline" interface as Figure 47, but with an additional section. Below the input fields, there is a "Resultados" section containing a yellow "Alerta" icon and the text "verificou-se as condições todas". The "ANTERIOR" and "CORRER" buttons are still present at the bottom.

Figura 48: Guidelines, demonstração de um exemplo de output gerado

É importante salientar que este *output* varia conforme os eventos definidos para as regras e também varia com a validação das regras e das pré-condições. Por exemplo, caso as pré-condições não se verificarem a seguinte mensagem será gerada.

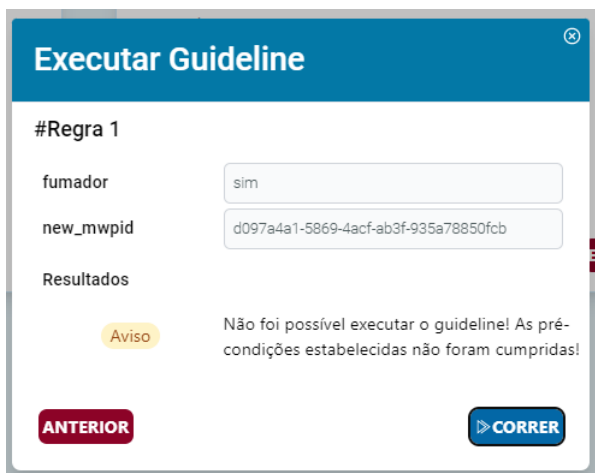


Figura 49: Guidelines, demonstração de um exemplo de output gerado, caso as pré-condições não sejam cumpridas

Caso sejam as regras a falhar, o *output* gerado pode ser observado na seguinte figura

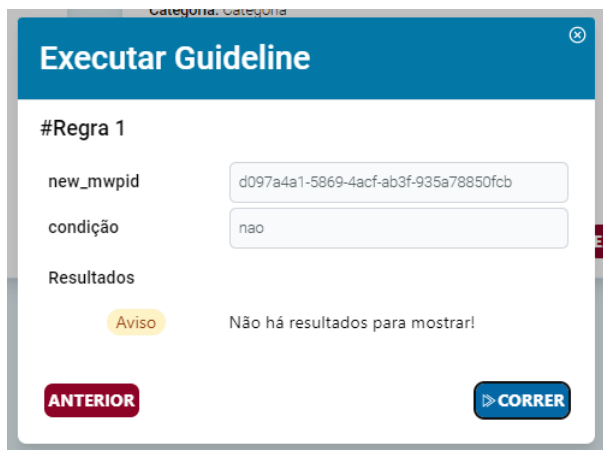


Figura 50: Guidelines, demonstração de um exemplo de output gerado, caso as regras não sejam cumpridas

### 4.5.5 Subject Proxy

No separador do Subject Proxy são apresentadas diversas funcionalidades.

Assim, em jeito de contextualização, o serviço do Subject Proxy possibilita que as variáveis relativas a um sujeito, variáveis estas que representam/caracterizam o estado real do paciente, possam ser monitorizadas ao longo do tempo [58]. Este serviço é de extrema relevância pois possibilita a obtenção

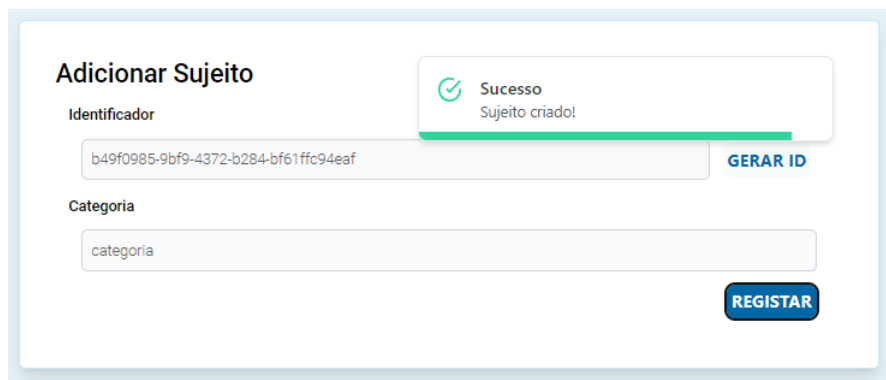
de dados fidedignos e atualizados, de forma padronizada, provenientes de diversos sistemas, como por exemplo, clínicas de análises laboratoriais, hospitais, entre outros. Adicionalmente, na eventualidade de a variável não existir, o serviço de Subject Proxy permite a adição de novas variáveis ao sujeito alvo. Este serviço também oferece abstração para qualquer variável em relação ao padrão que utiliza, bem como em relação ao modelo representacional e ainda em relação à linguagem de consulta ou à [API](#) dos dados fonte.

Resumidamente, o Serviço do Subject Proxy oferece as seguintes funcionalidades:

- Registrar o sujeito;
- Adicionar variável de sujeito
- Registrar o data binding: adiciona uma ligação para um ambiente, o qual geralmente é um contexto de computação contendo vários sistemas backend a partir dos quais as variáveis do sujeito podem ser preenchidas.
- Adicionar um binding data frame: adiciona um data retrieval frame a um ambiente de ligação - isto fornece um método de acesso aos dados para o ambiente alvo.

Por sua vez, um binding é compreendido como um conjunto de métodos de recuperação (por exemplo, invocações de api, consultas), sendo cada binding definido pelo **data frame**, para um ambiente de execução particular e independente de qualquer sujeito em particular [45].

#### 4.5.5.1 Sujeito



Adicionar Sujeito

Identificador

b49f0985-9bf9-4372-b284-bf61ffc94eaf **GERAR ID**

Categoria

categoria **REGISTAR**

Sucesso  
Sujeito criado!

Figura 51: Subject Proxy, Adicionar novo Sujeito

Como se pode visualizar na figura 51, é possível criar um novo sujeito e adicioná-lo ao serviço do Subject Proxy. Para tal, basta clicar no botão gerar id, para gerar automaticamente um identificador válido e preencher o campo de texto relativo à categoria. Uma vez executados estes passos resta apenas clicar no botão de registrar e uma mensagem será apresentada ao utilizador de modo a informar se a operação foi bem sucedida ou não. Neste caso a operação foi concluída com sucesso.

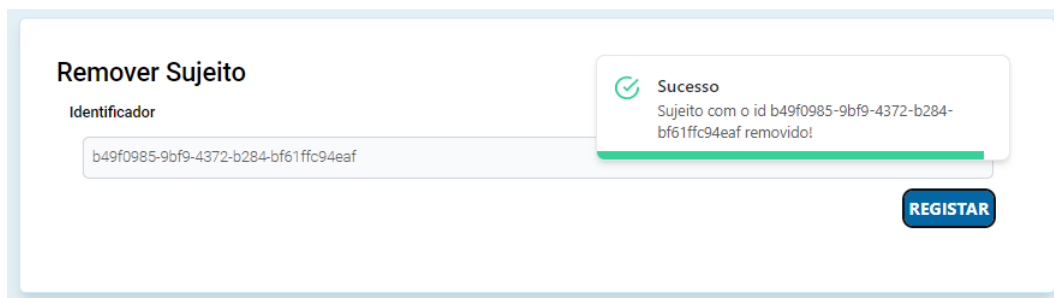


Figura 52: Subject Proxy, Remover Sujeito

A figura 52 corresponde a outra funcionalidade que o sistema apresenta relativa a esta parte do Sujeito que corresponde à remoção. Assim, para o utilizador remover o sujeito têm de apenas indicar o identificador do mesmo e registar. Uma mensagem será gerada a informar o utilizador se a operação foi ou não bem sucedida.

#### 4.5.5.2 Variáveis do Sujeito

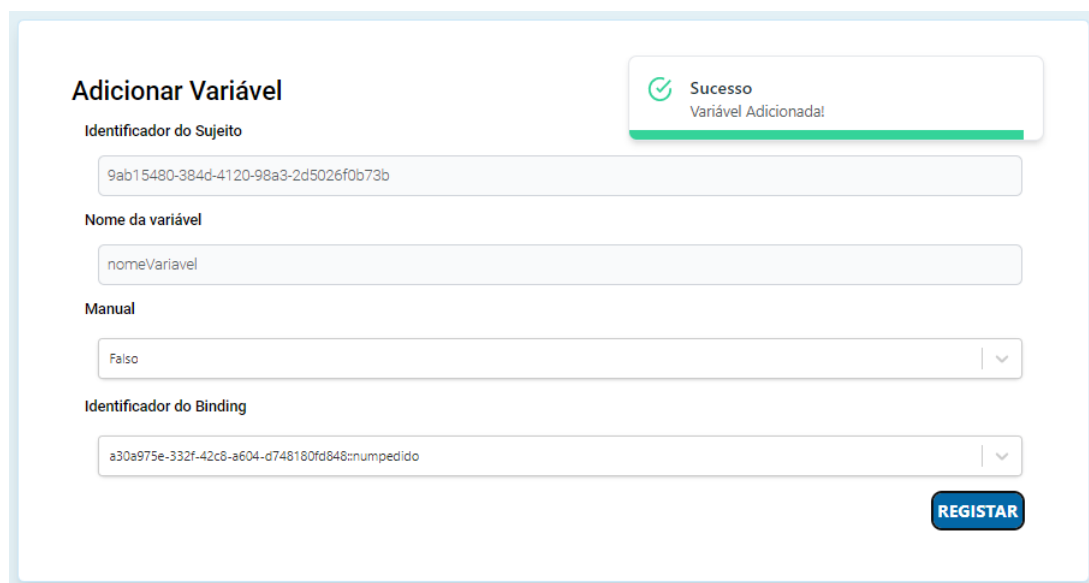


Figura 53: Subject Proxy, Adicionar nova variável ao Sujeito

É possível adicionar novas variáveis ao sujeito através da figura 53. Aqui é apenas necessário preencher os campos requeridos, como o identificador do sujeito e o nome da variável, selecionar se é manual ou não e ainda selecionar o identificador do binding, sendo que é facultado ao utilizador uma lista dos identificadores existentes. Para enviar para o serviço, basta apenas clicar no registar e uma mensagem informativa será apresentada, indicado se a operação foi bem sucedida ou não.

**Adicionar Dataframe**

Tipo

Nome

URL

**Adicionar Parâmetros**

Nome  Conteúdo

**Adicionar Parâmetros de Ligação**

Nome  Valor

Figura 54: Subject Proxy, Adicionar novo dataframe

### 4.5.5.3 Dataframes

Nesta secção é possível criar os dataframes que o utilizador bem entender, sendo que há dois tipos de dataframes que podem ser criados: do tipo Query ou do tipo [API](#).

No caso de se escolher o [API](#), os seguintes campos são mostrados ao utilizador

**Adicionar Dataframe**

Tipo:

Nome:

URL:

**Adicionar Parâmetros**

Nome:  Conteúdo:  **ADICIONAR**

Nome	Conteúdo
name	content
name2	content2

**Adicionar Parâmetros de Ligação**

Nome:  Valor:  **ADICIONAR**

Nome	Valor
name	value

**REGISTAR**

Figura 55: Subject Proxy, Adicionar novo dataframe do tipo API

Para criar um dataframe do tipo [API \(55\)](#), o user deve indicar o nome e o url, adicionar uma lista de parâmetros e de parâmetros de ligação. Uma vez preenchidos os respectivos campos resta apenas clicar no registrar para enviar o novo dataframe para o Serviço do Subject Proxy. No final é apresentado uma mensagem que informa se o operação foi ou não bem sucedida.

**Adicionar Dataframe**

Tipo:

Identificador:

Texto query:

Base de Dados:

**Adicionar Parâmetros**

Nome:  Conteúdo:  **ADICIONAR**

**REGISTAR**

**Sucesso**  
Dataframe criado!

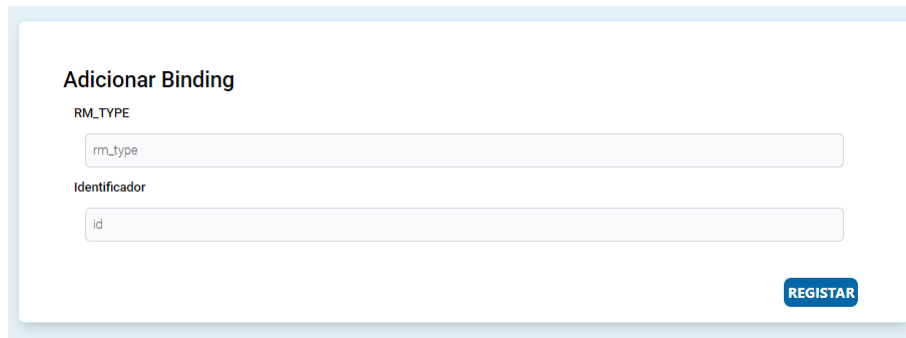
Figura 56: Subject Proxy, Adicionar novo dataframe do tipo Query

Para criar um dataframe do tipo Query, são apresentados os campos presentes na figura [56](#). Assim, o utilizador deve preencher os campos referentes ao identificador e texto da query, indicar o tipo de base

de dados e adicionar uma lista de parâmetros contendo cada entrada o nome e o respetivo conteúdo. Uma vez finalizado este processo, pode clicar no botão registar e é emitido uma mensagem que informa o utilizador se a operação foi ou não bem sucedida.

#### 4.5.5.4 Binding

O utilizador têm ainda a possibilidade de criar novos databinding e adicioná-los ao serviço do Subject Proxy.



The screenshot shows a web form titled "Adicionar Binding". It contains two input fields. The first is labeled "RM\_TYPE" and contains the text "rm\_type". The second is labeled "Identificador" and contains the text "id". At the bottom right of the form is a blue button with the text "REGISTAR".

Figura 57: Subject Proxy, Adicionar novo databinding

Como se pode observar através da análise da figura , a criação do Binding é relativamente simples e intuitiva. Basta preencher os campos necessários, neste caso o identificador e o `rm_type` e clicar no botão registar. No final será emitido uma mensagem informativa que indica ao utilizador se a operação foi ou não bem sucedida.

## Avaliação

Uma fase fundamental no processo de desenvolvimento refere-se à análise da viabilidade da solução apresentada. Nesta etapa é importante estudar a relevância da solução bem como avaliar o seu potencial. Para tal decidiu-se recorrer à Análise SWOT, um método que permite avaliar de forma clara o potencial da solução.

### 5.1 Análise SWOT

A análise SWOT, também conhecida como análise FOFA em português descreve-se nos seguintes parâmetros:

- S/F - *Strengths* ou Forças;
- W/F - *Weaknesses* ou Fraquezas;
- O - *Opportunities* ou Oportunidades;
- T/A - *Threats* ou Ameaças.

Estes elementos permitem relacionar de uma perspectiva interna e externa os pontos fracos e fortes da solução, bem como as ameaças e oportunidades do mercado.

Assim, e em jeito de contextualização as forças e fraquezas dizem respeito a fatores internos e intrínsecos à solução, concentrando-se principalmente no presente, isto é no estado atual da solução, sendo algo controlável pela entidade responsável pela solução.

Por outro lado, as oportunidades e ameaças dizem respeito a fatores externos e relativos ao ambiente que está inserido, estando relacionado com o futuro da solução, ou seja, por outras palavras "o que pode vir a acontecer".

Deste a seguinte tabela pretende enumerar respectivamente os parâmetros acima supracitados, bem como apresentar uma breve descrição para cada e possíveis soluções no casos aplicáveis.



Forças	Explicação	
Interoperabilidade semântica assegurada	O facto do sistema ser baseado no modelo openEHR assegura a capacidade deste em comunicar com outros sistemas informáticos de saúde sem que se perca o significado do seu conteúdo.	
Criação intuitiva e padronizada de guidelines clínicos	O Decision Maker disponibiliza uma interface que facilita a criação de guidelines clínicos de modo padronizado.	
Ligação inteligente a dados	O sistema é capaz de comunicar com o serviço do Subject Proxy que é responsável por fornecer dados onticos e relativos a um determinado sujeito/paciente.	
Integração fácil noutros sistemas informáticos de saúde	O Decision Maker pode ser facilmente adaptado e inserido noutros sistemas informáticos de saúde.	
Fraquezas	Explicação	Solução
Falta de controlo do input inserido nos campos de texto	O sistema não oferece mecanismos de controlo de tipos e sintaxe relativamente ao texto inserido nos campos de texto, como por exemplo: inserção de e-mails invalidos, insereção de texto em campos que deve apenas aceitar valores numéricos, entre outros.	Utilizar expressões regex de controlo no texto inserido, produzindo mensagens de erro informando o utilizador dos campos que necessitam de correção.
Sistema suscetível a ataques informáticos	Ausencia de mecanismos de segurança contra ataques informáticos que podem comprometer o correcto funcionamento do sistema	Pode ser culmatado com a utilização de mecanismos que melhorem a segurança da aplicação, tais como, a utilização de tokens de autenticação, encriptação das mensagens trocadas, entre outros.
Falta de suporte para dispositivos móveis	Devido à falta de reactividade para dispositivos móveis, o utlizador pode ver a sua experiencia limitada.	Este problema pode ser resolvido configurando a reactividade e expandido a sua configuração atual a ecrãs de dispositivos móveis.

Tabela 2: Análise SWOT: Forças e Fraquezas

Oportunidades	Explicação	
Auxiliar os profissionais de saúde no processo de tomada de decisão	Através do uso de guidelines clínicos o utilizador é assistido em tarefas específicas do seu quotidiano de trabalho, esperando-se ver um melhoramento da qualidade dos serviços de saúde prestrados.	
Utilização de um padrão mundialmente reconhecido	O Decision Maker é uma solução baseada nas especificações do Decision Logic do openEHR, um padrão mundialmente reconhecido pela comunidade científica.	
Eliminação de más práticas clínicas e redução/eliminação de erros humanos de trabalho	O sistema oferece uma forma padronizada de criar guidelines clínicos baseados na literatura científica visando eliminar más práticas clínicas.	
Ameaças	Explicação	Solução
Fraca adoção por parte dos utilizadores devido a presunções ou devido à falta de confiança no sistema de decisão	A utilização do Decision Maker pode ficar comprometida devido a certas presunções relativas à adoção de SADCs no quotidiano de trabalho dos profissionais de saúde.	O sistema oferece uma interface simples e intuitiva que visa distanciar-se de SADCs que outrora mostraram ser difíceis e demorosos de utilizar. Para além disto, como o sistema é baseado no modelo openEHR, um padrão mundialmente reconhecido pela comunidade científica, este visa restaurar a confiança na utilização de SADCs e na capacidade de decisão destes sistemas.
Utilização de terminologias técnicas que podem ser desconhecidas ao utilizador	A utilização de terminologias demasiado técnicas a nível da interface pode comprometer o correcto uso e criação de guidelines, limitando assim a longevidade de utilização do Decision Maker	Este problema pode ser mitigado desenvolvendo um guia de utilização. Este guia seria apresentado na forma de mensagens, realçando cada passo individual no processo de criação e utilização de guidelines clínicos. Deste modo, o objetivo é ter mensagens pop-up e botões de ajuda que permitam clarificar conceitos e guiar o utilizador durante a sua navegação pelo sistema.

Tabela 3: Análise SWOT: Oportunidades e Ameaças

## Conclusão

Com o desenvolvimento da dissertação do curso de Mestrado Integrado de Engenharia foi possível adquirir novos conhecimentos relativos à temática em questão.

A fase de pesquisa bibliográfica foi sem dúvida a parte mais extensa e valiosa na realização deste documento, pois permitiu tomar conhecimento de conceitos importantes na área como a definição de sistemas de apoio à decisão clínica, o modelo openEHR, assim como, terminologias específicas do domínio. No entanto, o processo de pesquisa bibliográfica consistiu um desafio, na medida em que o leque de informações é tão vasto que se tornou complicado aglomerar a informação estudada toda num só texto.

A fase de desenvolvimento foi indubitavelmente a etapa mais divertida e ao mesmo tempo difícil. A implementação da aplicação Decision Maker constituiu um desafio que se provou ser capaz de alcançar. O seu desenvolvimento possibilitou a assimilação de conhecimentos valiosos no que toca à adoção de novas tecnologias e à utilização das que já estava à vontade. Complementarmente, o facto de desenvolver uma aplicação de raiz, permitiu expandir e abordar áreas que estava menos à vontade, forçando assim que saísse da minha zona de conforto.

Embora as dificuldades e desafios apresentados, foi-se capaz de ultrapassá-los e produzir, assim, uma solução viável e madura face ao problema que se estava a resolver. Deste modo, a aplicação *web* desenvolvida para criar e gerir *guidelines* clínicos, mostrou-se ser viável na medida que representa uma solução intuitiva, apelativa e de fácil utilização. Como referido ao longo deste relatório de dissertação, esta visa culmutar certas presunções que existem por parte dos profissionais de saúde relativas à sua utilização, pois em muitos casos as soluções apresentadas mostravam ser pouco práticas e difíceis de utilizar. Adicionalmente, mas não menos importante, esta aplicação visa auxiliar o profissional de saúde na tomada de decisões, que através do desenvolvimento de *guidelines* com base em literatura científica permitem produzir *outputs* específicos que no momento de decisão, relativamente a um cuidado específico, mostram-se úteis resultando, assim, no melhoramento dos cuidados prestrados ao paciente.

É importante salientar que esta aplicação integrou meios que possibilitaram a obtenção de dados valiosos e o mais atuais possíveis, relativos ao sujeito alvo de cuidados. Este factor auxilia imenso o profissional de saúde, na medida em que torna mais eficiente o seu trabalho, pois caso deseje, não necessita de se preocupar com a obtenção de dados específicos e que por vezes são provenientes de

diversas fontes. A aplicação em si fornece os meios necessários para obter estes dados através do serviço fornecido pelo Subject Proxy.

A par com a revisão literária e a implementação da aplicação *web*, elaborou-se um artigo científico com base no presente projecto que ainda se encontra em fase de avaliação.

Futuramente, apesar de a solução implementada ser madura, há um vasto leque de funcionalidades que podem ser facilmente integrados. Uma funcionalidade que pode-ser implementada futuramente é a criação de regras que utilizem funções para produzir o *output* desejado. Outra funcionalidade que pode ser útil é a criação de janelas informativas de ajuda que permitam guiar o utilizador no processo de criação e de execução de *guidelines*. Por fim, seria interessante desenvolver *guidelines baseados em pontuação* que estratificam riscos associados a um determinado tipo de intervenção.

Conclusivamente, os objetivos delineados inicialmente foram cumpridos, pelo que se pode afirmar que a solução produzida apresenta uma solução viável e madura de utilização.

## Bibliografia

- [1] S. S. and Alqahtani et al. “The Implementation of Clinical Decision Support System: A Case Study in Saudi Arabia”. Em: *International Journal of Information Technology and Computer Science* 8.8 (2016), pp. 23–30. issn: 20749007. doi: [10.5815/ijitcs.2016.08.03](https://doi.org/10.5815/ijitcs.2016.08.03) (ver p. 8).
- [2] G. Bacelar e R. Correia. “As bases do openEHR”. Em: September (2015), p. 42. doi: [10.13140/RG.2.1.3248.9687](https://doi.org/10.13140/RG.2.1.3248.9687) (ver pp. 2, 12–14).
- [3] W. G. Baxt. “Application of artificial neural networks to clinical medicine”. Em: *The Lancet* 346.8983 (1995), pp. 1135–1138. issn: 01406736. doi: [10.1016/S0140-6736\(95\)91804-3](https://doi.org/10.1016/S0140-6736(95)91804-3) (ver p. 8).
- [4] C. A. F. BISPO. “Uma análise da nova geração de sistemas de apoio à decisão”. Em: (1998), p. 174 (ver p. 6).
- [5] H. L. Bleich. “Computer evaluation of acid-base disorders”. eng. Em: *The Journal of clinical investigation* 48.9 (set. de 1969), pp. 1689–1696. issn: 0021-9738. doi: [10.1172/JCI106134](https://doi.org/10.1172/JCI106134). url: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC535740/> (ver p. 5).
- [6] A. A. Boxwala et al. “GLIF3: A representation format for sharable computer-interpretable clinical practice guidelines”. Em: *Journal of Biomedical Informatics* 37.3 (2004), pp. 147–161. issn: 15320464. doi: [10.1016/j.jbi.2004.04.002](https://doi.org/10.1016/j.jbi.2004.04.002) (ver pp. 17, 18).
- [7] M. F. COLLEN et al. “AUTOMATED MULTIPHASIC SCREENING AND DIAGNOSIS”. Em: *American journal of public health and the nation’s health* 54 (1964), pp. 741–750. issn: 0002-9572. doi: [10.2105/ajph.54.5.741](https://doi.org/10.2105/ajph.54.5.741). url: <https://doi.org/10.2105/ajph.54.5.741> (ver p. 5).
- [8] M. contributors. *Introdução Express/Node*. 2022. url: [https://developer.mozilla.org/pt-BR/docs/Learn/Server-side/Express\\_Nodejs/Introduction](https://developer.mozilla.org/pt-BR/docs/Learn/Server-side/Express_Nodejs/Introduction) (ver p. 34).
- [9] M. David. *Working With MERN (MongoDB, ExpressJS, ReactJS, and NodeJS) Stack*. 2021. url: <https://www.simplilearn.com/working-with-mern-stack-article> (ver p. 35).

- [10] P. A. De Clercq et al. "Approaches for creating computer-interpretable guidelines that facilitate decision support". Em: *Artificial Intelligence in Medicine* 31.1 (2004), pp. 1–27. issn: 09333657. doi: [10.1016/j.artmed.2004.02.003](https://doi.org/10.1016/j.artmed.2004.02.003) (ver pp. 2, 3).
- [11] R. C. Deo. "Machine learning in medicine". Em: *Circulation* 132.20 (2015), pp. 1920–1930. issn: 15244539. doi: [10.1161/CIRCULATIONAHA.115.001593](https://doi.org/10.1161/CIRCULATIONAHA.115.001593) (ver p. 8).
- [12] F. T. de Dombal et al. "Computer-aided diagnosis of acute abdominal pain". eng. Em: *British medical journal* 2.5804 (abr. de 1972), pp. 9–13. issn: 0007-1447. doi: [10.1136/bmj.2.5804.9](https://doi.org/10.1136/bmj.2.5804.9). url: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1789017/> (ver p. 5).
- [13] F. T. de Dombal et al. "Construction and uses of a 'data-base' of clinical information concerning 600 patients with acute abdominal pain". Em: *Proceedings of the Royal Society of Medicine* 64.9 (1971), p. 978. issn: 0035-9157. url: <https://europepmc.org/articles/PMC1812837> (ver p. 5).
- [14] D. Dowding et al. "Nurses' use of computerised clinical decision support systems: a case site analysis." eng. Em: *Journal of clinical nursing* 18.8 (abr. de 2009), pp. 1159–1167. issn: 1365-2702 (Electronic). doi: [10.1111/j.1365-2702.2008.02607.x](https://doi.org/10.1111/j.1365-2702.2008.02607.x) (ver p. 11).
- [15] U. E. "Clinical guidelines". Em: *Practising Midwife* 19 (2016), pp. 13–16. doi: [10.7748/phc.9.1.28.s14](https://doi.org/10.7748/phc.9.1.28.s14) (ver p. 1).
- [16] I. C. Education. *Three-Tier Architecture*. 2020. url: <https://www.ibm.com/cloud/learn/three-tier-architecture> (ver p. 42).
- [17] K. Fehre. *How to write Arden Syntax y MLMs - An introduction*. url: <http://www.medexter.com/phocadownload/pr/arden%20syntax%20tutorial.pdf> (ver p. 15).
- [18] J. Feiler. "Architecture Overview". Em: *Mac OSX Developer Guide* (2002), pp. 21–48. doi: [10.1016/b978-012251341-1/50004-x](https://doi.org/10.1016/b978-012251341-1/50004-x) (ver p. 13).
- [19] openEHR Foundation. *What is openEHR?* 2021. url: [http://www.openehr.org/what\\_is\\_openehr](http://www.openehr.org/what_is_openehr) (ver p. 2).
- [20] J. Fox e S. Das. "Safe and sound". Em: *Artificial intelligence in hazardous applications* 307 (2000) (ver p. 19).
- [21] J. Fox, R. Thomson e M. A. Project Manager. "Decision support for health care: the PROforma evidence base Vivek Patkar MBBS MS Clinical Research Fellow PROforma: modelling decisions and care pathways". Em: (2006), pp. 49–54 (ver p. 19).
- [22] "Fundamentals of clinical data science". Em: *Fundamentals of Clinical Data Science* (2018), pp. 1–219. doi: [10.1007/978-3-319-99713-1](https://doi.org/10.1007/978-3-319-99713-1) (ver p. 3).
- [23] Geeksforgeeks e @Jasraj. *MERN Stack*. 2021. url: <https://www.geeksforgeeks.org/mern-stack/> (ver pp. 33, 34).

- [24] F. Hak et al. "An OpenEHR Adoption in a Portuguese Healthcare Facility". Em: *Procedia Computer Science* 170 (2020), pp. 1047–1052. issn: 18770509. doi: [10.1016/j.procs.2020.03.075](https://doi.org/10.1016/j.procs.2020.03.075). url: <https://doi.org/10.1016/j.procs.2020.03.075> (ver pp. 12, 14).
- [25] M. Healthcare. "Medical logic modules". Em: (). url: [https://www.medexter.com/ardensuite/educational\\_material\\_part2\\_arden\\_syntax\\_overview.pdf](https://www.medexter.com/ardensuite/educational_material_part2_arden_syntax_overview.pdf) (ver p. 16).
- [26] G. Hripcsak. "The Arden Syntax for medical logic modules: Introduction". Em: *Computers in Biology and Medicine* 24.5 (1994), pp. 329–330. issn: 00104825. doi: [10.1016/0010-4825\(94\)90001-9](https://doi.org/10.1016/0010-4825(94)90001-9) (ver p. 15).
- [27] H. Information e M. S. Society. *EMRAM: A strategic roadmap for effective EMR adoption and maturity*. url: <https://www.himssanalytics.org/emram> (ver p. 3).
- [28] J. Kabachinski. "A look at clinical decision support systems". Em: *Biomedical instrumentation & technology* 47.5 (2013), p. 432 (ver p. 12).
- [29] V. Kreinovich e N. H. Phuong. *Soft Computing for Biomedical Applications and*. isbn: 9783030495350. url: <https://books.google.com.lb/books?hl=en&lr=&id=t1juDwAAQBAJ&oi=fnd&pg=PA37&dq=Artificial+Intelligence+in+Infection+Control%E2%80%9494Healthcare+Institutions+Need+Intelligent+Information+and+Communication+Technologies+for+Surveillance+and+Benchmarking&ots=B12zk0nXjN> (ver p. 15).
- [30] B. Kuechler e S. Petter. "Design Science Research in". Em: 1 (2012), pp. 1–66. doi: [1756-0500-5-79](https://doi.org/10.1186/1756-0500-5-79). url: <http://www.desrist.org/design-research-in-information-systems/> [Accessed%2011%20may%202017] (ver p. 32).
- [31] G. J. Kuperman, R. M. Gardner e T. A. Pryor. "Decision Support on the HELP System". Em: *HELP: A Dynamic Hospital Information System*. New York, NY: Springer New York, 1991, pp. 53–67. isbn: 978-1-4612-3070-0. doi: [10.1007/978-1-4612-3070-0\\_5](https://doi.org/10.1007/978-1-4612-3070-0_5). url: [https://doi.org/10.1007/978-1-4612-3070-0\\_5](https://doi.org/10.1007/978-1-4612-3070-0_5) (ver p. 6).
- [32] R. Kwok et al. "Improving adherence to asthma clinical guidelines and discharge documentation from emergency departments: implementation of a dynamic and integrated electronic decision support system." eng. Em: *Emergency medicine Australasia : EMA* 21.1 (fev. de 2009), pp. 31–37. issn: 1742-6723 (Electronic). doi: [10.1111/j.1742-6723.2008.01149.x](https://doi.org/10.1111/j.1742-6723.2008.01149.x) (ver p. 10).
- [33] D. P. Lacerda et al. "Design Science Research: A research method to production engineering". Em: *Gestão & Produção* 20.4 (2013), pp. 741–761. doi: [10.1590/S0104-530X2013005000014](https://doi.org/10.1590/S0104-530X2013005000014) (ver pp. 31, 32).

- [34] R. S. Ledley e L. B. Lusted. "Reasoning Foundations of Medical Diagnosis". Em: *Science* 130.3366 (1959), pp. 9–21. issn: 0036-8075. doi: [10.1126/science.130.3366.9](https://doi.org/10.1126/science.130.3366.9). eprint: <https://science.sciencemag.org/content/130/3366/9.full.pdf>. url: <https://science.sciencemag.org/content/130/3366/9> (ver p. 5).
- [35] A. A. Litvin e V. A. Litvin. *Clinical decision support systems for surgery*. Vol. 22. 1. 2014, pp. 96–100. isbn: 9783319319117. doi: [10.18484/2305-0047.2014.1.96](https://doi.org/10.18484/2305-0047.2014.1.96) (ver p. 8).
- [36] N. Manson. "Is operations research really research?" Em: *ORiON* 22.2 (2006), pp. 155–180. issn: 0259-191X. doi: [10.5784/22-2-40](https://doi.org/10.5784/22-2-40) (ver p. 32).
- [37] G. Marakas. *Decision support systems in 21st century–US edition*. 1999 (ver p. 8).
- [38] C. J. McDonald et al. "The Regenstrief Medical Record System: a quarter century experience." eng. Em: *International journal of medical informatics* 54.3 (jun. de 1999), pp. 225–253. issn: 1386-5056 (Print). doi: [10.1016/s1386-5056\(99\)00009-x](https://doi.org/10.1016/s1386-5056(99)00009-x) (ver p. 6).
- [39] MongoDB. *MERN Stack Explained*. url: <https://www.mongodb.com/mern-stack> (ver pp. 33, 34).
- [40] NodeJs. *About Node.js*. url: <https://nodejs.org/en/about/> (ver p. 34).
- [41] L. Ohno-Machado et al. "The guideline interchange format: a model for representing guidelines". eng. Em: *Journal of the American Medical Informatics Association : JAMIA* 5.4 (1998), pp. 357–372. issn: 1067-5027. doi: [10.1136/jamia.1998.0050357](https://doi.org/10.1136/jamia.1998.0050357). url: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC61313/> (ver pp. 6, 17, 18).
- [42] D. Oliveira et al. "Management of a Pandemic Based on an openEHR approach". Em: *Procedia Computer Science* 177 (2020), pp. 522–527. doi: [10.1016/j.procs.2020.10.072](https://doi.org/10.1016/j.procs.2020.10.072) (ver p. 13).
- [43] D. Oliveira et al. "ScienceDirect Steps towards an Healthcare Information Model based on openEHR". Em: *Procedia Computer Science* 184 (2021), pp. 893–898. doi: [10.1016/j.procs.2021.04.015](https://doi.org/10.1016/j.procs.2021.04.015). url: [www.sciencedirect.comwww.sciencedirect.comwww.elsevier.com/locate/procedia](http://www.sciencedirect.com/locate/procedia) (ver p. 13).
- [44] OpenEHR. *Foundation Types*. url: [https://specifications.openehr.org/releases/BASE/latest/foundation\\_types.html](https://specifications.openehr.org/releases/BASE/latest/foundation_types.html) (ver p. 25).
- [45] OpenEHR. *openEHR Platform Service Model*. url: [https://specifications.openehr.org/releases/SM/latest/openehr\\_platform.html](https://specifications.openehr.org/releases/SM/latest/openehr_platform.html) (ver p. 67).
- [46] "OpenEHR modeling: improving clinical records during the COVID-19 pandemic". Em: *Health and Technology* 11 (5 2021), pp. 1109–1118. issn: 21907196. url: <https://doi.org/10.1007/s12553-021-00556-4> (ver p. 13).



- [47] W. Oude Nijeweme -D'hollosy et al. "Clinical Decision Support Systems for Primary Care: The Identification of Promising Application areas and an Initial Design of a CDSS for lower back pain". Em: November (2015). doi: [10.13140/RG.2.1.3821.0003](https://doi.org/10.13140/RG.2.1.3821.0003) (ver pp. 1, 8).
- [48] K. Peffers et al. "A Design Science Research Methodology for Information Systems Research". Em: *Journal of Management Information Systems* 24.3 (2007), pp. 45–77. doi: [10.2753/MIS0742-1222240302](https://doi.org/10.2753/MIS0742-1222240302). eprint: <https://doi.org/10.2753/MIS0742-1222240302>. url: <https://doi.org/10.2753/MIS0742-1222240302> (ver p. 32).
- [49] M. Peleg et al. "Sharable representation of clinical guidelines in GLIF: Relationship to the Arden Syntax". Em: *Journal of Biomedical Informatics* 34.3 (2001), pp. 170–181. issn: 15320464. doi: [10.1006/jbin.2001.1016](https://doi.org/10.1006/jbin.2001.1016) (ver p. 18).
- [50] L. E. Perreault e J. B. Metzger. "A pragmatic framework for understanding clinical decision support". Em: *Journal of Healthcare Information Management* 13 (1999), pp. 5–22 (ver p. 9).
- [51] M. Pimentel e D. Filippo. "Design Science Research: pesquisa científica atrelada ao design de artefatos". Em: 3 (2020), pp. 37–61 (ver p. 31).
- [52] D. J. Power. "Decision Support Systems: A Historical Overview". Em: *Handbook on Decision Support Systems 1: Basic Themes*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 121–140. isbn: 978-3-540-48713-5. doi: [10.1007/978-3-540-48713-5\\_7](https://doi.org/10.1007/978-3-540-48713-5_7). url: [https://doi.org/10.1007/978-3-540-48713-5\\_7](https://doi.org/10.1007/978-3-540-48713-5_7) (ver p. 5).
- [53] P. Ram et al. "Executing clinical practice guidelines using the SAGE execution engine." eng. Em: *Studies in health technology and informatics* 107.Pt 1 (2004), pp. 251–255. issn: 0926-9630 (Print) (ver p. 7).
- [54] React. *React*. url: <https://reactjs.org/> (ver p. 34).
- [55] K. Sadegh-Zadeh. *Clinical Decision Support Systems*. Vol. 119. 2015, pp. 705–722. isbn: 9780387339146. doi: [10.1007/978-94-017-9579-1\\_20](https://doi.org/10.1007/978-94-017-9579-1_20) (ver p. 8).
- [56] E. H. Shortliffe et al. "Computer-based consultations in clinical therapeutics: explanation and rule acquisition capabilities of the MYCIN system". Em: *Computers and biomedical research, an international journal* 8.4 (1975), pp. 303–320. issn: 0010-4809. doi: [10.1016/0010-4809\(75\)90009-9](https://doi.org/10.1016/0010-4809(75)90009-9). url: [https://doi.org/10.1016/0010-4809\(75\)90009-9](https://doi.org/10.1016/0010-4809(75)90009-9) (ver p. 5).
- [57] E. H. Shortliffe. "Biomedical Informatics: Defining the Science and Its Role in Health Professional Education". Em: *Information Quality in e-Health*. Ed. por A. Holzinger e K.-M. Simonc. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 711–714. isbn: 978-3-642-25364-5 (ver p. 8).
- [58] openEHR Specification Program. "CDS, Guidelines and Planning Overview". Em: (). url: [https://specifications.openehr.org/releases/PROC/latest/overview.html#\\_conceptual/\\_framework](https://specifications.openehr.org/releases/PROC/latest/overview.html#_conceptual/_framework) (ver pp. 20–22, 25, 29, 30, 51, 58, 66).

- [59] openEHR Specification Program. *Decision Language Specification*. url: [https://specifications.openehr.org/releases/PROC/Release-1.6.0/decision\\_language.html#\\_decision\\_language\\_specification](https://specifications.openehr.org/releases/PROC/Release-1.6.0/decision_language.html#_decision_language_specification) (ver pp. 21, 23–29).
- [60] openEHR Specification Program. *Task Planning (TP) Specification*. url: [https://specifications.openehr.org/releases/PROC/latest/task\\_planning.html](https://specifications.openehr.org/releases/PROC/latest/task_planning.html) (ver p. 20).
- [61] D. R. Sutton e J. Fox. “The syntax and semantics of the PRO forma guideline modeling language”. Em: *Journal of the American Medical Informatics Association* 10.5 (2003), pp. 433–443 (ver p. 19).
- [62] R. T. Sutton et al. *An overview of clinical decision support systems: benefits, risks, and strategies for success*. 2020. doi: [10.1038/s41746-020-0221-y](https://doi.org/10.1038/s41746-020-0221-y) (ver pp. 1, 9, 10, 12).
- [63] H. Takeda, P. Veerkamp e H. Yoshikawa. “Modeling Design Process”. Em: *AI Magazine* 11.4 (dez. de 1990), p. 37. doi: [10.1609/aimag.v11i4.855](https://ojs.aaai.org/index.php/aimagazine/article/view/855). url: <https://ojs.aaai.org/index.php/aimagazine/article/view/855> (ver p. 32).
- [64] L. Thé. “OLAP Answers Tough Business Questions”. Em: *Datamation* 41.8 (1995), pp. 65–67. issn: 0011-6963 (ver pp. 6, 7).
- [65] S. Tiffe. “Arden Syntax for Medical Logic Systems”. Em: *Practice* April 1992 (2005), pp. 1–116 (ver p. 6).
- [66] H. R. WARNER et al. “A mathematical approach to medical diagnosis. Application to congenital heart disease”. Em: *JAMA* 177 (1961), pp. 177–183. issn: 0098-7484. doi: [10.1001/jama.1961.03040290005002](https://doi.org/10.1001/jama.1961.03040290005002). url: <https://doi.org/10.1001/jama.1961.03040290005002> (ver p. 5).
- [67] J. Weldon. “A career in data modeling”. Em: 1997 (ver pp. 6, 7).
- [68] Wolandscat. *Towards a standard analysis of computable guidelines, clinical workflow, decision support and ... the curly braces problem*. 200. url: <https://wolandscat.net/2020/06/24/towards-a-standard-analysis-of-computable-guidelines-clinical-workflow-decision-support-and-the-curly-braces-problem/> (ver p. 20).
- [69] A. Wright e D. F. Sittig. “A four-phase model of the evolution of clinical decision support architectures”. Em: *International Journal of Medical Informatics* 77.10 (2008), pp. 641–649. issn: 13865056. doi: [10.1016/j.ijmedinf.2008.01.004](https://doi.org/10.1016/j.ijmedinf.2008.01.004) (ver pp. 5, 6).
- [70] A. Wulff et al. “An interoperable clinical decision-support system for early detection of SIRS in pediatric intensive care using openEHR”. Em: *Artificial Intelligence in Medicine* 89.September 2017 (2018), pp. 10–23. issn: 18732860. doi: [10.1016/j.artmed.2018.04.012](https://doi.org/10.1016/j.artmed.2018.04.012) (ver p. 16).
- [71] P. P. Yu. “Knowledge bases, clinical decision support systems, and rapid learning in oncology”. Em: *Journal of Oncology Practice* 11.2 (2015), e206–e211. issn: 1935469X. doi: [10.1200/JOP.2014.000620](https://doi.org/10.1200/JOP.2014.000620) (ver p. 1).





