NITERÓI - RJ

SBCAS 2019

19° SIMPÓSIO BRASILEIRO DE COMPUTAÇÃO APLICADA À SAÚDE

DE 11 A 14 DE JUNHO DE 2019 EM NITERÓI-RJ

www.sbc.org.br/sbcas2019

LIVRO DE MINICURSOS DO SBCAS 2019

REALIZAÇÃO

SBC

PATROCÍNIO

CAPES CNPq

APOIO

sbis ABTms

ORGANIZAÇÃO

Laboratório MídiaCom — Instituto de Computação

19º Simpósio Brasileiro de Computação Aplicada à Saúde

Niterói, RJ, 11 a 14 de junho de 2019

LIVRO DE MINICURSOS DO SBCAS 2019

Organização do Livro
Artur Ziviani (LNCC), Natalia Castro Fernandes (UFF) e Débora Christina Muchaluat Saade (UFF)

Coordenação-Geral
Débora Christina Muchaluat Saade (UFF)

Realização
Universidade Federal Fluminense – UFF
Sociedade Brasileira de Computação – SBC

19a Edição
Sociedade Brasileira de Computação – SBC
Porto Alegre

Capa: Emanuel Antunes Machado
Editoração: Natalia Castro Fernandes

Simpósio Brasileiro de Computação Aplicada à Saúde (19. :2019 : Niterói, RJ)

Livro de Minicursos [do] 19o Simpósio Brasileiro de Computação Aplicada à Saúde, 11 a 14 de junho de 2019 / Sociedade Brasileira de Computação ; Organizadores, Artur Ziviani, Natalia Castro Fernandes e Débora Christina Muchaluat Saade -Niterói, RJ: Sociedade Brasileira de Computação, 2019.
196 p.

ISBN: 978-85-7669-472-4

1. Ciência da computação. 2. Saúde. 3. Sistemas de computação. I. Artur Ziviani (org.) II. Natalia Castro Fernandes (org.), Débora Christina Muchaluat Saade (org.) IV Título.

(19. ed)

Sociedade Brasileira de Computação – SBC

Av. Bento Gonçalves, 9500
Setor 4 - Prédio 43.412 - Sala 219
Bairro Agronomia
91.509-900 – Porto Alegre RS

CNPJ: 29.532.264/0001-78
http://www.sbrc.org.br

Mensagem da Coordenação Geral

Gostaríamos de dar as boas-vindas a todos os participantes do 19a. Simpósio Brasileiro de Computação Aplicada à Saúde (SBCAS 2019), realizado no Instituto de Computação da Universidade Federal Fluminense de 11 a 14 de junho de 2019 na cidade de Niterói, RJ. É uma grande satisfação poder reunir as comunidades de computação e de saúde em um evento multidisciplinar, que promove e incentiva a realização de pesquisas e trabalhos inovadores com potencial de trazer benefícios diretos à nossa sociedade. A programação técnica do SBCAS 2019 está bastante abrangente e de grande qualidade, formada por minicursos, sessões técnicas, palestras, painéis, Workshop de Ferramentas e Aplicações – WFA, Concurso de Teses e Dissertações – CTD, Concurso de Trabalhos de Iniciação Científica, Trilha de Grandes Desafios em Saúde Digital e II Hackathon sobre Desafios em Saúde. A qualidade desta grade de programação é resultado do empenho de diversas pessoas. Portanto, um agradecimento a todos os membros do comitê de programa e em especial aos coordenadores do comitê de programa da trilha principal e da trilha de minicursos Artur Ziviani (LNCC) e Natalia Castro Fernandes (UFF), aos coordenadores do Workshop de Ferramentas e Aplicações – WFA – Rodrigo Rafael Villarreal Goulart (Feevale) e Flávio Luiz Seixas (UFF), do Concurso de Teses e Dissertações – CTD – Paulo Eduardo Ambrósio (UESC) e Alexandre Sztajnberg (UERJ), do Concurso de Trabalhos de Iniciação Científica – CTIC – Lucas Ferrari de Oliveira (UFPR) e Luciana Cardoso de Castro Salgado (UFF), da trilha de Grandes Desafios em Saúde Digital – GDS – Alexandra Monteiro (UERJ/ABTms) e Marcia Ito (Fatec-SP/SBIS) e do II Hackathon sobre Desafios em Saúde Digital Flávio Luiz Seixas (UFF) e Luciana Cardoso de Castro Salgado (UFF). Agradecemos o apoio financeiro da CAPES, do CNPq, e do Programa de Pós- graduação em Computação da UFF, sem o qual seria inviável realizar o evento. Agradecemos o apoio da Sociedade Brasileira de Computação (SBC), de sua Comissão Especial de Computação Aplicada à Saúde (CE-CAS) e de sua Secretaria Regional do Rio de Janeiro, representada pela Profa. Flavia Cristina Bernardini, na realização do evento. Agradecemos também o apoio institucional da Sociedade Brasileira de Informática em Saúde (SBIS) e da Associação Brasileira de Teledicina e Telessaúde (ABTms). Agradecemos o apoio local do Laboratório MídiaCom, do Instituto de Computação, na pessoa de seu Diretor Prof. José Raphael Bokehi, e da reitoria da Universidade Federal Fluminense, na pessoa do Magnífico Reitor Prof. Antonio Claudio Lucas da Nóbrega. Por fim, fazemos um agradecimento especial ao grupo de alunos de graduação e de pós-graduação da UFF, que atuam como voluntários durante a realização do SBCAS 2019. Sem o apoio de todos os colaboradores, seria inviável a realização deste evento. Em nome da comissão organizadora do SBCAS 2019, desejamos a todos os participantes dias enriquecedores em conhecimento e troca de experiências e de agradável convívio em Niterói.

Débora Christina Muchaluat Saade, UFF
Coordenadora Geral do SBCAS 2019

Mensagem da Coordenação de Minicursos

O Livro de Minicursos do 19o Simpósio Brasileiro de Computação Aplicada à Saúde (SBCAS 2019) traz os textos dos minicursos selecionados e apresentados nessa edição do evento, incluindo quatro minicursos aceitos. Os minicursos são uma oportunidade de atualizar os conhecimentos da comunidade com novos temas relacionados à saúde e à computação, de uma forma didática e de amplo acesso ao público.

Os minicursos foram selecionados através de um processo de revisão por pares do tipo "blind", de um total de 12 propostas de minicursos submetidas, o que implicou uma taxa de aceitação de 33%. O comitê de avaliação contou com 10 pesquisadores de grande renome na área, que realizaram uma criteriosa seleção. Agradecemos a todos os membros do comitê, por suas valiosas contribuições para os trabalhos e dedicação no processo de revisão.

Os quatro trabalhos selecionados apresentam alta qualidade, resultado do esforço dos autores, que dedicaram muitas horas para a produção do conteúdo escrito e da apresentação. Somos gratos a eles também pelo esforço aplicado e pelo sucesso do trabalho.

Agradecemos também ao Comitê de Organização do SBCAS 2019, em especial à Coordenadora Geral, Prof. Débora Saade, por todo o suporte durante a seleção e elaboração desse livro.

Finalmente, desejamos que todos os participantes dos minicursos do SBCAS 2019 aproveitem as apresentações!

Artur Ziviani, LNCC
Natalia Castro Fernandes, UFF
Coordenadores de Minicursos do SBCAS 2019

Comitê de Programa de Minicursos

Artur Ziviani (LNCC),
Débora Muchaluat-Saade (UFF),
Lucas Ferrari de Oliveira (UFPR),
José Raphael Bokehi(SBEB e UFF),
Márcia Ito (FATEC),
Marco Antonio Gutierrez(InCor/USP),
Natalia Castro Fernandes (UFF),
Paulo Eduardo Ambrósio (UESC),
Rodrigo Rafael Villarreal Goulart (FEEVALE),
Sergio Miranda Freire(UERJ)

CE-CAS – Comissão Especial de Computação Aplicada à Saúde

Débora Christina Muchaluat Saade, UFF (Coordenadora)
Lucas Ferrari de Oliveira, UFPR (Vice-Coordenador)
Artur Ziviani, LNCC
José Raphael Bokehi, UFF
Márcia Ito, IBM Research Brasil
Marco Antonio Gutierrez, USP – InCor
Paulo Eduardo Ambrósio, UESC
Rodrigo Rafael Villarreal Goulart, FEEVALE
Sergio Miranda Freire, UERJ

Sumário

Capítulo 1

Machine Learning aplicado à Saúde

André Filipe de Moraes Batista, Alexandre Dias Porto Chiavegatto Filho

Laboratório de Big Data e Análise Preditiva em Saúde (LABDAPS)
Faculdade de Saúde Pública da Universidade de São Paulo

Abstract

The increase in the complexity of computationally-treated challenges as well as the large amount of data generated by different activities fosters the need for sophisticated computational tools that are capable of inducing knowledge from past experience to solve real problems, an area known as Machine Learning. These algorithms learn to induce a function or hypothesis capable of solving a problem from previous data that represent instances of the problems to be solved. In the area of Health Informatics, the data available in electronic health records can serve as input for Machine Learning models aimed at improving diagnoses and predictions, leading to better treatments for patients. This mini-course will present the current landscape of the use of Machine Learning techniques applied in the health area, contextualizing the main algorithms currently used

Resumo

O aumento na complexidade dos problemas a serem tratados computacionalmente, assim como no volume de dados gerados por diferentes atividades, fomenta a necessidade de ferramentas computacionais sofisticadas que sejam capazes de induzir uma hipótese a partir da experiência passada para solucionar problemas reais. A esse cenário dá-se o nome Machine Learning. Algoritmos de Machine Learning aprendem a induzir uma função ou hipótese capaz de resolver um problema a partir de dados que representam observações dos problemas a ser resolvido. Na área de Informática em Saúde, os dados disponibilizados nos registros eletrônicos podem servir de insumos para modelos de Machine Learning voltados para o aprimoramento de diagnósticos e predições, possibilitando um melhor tratamento para os doentes. Este minicurso irá apresentar o panorama atual do uso das técnicas de Machine Learning aplicadas na área de saúde, contextualizando os principais algoritmos utilizados.

1.1. Introdução

Nos últimos anos a ciência tem se deparado com uma enorme quantidade de dados provenientes das mais diversas fontes como, por exemplo, a Internet, as mídias sociais, os dispositivos de coleta de dados, etc. Diversas agências de pesquisas estão financiando projetos que buscam promover a gestão, o acesso e a descoberta de conhecimento a partir de tais dados. O cenário formado por amplos conjuntos de dados em larga escala, normalmente gerados e consumidos em tempo real é conhecido como "dilúvio de dados" (em inglês, *Data Deluge*) (SELTZER; ZHANG, 2009; MATTMANN, 2013).

O termo Big Data tem sido utilizado para representar esta metáfora. De acordo com Mattmann (2013), existem três dimensões que classificam um conjunto de dados como big data, sendo elas: o *volume* dos dados que um sistema recebe, processa e/ou dissemina; a *variedade*, isto é, o número e a complexidade dos tipos de dados manipulados; e a *velocidade* com o qual os dados são criados e/ou disponibilizados para outros usos.

O uso de Big Data em saúde é a grande novidade da área nos últimos anos (CHIAVEGATTO FILHO, 2015). Hospitais e governos têm acumulado uma quantidade imensa de dados sobre diagnósticos clínicos e tratamento de doenças. Há a necessidade para que esses dados sejam utilizados para o benefício da saúde dos pacientes, com o objetivo de melhorar a tomada de decisão e contribuir para a melhoria da saúde pública.

Devido aos grandes e crescentes volume de dados, a Inteligência Artificial (IA) ganhou notória popularidade nas últimas décadas. Machine Learning (ML) é uma das áreas da Inteligência Artificial que vem apresentando maior destaque na área da saúde e em outras áreas do conhecimento, apoiando-se na teoria de que os computadores podem aprender com dados sem programação manual para executar determinadas atividades.

Tendo em vista a sua inerente complexidade, a atenção à saúde ainda é dominada por incertezas, com frequentes mudanças de protocolos e práticas clínicas. O uso de modelos preditivos de machine learning tem o potencial de auxiliar na tomada de decisão nos diversos momentos da atenção à saúde, especialmente no diagnóstico, intervenção e acompanhamento de problemas de saúde (OBERMEYER; LEE, 2017).

De acordo com Obermeyer e Emanuel (2016), os modelos de Machine Learning abordam problemas de maneira análoga a médicos que estão no período de residência: aprendendo as regras a partir dos dados. Tais modelos iniciam com observações no nível do paciente, percorrendo um grande número de variáveis, na busca por predizer resultados de maneira confiável. Tal processo não é muito diferente de um modelo tradicional de regressão: há um resultado a ser modelado, covariáveis e uma função estatística relacionando-os. Porém, o que faz dos modelos de Machine Learning promissores é a capacidade de lidar com um grande número de preditores – às vezes, inclusive, superior ao número de observações – e combiná-los de maneiras não-lineares e altamente interativas, obtendo resultados superiores aos métodos tradicionais.

Este Capítulo tem por objetivo apresentar os principais conceitos de Machine Learning, juntamente com a implementação de modelos de aprendizado supervisionados, aplicados em banco de dados sintéticos para a predição de risco cardiovascular (LADERAS et al., 2018). Os modelos foram implementados fazendo-se uso da biblioteca Scikit-Learn (BUITINCK et al., 2013) da linguagem de programação Python, sobre a qual os

principais fundamentos são apresentados.

1.2. Panorama do uso de Machine Learning na Área da Saúde

Diversos estudos têm sido desenvolvidos no âmbito de machine learning aplicado à saúde. O crescimento das pesquisas nessa área ocorrem conjuntamente ao aumento da demanda por métodos que possam facilitar diagnósticos e otimizar o tempo dos profissionais de saúde.

Weng et al. (2017) objetivaram comparar modelos resultantes de quatro algoritmos de machine learning com recomendações estabelecidas pelo Colégio Americano de Cardiologia para predizer doença cardiovascular em 10 anos, utilizando dados de uma coorte prospectiva de 378.256 pacientes. O modelo de machine learning de redes neurais apresentou o melhor desempenho dentre os modelos analisados, resultando em 355 predições corretas adicionais de doença cardiovascular quando comparado ao modelo baseado nas recomendações do Colégio Americano de Cardiologia.

O estudo de Green (2018) teve como objetivo compreender os determinantes de saúde utilizando machine learning para construir modelos preditivos para a ocorrência de doenças (e.g., hipertensão) em 1 e em 5 anos a partir do baseline. O modelo desenvolvido fez uso dos dados da pesquisa social *Understanding Society*, que coletou dados pessoais, sociais, de saúde, biomarcadores e genéticos de cerca de 6800 indivíduos. Os resultados obtidos indicam que o uso de dados de saúde para a construção de modelos preditivos apresentou o melhor resultados dentre os testes realizados, com uma acurácia de predição de 71%. Dentre os dados de saúde, a atividade física e a presença de algumas condições de saúde foram fortes preditores individuais.

No contexto brasileiro, Olivera et al. (2017) desenvolveram modelos preditivos de diabetes não diagnosticada a partir de dados de 12.447 adultos entrevistados para o Estudo Longitudinal de Saúde do Adulto (ELSA), utilizando cinco algoritmos de machine learning. A frequência de diabetes não diagnosticada foi de 11%. Entre os 403 indivíduos do conjunto de dados de teste que tinham diabetes não diagnosticada, 274 foram identificados como casos positivos.

1.3. Machine Learning

O aumento na complexidade dos problemas a serem tratados computacionalmente, assim como no volume de dados gerados por diferentes atividades, fomenta a necessidade de ferramentas computacionais sofisticadas que sejam capazes de induzir uma hipótese a partir da experiência passada para a solução de problemas reais. A este cenário dá-se o nome de Machine Learning (ML).

Por meio das técnicas de Machine Learning, computadores são programados para aprender com a experiência passada. Para isso, empregam um princípio de inferência denominado indução. Dessa forma, algoritmos de ML aprendem a induzir uma função ou hipótese capaz de resolver um problema a partir de dados que representam observações do problema a ser resolvido (FACELI et al., 2011).

Além do volume de aplicações que se beneficia das técnicas de ML, outros fatores têm favorecido a expansão dessa área, como o desenvolvimento de algoritmos cada vez

mais eficazes e eficientes, e a elevada capacidade dos recursos computacionais atualmente disponíveis.

As tarefas de ML podem ser divididas em duas categorias: **preditivas** e **descritivas**. Para as preditivas, a meta é encontrar uma função a partir dos dados de treinamento que possa ser utilizada para prever um rótulo ou valor que caracterize um novo exemplo, com base nos valores de seus atributos de entrada. Os algoritmos utilizados nessa tarefa seguem o paradigma de **aprendizado supervisionado**. Esse termo vem da simulação da presença de um "supervisor externo", que conhece a saída desejada para cada exemplo. Com isso, o supervisor externo pode avaliar a capacidade da hipótese induzida de predizer o valor de saída para novos exemplos (FACELI et al., 2011).

Em tarefas descritivas, a meta consiste na exploração ou descrição de um conjunto de dados. Os algoritmos utilizados nessas tarefas não fazem uso do atributo de saída (variável de interesse). Tais algoritmos seguem o paradigma de **aprendizado não supervisionado**. Uma tarefa descritiva de agrupamento de dados, por exemplo, tem por meta encontrar grupos de objetos semelhantes no conjuntos de dados (FACELI et al., 2011).

Os métodos de ML consistem em algoritmos computacionais para relacionar todos ou alguns elementos de um conjunto de variáveis preditoras a um resultado. Para estimar o modelo, eles buscam de forma estocástica ou determinística o melhor ajuste. Esse processo de busca por um melhor ajuste do modelo aos dados difere entre os algoritmos existentes. No entanto, ao longo desse processo, cada algoritmo tenta equilibrar dois interesses: **viés** (*bias*) e **variância**. O viés é o ponto até o qual as previsões ajustadas pelo modelo correspondem aos valores verdadeiros. Um modelo com alto viés pode não ter a complexidade necessária para classificar as observações corretamente. Variância é a sensibilidade das previsões na presença de alguma alteração nos dados de entrada. Embora se busca reduzir tanto o viés quanto a variância, esses dois objetivos geralmente estão em conflito: o viés diminuído pode aumentar a variância e vice-versa. Por exemplo, pode-se criar um algoritmo que prevê corretamente todas as mortes em um conjunto de dados. No entanto, esse modelo pode estar vinculado demasiadamente aos detalhes individuais do conjunto de dados de treinamento (baixo viés), modelando fortemente o 'ruído estatístico'. O modelo teria um desempenho ruim quando aplicado em um novo conjunto de dados, apresentando portanto uma alta variância (FACELI et al., 2011).

Este capítulo irá apresentar métodos supervisionados utilizados para a modelagem preditiva de respostas quantitativas e categóricas de interesse para a área da saúde. No aprendizado supervisionado, um conjunto de dados dispõe de n observações, cada uma representada por um vetor de mensurações das m variáveis independentes (ou variáveis preditoras), bem como da mensuração correspondente da variável dependente (resposta de interesse ou *output*). Quando a resposta de interesse é uma variável quantitativa, trata-se de um modelo de *regressão*. Quando se tratar de uma variável qualitativa ou categórica, tem-se um modelo de *classificação*. A maior parte dos modelos de aprendizado supervisionado podem ser aplicados tanto para a predição de variáveis qualitativas quanto para variáveis quantitativas.

Qualquer que seja o tipo da variável de interesse, o algoritmo supervisionado modela uma função de maneira indutiva, isto é, a partir das informações contidas nos dados

utilizados para treinamento. O objetivo é predizer o valor da variável de interesse da maneira mais precisa possível, além de dotar o modelo de uma boa capacidade de generalização, isto é, ter um bom desempenho diante de dados que não foram utilizados para seu treinamento.

Ao longo do aprendizado, os parâmetros da função preditora são ajustados aos dados, de modo a minimizar o erro de predição e aumentar a capacidade de generalização. Além dos parâmetros automaticamente definidos pelos algoritmo, existem também os **hiperparâmetros**. Os hiperparâmetros de um algoritmo devem ser ajustados de modo que o algoritmo evite o sobreajuste (em inglês, *overfitting*), quando o modelo gerado se torna muito especializado no conjunto de treinamento, obtendo baixo desempenho quando confrontado com novos dados. O contrário também é necessário ser evitado, ou seja, quando o algoritmo não se ajusta adequadamente ao padrão, obtendo um baixo desempenho frente ao conjunto de treinamento (*underfitting*). Dessa forma, busca-se minimizar a probabilidade de erros de predição para dados futuros, reduzindo o viés indutivo, assim como reduzindo a variância do mesmo quando da previsão de novos casos.

O trabalho de Olson et al. (2017) buscou avaliar 13 algoritmos de classificação sob 165 conjuntos de dados na área de bioinformática de modo a prover um conjunto de recomendações para o uso de modelos de preditivos. Os modelos preditivos baseados em árvores e que fazem uso das técnicas de comitê de modelos (*ensamble*) apresentaram os melhores resultados. Os autores enfatizam que o processo de ajuste dos hiperparâmetros dos modelos preditivos pode contribuir significativamente para o desempenho dos modelos preditivos.

1.3.1. Processo de Aplicação das Técnicas de Aprendizado Supervisionado

Faz-se necessário apresentar o *workflow* utilizado para a aplicação das técnicas de ML, ilustrado na Figura 1.1. Inicialmente, o conjunto de dados é dividido em dois sub-conjuntos: treinamento e teste. O sub-conjunto de treinamento será utilizado para o ajuste do modelo preditivo aos dados, enquanto que o sub-conjunto de teste é utilizado para verificar a capacidade de generalização do modelo diante de novas observações. Ao longo do ajuste e avaliação do modelo preditivo, apenas o sub-conjunto de treinamento é apresentado ao modelo, ficando como último passo a aplicação do conjunto de teste, para a obtenção de uma estimativa mais precisa sobre o desempenho da predição futura do modelo.

Os dados originais em sua forma bruta normalmente precisam ser adequados aos modelos preditivos, de modo a otimizar o desempenho de tais modelos. A tais tarefas dá-se o nome de pré-processamento dos dados. Entre as principais tarefas de pré-processamento, estão incluídas (FACELI et al., 2011; SAKR et al., 2017; GÉRON, 2017; KUHN; JOHNSON, 2013):

- Transformação dos dados originais: boa parte dos algoritmos de predição apresenta melhor resultado quando as variáveis preditoras estão sob uma mesma escala. Para isto, técnicas de padronização (como a normalização z-score) podem ser aplicadas, em que os valores originais da variável são redimensionados para terem média igual a 0 e desvio-padrão igual a 1. A partir da análise da estatística de simetria dos dados,

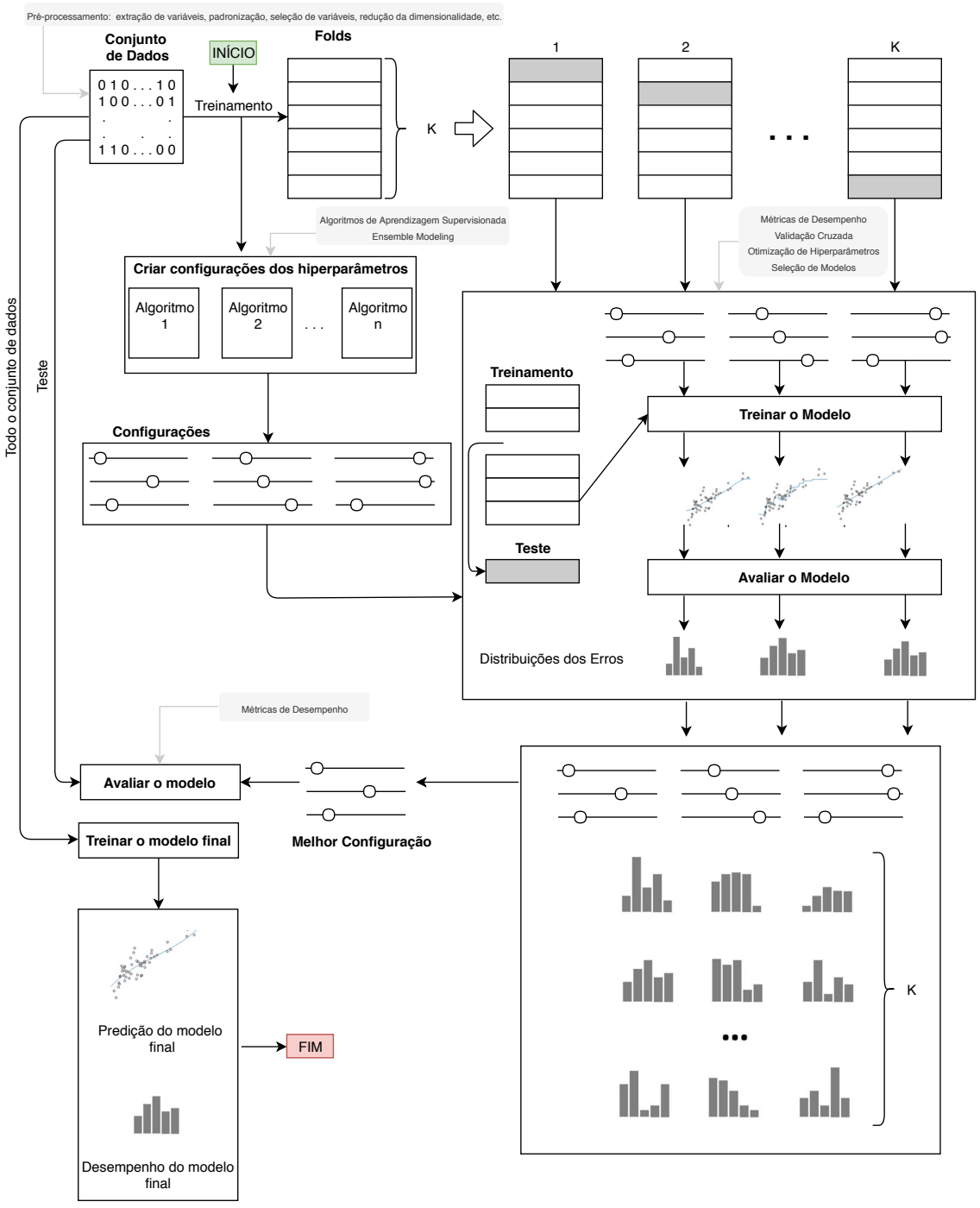


Figura 1.1. *Workflow* de aplicação das técnicas de machine learning para a análise preditiva. Adaptado de Borboudakis et al. (2017).

ou da razão entre o maior e o menor valor da variável preditora, pode ser feito o diagnóstico da necessidade de transformação dos dados;

- Remoção de preditores altamente correlacionados: normalmente os modelos preditivos apresentam desempenho inferior quando preditores altamente correlacionados estão medindo a mesma informação subjacente, além do problema do tempo desnecessariamente perdido para modelar variáveis semelhantes. Normalmente remove-se os preditores altamente correlacionados, de modo a garantir que as correlações de pares de variáveis preditoras estejam abaixo de um determinado limiar. Há também a possibilidade de utilização de técnicas para redução da dimensionalidade (como a análise de componentes principais ou o método dos mínimos quadrados parciais);

- Exclusão de preditores não informativos: variáveis preditoras que apresentam distribuição degenerada ou variância próxima a zero podem ser excluídas do conjunto de dados, resultando no aumento do desempenho do algoritmo;

- Tratamento de variáveis preditoras qualitativas: Se as variáveis qualitativas são ordinais, é importante convertê-las para números inteiros. Se tais variáveis forem nominais, é interessante transformá-las em um conjunto de variáveis indicadoras (denominadas *dummies*), as quais são representadas por valores binários. Por exemplo, para a variável qualitativa idr que representa o período do dia em que ocorreu um evento (manhã, tarde ou noite, por exemplo), três variáveis indicadoras serão criadas: $idr_{manha}, idr_{tarde}, idr_{noite}$, de modo que para cada instância apenas uma delas apresente o valor 1, enquanto as outras irá apresentar o valor 0, mapeando assim todos os valores possíveis da variável qualitativa;

- Tratamento para dados ausentes: boa parte dos conjuntos de dados apresentam variáveis que possuem valores ausentes. Como técnicas para tratamento dessas ausências, pode-se optar por remover a variável preditora, remover a observação (linha) do conjunto de dados que possui tal ausência, ou ainda fazer uso da imputação de valores interpolados, como a média, mediana ou o valor mais frequente de um preditor. Há também a possibilidade do uso de técnicas de imputação para determinar o valor a ser inserido como, por exemplo, por meio de modelos de regressão linear ou do algoritmo k-NN. Para o caso de variáveis categóricas dummificadas é também possível criar uma nova categoria específica para os valores faltantes.

Todas as tarefas de pré-processamento dos dados devem ser feitas no conjunto de treinamento. Os dados de teste não devem ser incluídos na consideração desses ajustes. Por exemplo, suponha que um valor ausente de uma variável preditora será substituído pela média dos valores dessa variável. Para o cálculo dessa média, apenas o conjunto de dados de treinamento deve ser levado em consideração. Quando tal procedimento for feito no conjunto de teste, ele será feito com base na média do conjunto de treinamento, não levando em conta as observações do conjunto de teste. Busca-se, dessa forma, evitar que o modelo tenha um conhecimento prévio sobre o conjunto de teste, o que "contaminaria" o modelo desenvolvido (KAUFMAN et al., 2012).

Normalmente após as tarefas de pré-processamento, inicia-se a fase de ajuste do modelo ao conjunto de dados de treinamento. Cada vez mais está se fazendo uso de

técnicas de reamostragem (pela limitação no número de observações disponíveis para treinamento/teste, por exemplo), assim como técnicas de comitê de modelos preditivos (*ensemble*), na qual um ou mais algoritmos de ML são utilizados para a realização da predição. Como cada algoritmo possui um conjunto de hiperparâmetros a serem ajustados, diversas configurações de hiperparâmetros são avaliadas, de modo a obter qual é a configuração que maximiza o desempenho de um modelo preditivo (GÉRON, 2017; KUHN; JOHNSON, 2013).

Dentre as técnicas de reamostragem, destaca-se a validação cruzada *k-fold*, na qual o conjunto de treinamento é dividido em k partes de tamanhos aproximadamente iguais, em que $k-1$ partes serão utilizadas para treinar o modelo preditivo, enquanto que a parte remanescente será utilizada para a validação do modelo e estimativa de seu desempenho. O processo se repete até que todas as partes tenham participado tanto do treinamento quanto da validação do modelo, resultando assim em k estimativas de desempenho que serão resumidas, normalmente, pelo cálculo da média e do erro padrão (GÉRON, 2017; KUHN; JOHNSON, 2013).

Para modelos de *ensamble*, métricas como *out-of-bag error* podem ser utilizadas para verificar o desempenho dos preditores. Uma vez que o desempenho de todos os preditores utilizados na fase de treinamento for conhecido, é possível determinar a melhor configuração dos hiperparâmetros, configurando-se assim o modelo final, o qual será testado com o conjunto de teste, possibilitando o cálculo de seu desempenho (GÉRON, 2017).

O último passo consiste no treinamento do modelo final sob todo o conjunto de dados, estabelecendo-se dessa forma um modelo preditivo a ser utilizado na predição da variável dependente a partir de novos dados (FACELI et al., 2011).

1.3.2. Avaliação do Desempenho

Há diversas métricas para a avaliação do desempenho de modelos preditivos com base no paradigma de aprendizado supervisionado. Tais métricas buscam mensurar o desempenho das predições decorrente do modelo ajustado, avaliando o quanto ele reproduz o valor observado para a resposta de interesse.

Para modelos de regressão, a métrica mais utilizada é a de erro quadrático médio (MSE, do inglês *Mean Squared Error*), de modo a quantificar o quanto o valor predito ($\hat{y}_i$) para a resposta de uma observação se aproxima de seu valor observado, y_i. O MSE é dado por:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{1}$$

A raiz quadrada desse valor possibilita obter o erro quadrático médio na mesma unidade da variável de interesse. Um modelo que apresenta respostas preditas muito próximas das observadas irá apresentar um baixo MSE. A comparação entre múltiplos modelos, sob as mais variadas configurações de hiperparâmetros também podem ser feitas por meio da métrica de MSE.

Para modelos de predição por classificação, normalmente se realiza uma tabulação cruzada das classes observadas e preditas em uma *matriz de confusão*, apresentada na Figura 1.2.

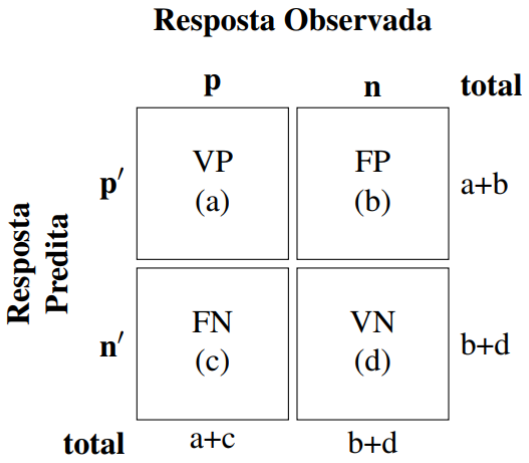


Figura 1.2. Exemplo de Matriz de Confusão para Algoritmos de Classificação.

A partir dessa matriz, diversas métricas podem ser mensuradas, apresentadas na Tabela 1.1.

Tabela 1.1. Métricas de desempenho derivadas da matriz de confusão.

Métrica (nomenclatura)	Cálculo	Descrição
Acurácia; Exatidão; Precisão	$ACC = \frac{VP+VN}{VP+VN+FP+FN}$	Proporção de observações classificadas corretamente
Taxa de falsos positivos; Fall-out	$FPR = \frac{FP}{FP+VN}$	Proporção de observações classificadas incorretamente como verdadeiras
Sensibilidade; taxa de verdadeiro positivos; recall	$TPR = \frac{VP}{VP+FN}$	Proporção de verdadeiros positivos corretamente identificados
Especificidade; taxa de verdadeiro negativos	$TNR = \frac{VN}{VN+FP}$	Proporção de verdadeiros negativos corretamente identificados
F-score (F1)	$F1 = \frac{2*TP}{2*TP+FP+FN}$	Média harmônica entre a acurácia e o recall de um classificador

Uma técnica adotada para resumir o desempenho do modelo em diferentes situações é a construção da curva *Receiver Operating Characteristic* (ROC). Essa curva avalia diferentes pontos de limiar para discriminação do que é considerado valor positivo:

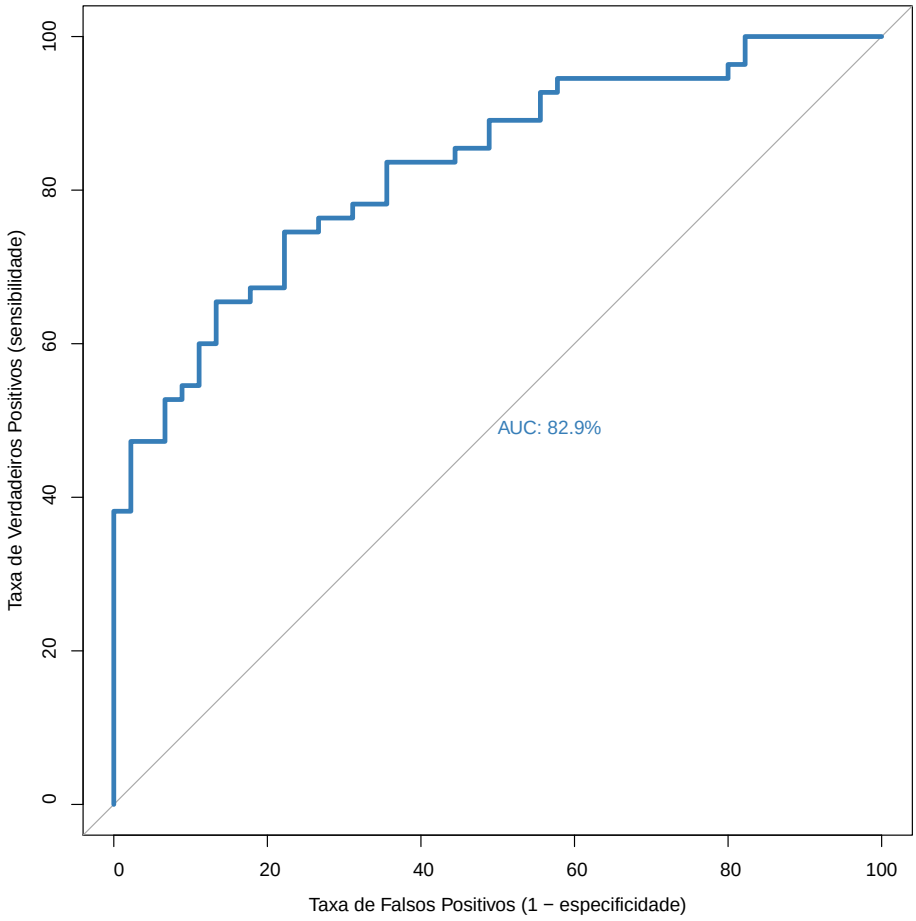


Figura 1.3. Exemplo de uma curva ROC.

enquanto o eixo vertical indica a sensibilidade (taxa de verdadeiros positivos), o eixo horizontal indica a taxa de falsos positivos (1 - sensibilidade), onde cada ponto no espaço representa os respectivos valores obtidos de uma matriz de confusão (GÉRON, 2017).

A Figura 1.3 apresenta um exemplo da curva ROC para um determinado algoritmo de classificação. A diagonal que corta o espaço ROC ao meio é conhecida como a linha de zero discriminação, onde permanecem resultados de decisões aleatórias. Um classificador satisfatório deve, no mínimo, apresentar uma curva acima dessa linha. O cálculo da área sob a curva obtida, conhecido como AUC (*area under curve*) é uma métrica pode ser utilizada para avaliar a probabilidade que um modelo possui de discriminar o exemplo positivo de um par escolhido aleatoriamente. Quanto mais próximo de 1 for o valor da AUC, melhor será o algoritmo de classificação.

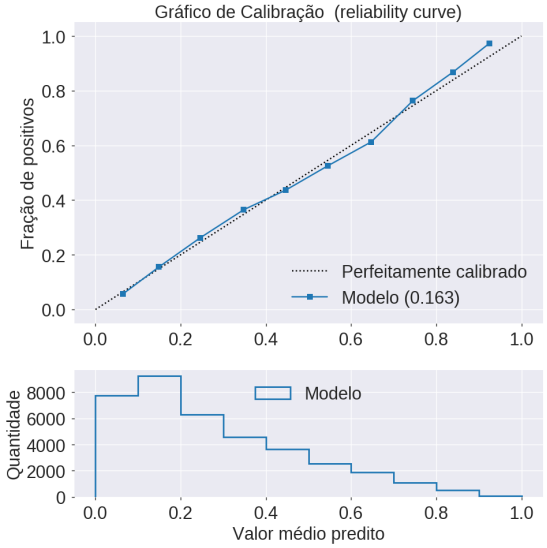


Figura 1.4. Exemplo de curva de calibração para um modelo preditivo.

Ao realizar a classificação, muitas vezes é necessário prever não apenas a classe, mas também a probabilidade associada. Dizemos que um algoritmo apresenta boa calibração quando pode ser utilizado como um estimador de probabilidade. Uma maneira de obter o valor da calibração de um modelo é pela mensuração do erro dado por $\hat{P}(+|\mathbf{x}_i)$ e $P(+|\mathbf{x}_i)$, onde $\hat{P}$ e P representam a probabilidade de ocorrência do evento predito pelo modelo e a classe verdadeira do exemplo, respectivamente. Dessa forma, o erro pode ser calculado como a diferença quadrática entre $\hat{P}(+|\mathbf{x}_i)$ e a classe verdadeira do exemplo (y_i), conforme a equação abaixo, conhecida como *Brier Score* (bs):

$$bs(y,\hat{P}) = \frac{1}{n}\sum_{1}^{n}(\hat{P}(+|\mathbf{x}_i) - y_i)^2 \tag{2}$$

Visualmente, faz-se uso do gráfico da curva de calibração do modelo. A Figura 1.4 apresenta um exemplo de curva de calibração. O diagrama agrupa as predições em compartimentos de acordo com a probabilidade obtida do modelo (eixo horizontal). A frequência com que o evento foi observado para esse subgrupo de predições é então plotada no eixo vertical. Para uma perfeita calibração, a probabilidade do evento e a frequência de ocorrência devem ser iguais, e os pontos plotados devem estar na diagonal. Assim, por exemplo, quando o modelo declara que há uma probabilidade de 25% de ocorrência de um evento, estará calibrado se tal ocorrência se concretizar em 25% das observações que tiveram tal probabilidade.

1.4. Algoritmos de ML

A seguir são apresentados os conceitos básicos dos seguintes algoritmos de aprendizado supervisionado: Regressão Linear, Regressão Logística, K-Vizinhos mais Próximos, Árvores de Decisão, Random Forest, Gradient Boosted Trees e Redes Neurais Artificiais.

1.4.1. Regressão Linear

Os modelos preditivos de regressão linear visam verificar se existe uma relação entre duas ou mais variáveis. O problema, portanto, se encontra na predição de valores contínuos para a variável de interesse (MAGALHÃES; LIMA, 2002).

Dado um conjunto com n observações e $(m+1)$ variáveis, cujos valores são representados por $(y_i, x_{i1}, x_{i2}, \ldots, x_{im})$, $i = 1, 2, \ldots, n$, o modelo de regressão linear estabelece uma relação entre a variável resposta ou de interesse ($\hat{y}$) e as variáveis explicativas ou preditoras $(x_{i1}, x_{i2}, \ldots, x_{im})$, da seguinte forma:

$$\hat{y}_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_m x_{mi} + e_i \tag{3}$$

onde $\beta_0, \ldots, \beta_m$ são chamados coeficientes e representam o aumento/decréscimo médio esperado para quaisquer mudanças na respectiva variável preditora, independentemente das restantes preditoras. e_i é o o erro ou parte não capturada pelo modelo, independente e identicamente distribuído.

A estimação dos coeficientes $\beta_0, \ldots, \beta_m$ é realizada de forma a minimizar a soma dos quadrados das diferenças entre cada ponto observado (y) e o seu valor estimado ($\hat{y}$) pela reta. Esta diferença é conhecida como resíduo. Em outras palavras, o que pretende-se é predizer um valor para y, denominado $\hat{y}$, minimizando o erro quadrático médio (MSE), dado por:

$$\begin{aligned} MSE(\hat{y}, y) &= \sum_{i=1}^{n} (e_i)^2 \\ &= \sum_{i=1}^{n} \left(\hat{y}(x_i) - y(x_i)\right)^2 \\ &= \sum_{i=1}^{n} [y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_m x_{im})]^2 \end{aligned} \tag{4}$$

onde n é o número de observações de treinamento, utilizadas na elaboração do modelo.

A Figura 1.5 apresenta um exemplo de ajuste do modelo de regressão linear simples, ou seja, com uma variável de interesse e uma variável preditora, para um determinado conjunto de dados, cuja raiz do erro médio quadrático (RMSE, do inglês *Root Mean Square Error*) é de 26,84. O RMSE é um medida utilizada para verificar a qualidade do ajuste, sendo dada na mesma escala da varável de interesse (y). Quanto menor o valor do RMSE, melhor a qualidade do ajuste.

1.4.2. Regressão Logística

O modelo de regressão logística possui sua representação gráfica em formato de "S", sendo esta conhecida como curva logística. Este modelo estabelece uma relação entre a

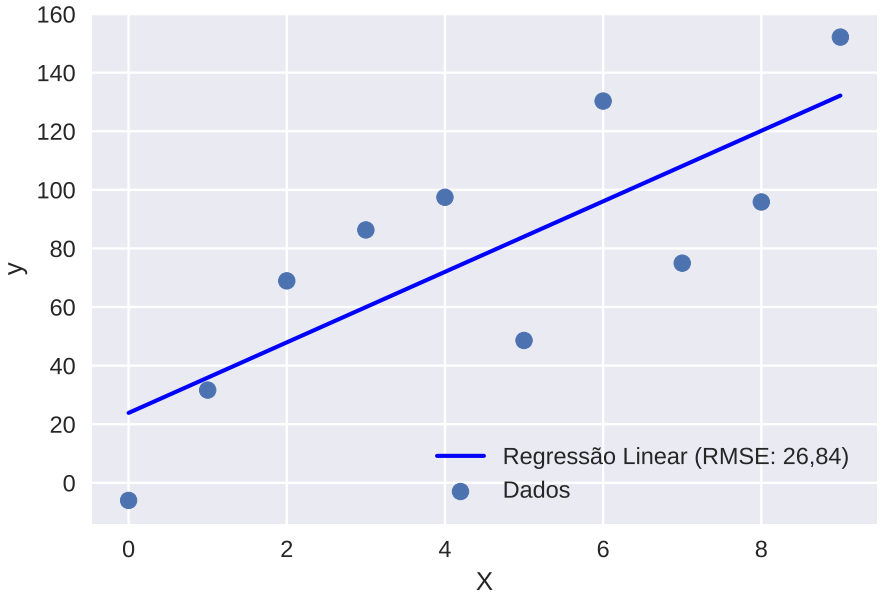


Figura 1.5. Exemplo de regressão linear simples.

probabilidade de ocorrência da variável categórica de interesse e as variáveis preditoras. A Figura 1.6 apresenta um exemplo da curva logística.

Uma das expressões mais simples da função que descreve a curva logística é dada por:

$$f(x) = \frac{1}{1+e^{(-x)}} \tag{5}$$

onde e é conhecido como número de Euler (aproximadamente 2,72).

Em Machine Learning, a regressão logística é utilizada para tarefas de classificação, ou seja, em que a variável de interesse é categórica. Nesse caso, a variável de interesse assume valores 0 ou 1, onde 1 é geralmente utilizado para indicar a ocorrência do evento de interesse. Uma nomenclatura bastante utilizada para os valores 0 e 1 é classe negativa e classe positiva, respectivamente.

De forma simplificada, o modelo logístico pode ser definido da seguinte maneira:

$$\hat{y} = f(Z) = P(y = 1|Z) = \frac{1}{1+e^{(-Z)}} \tag{6}$$

onde $P(y = 1|Z)$ é a probabilidade de ocorrência da classe positiva dado Z. Sendo Z:

$$Z = ln(\frac{p}{1-p}) = \alpha + \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_k X_{ik} \tag{7}$$

onde p indica a probabilidade de ocorrência do evento de interesse, X é o conjunto das variáveis preditoras e α e β são parâmetros do modelo, os quais estão estimados pela

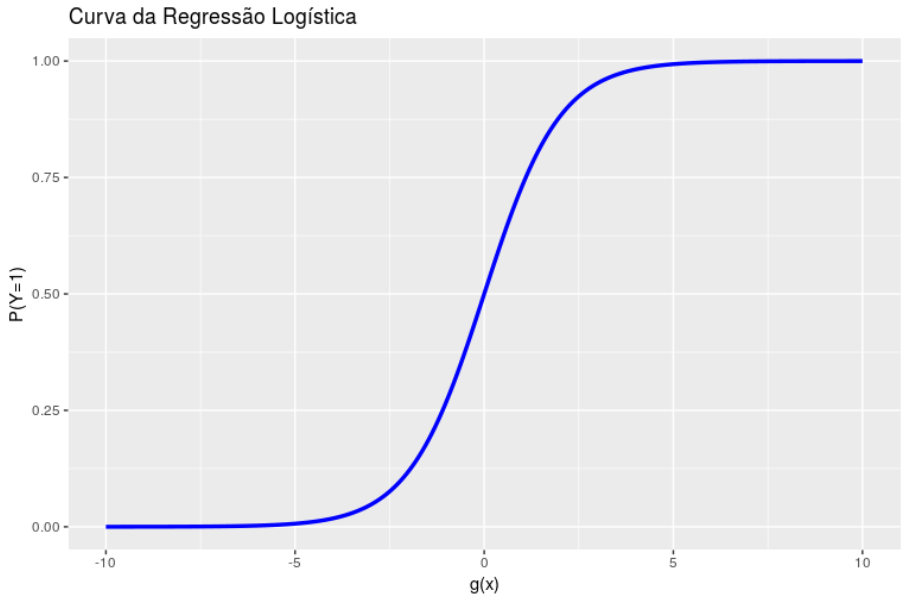


Figura 1.6. Exemplo de regressão logística.

máxima verossimilhança (FAVERO et al., 2009), encontrando uma combinação de coeficientes que maximiza a probabilidade de ocorrência do evento de interesse.

Reescrevendo o modelo, obtém-se:

$$\hat{y} = f(Z) = \frac{1}{1+e^{(-Z)}} = Prob(y = 1 \mid X_1, X_2, \ldots, X_k) = \frac{1}{1+e^{-(\alpha - \sum B_i X_i)}} \tag{8}$$

Dessa forma, um modelo de regressão logística utilizado para a classificação binária de uma variável de interesse irá predizer a probabilidade de pertencer à classe positiva. Tem-se, portanto, que $\hat{y}$ é dado por:

$$\hat{y} = \begin{cases} 0, & se\ Prob(y = 1 \mid X) < 0,5 \\ 1, & se\ Prob(y = 1 \mid X) \geq 0,5 \end{cases} \tag{9}$$

onde X é o vetor das variáveis preditoras.

1.4.3. K-vizinhos mais próximos

O algoritmo de "K-vizinhos mais próximos", conhecido por k-NN (*k-nearest neighbors*), é um algoritmo de classificação supervisionado não-paramétrico e baseado em instâncias. Este classificador é ajustado aos dados de treinamento (x, y) de modo a obter uma função $h : X \rightarrow Y$ de modo que, quando diante de uma nova observação x, prediz com uma acurácia aceitável em relação a y.

O fato de ser não-paramétrico indica que não há premissas por parte do algoritmo em relação à distribuição dos dados de treinamento. Por ser baseado em instâncias, o k-NN armazena os exemplos de treinamento para serem utilizados no momento da predição.

Quando diante de uma nova instância, o modelo irá utilizar as observações de treinamento para classificá-la.

O algoritmo k-NN assume que todas as observações correspondem a pontos em um espaço n-dimensional. Os "vizinhos mais próximos" de uma instância são mensurados por métricas de distância, tais como: distância Euclidiana, distância de Hamming, distância de Manhattan, etc. A título de exemplo, será apresentada a distância Euclidiana, definida por:

$$d(x,x') = \sqrt{(x_1 - x'_1)^2 + (x_2 - x'_2)^2 + \cdots + (x_n - x'_n)^2} \tag{10}$$

Dado um número inteiro positivo ímpar k, e uma instância a ser classificada x, sob uma métrica de distância d, o classificador k-NN executa as seguintes ações:

- Para todo o conjunto de observações de treinamento, calcula-se a distância d em relação a x;
- As k observações de treinamento mais próximas de x são agrupadas em um conjunto A;
- A probabilidade condicional para cada classe é definida por:

$$P(y = j|X = x) = \frac{1}{K} \sum_{i \in A} I(y^{(i)} = j) \tag{11}$$

  onde $I(x)$ é uma função que retorna 1 se seu argumento for verdadeiro e 0, caso contrário.

- Por último, a instância x é então classificada como a classe que apresentou maior probabilidade condicional.

O hiperparâmetro necessário para ajuste do modelo é o valor de k. Quando k apresenta um valor pequeno, o modelo não consegue observar a distribuição geral das classes no conjuntos de dados, o que leva para uma situação de baixo viés, porém alta variância, tendo então o modelo apresentando sobreajuste.

Já quando k é um valor muito alto, mais elementos irão compor o mecanismo de votação (vizinho mais próximo), de modo que o modelo poderá ficar sujeito a *outliers*. Altos valores de k levam a um maior viés, porém uma menor variância quando diante de novas observações. O trabalho de Jiang et al. (2007) sugere o uso da técnica de validação cruzada para estimar o valor adequado de k.

1.4.4. Árvores de Decisão

Os modelos preditivos baseados em árvores são comumente utilizados para tarefas de classificação, embora também possam ser utilizados para tarefas de regressão. Considerado um dos mais populares algoritmos de predição, o algoritmo de árvore de decisão proporciona uma grande facilidade de interpretação.

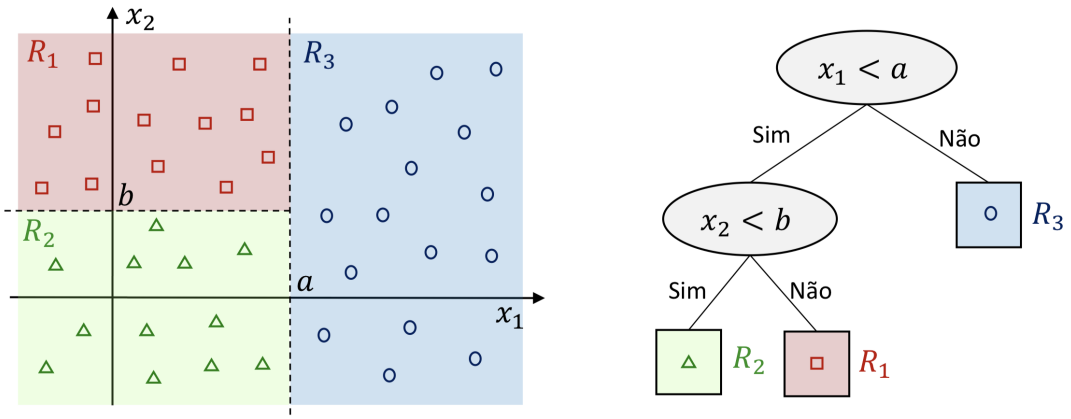


Figura 1.7. Árvore de decisão aplicada a uma tarefa de classificação. Na imagem da direita tem-se a árvore de decisão, ao passo em que a imagem da esquerda ilustra como a árvore está discriminando as classes a serem preditas (MAYRINK et al., 2016).

Uma árvore de decisão é uma estrutura hierárquica na qual cada nó interno representa uma decisão em um dos atributos do conjunto de dados, e cada ramo representa uma tomada de decisão. Nessa árvore, as folhas representam o rótulo da classe predita. A Figura 1.7 apresenta um exemplo de árvore de decisão, onde as variáveis x_1 e x_2 representam os atributos do conjunto de dados (variáveis preditoras) e os símbolos □ , △ e ○ indicam os rótulos de classe observados na amostra.

É possível observar que a partir do aprendizado efetuado nos atributos preditores, a árvore de decisão constrói regiões de discriminação das classes do conjunto de dados.

Uma vez construída a árvore, dado um conjunto de atributos preditores para os quais se deseja conhecer o valor da classe, tais atributos são testados por meio da árvore de decisão, onde um caminho é traçado a partir da raiz até um nó folha, o qual irá representar a classe predita.

O processo de dividir o conjunto de dados D em partições, por meio de um atributo que virá a ser um nó da árvore, busca gerar subdivisões de D em que, idealmente, cada partição tenha mais dados pertencentes a uma mesma classe. Na busca por otimizar esse resultado, o critério de seleção do atributo que irá particionar os dados será pelo o atributo que resulta em um maior número de partições que se aproximam do objetivo proposto. Dizemos então que este atributo tem uma maior capacidade de resumir D.

Há na literatura alguns algoritmos propostos para a construção da árvore, tais como ID3 (*Iterative Dichotomiser*), C4.5 e CART (*Classification and Regression Tree*). O algoritmo ID3 elenca os nós da árvore por meio da métrica do ganho de informação. Já o algoritmo CART faz normalmente uso do índice de Gini (KUHN; JOHNSON, 2013).

As árvores de decisão têm alta tendência ao sobreajuste, uma vez que para n observações de treinamento, a árvore pode construir n caminhos diferentes. Isto é, a árvore de decisão mapeou cada instância do treinamento como uma folha distinta. Para evitar situações como essa podem utilizados métodos de poda (*prunning*), o qual é controlado

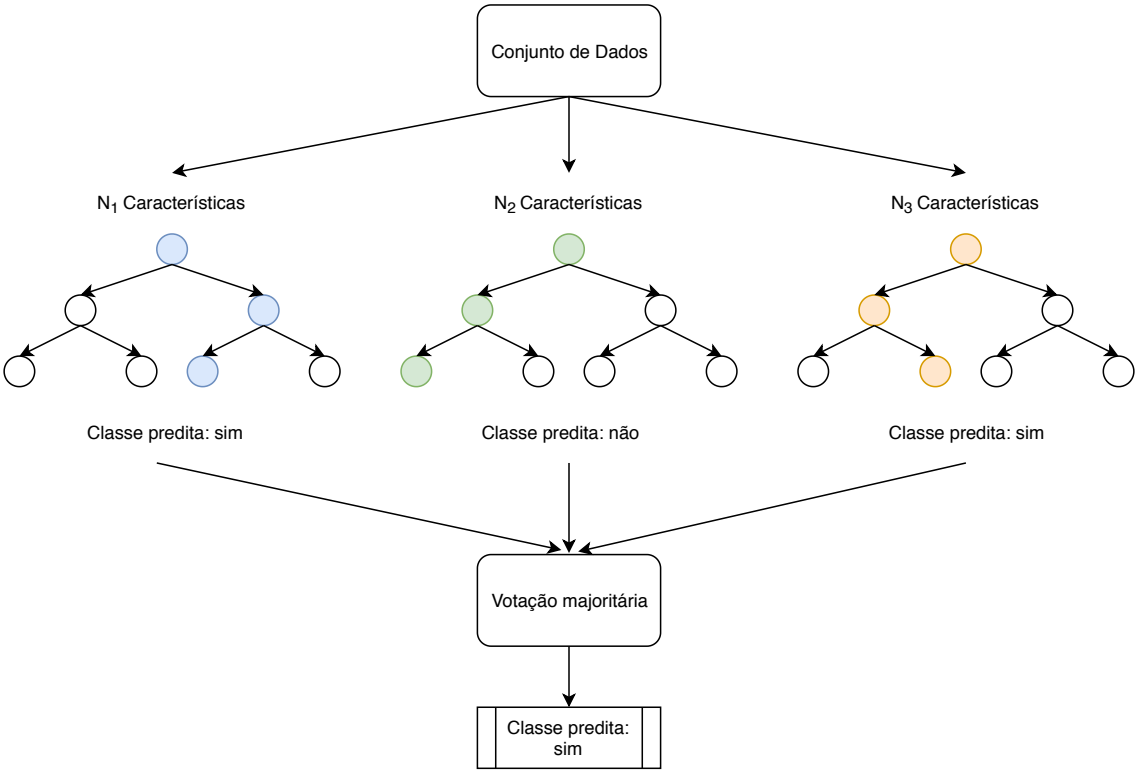


Figura 1.8. Ilustração do algoritmo Random Forest.

como um hiperparâmetro do modelo.

1.4.5. Random Forest e Gradient Boosted Trees

Os algoritmos Random Forest e Gradient Boosted Trees são exemplos de comitês (*ensembles*) de algoritmos de machine learning. Por meio desses algoritmos, diversas árvores de decisão são criadas. Trabalhos recentes demonstram que o uso destes algoritmos tem proporcionado resultados superiores na capacidade de predição quando comparado com outros algoritmos de machine learning (GÉRON, 2017; RASCHKA; MIRJALILI, 2017).

O algoritmo de Random Forest é utilizado para tarefas de regressão e classificação. Trata-se de um ensemble de árvores de decisão que são construídas com base na estratégia de bagging (*boostrap aggregation*). Neste algoritmo são geradas n árvores de decisão. Na ocasião da predição para uma determinada instância, as n árvores fazem a predição e, para os problemas de classificação, a classe majoritariamente predita é considerada a classe da instância.

Ao longo do processo de treinamento, de modo a construir as n árvores de decisão, o algoritmo seleciona, para cada *split* a ser feito em uma árvore, um conjunto p de variáveis preditoras (p é normalmente dado como $\sqrt{m}$, onde m é o número de variáveis preditoras), que será utilizado para a construção da árvore. Cada árvore é treinada com um conjunto de observações de treinamento obtidas por meio de uma seleção aleatória com repetição (técnica conhecida como Booststrap).

A Figura 1.8 ilustra a construção de três árvores de decisão para um problema de

classificação, construídas pelo algoritmo Random Forest. A classe predita da instância em questão é a classe majoritária predita pelo comitê de árvores. O método de bagging, portanto, baseia-se em criar preditores em amostras bootstrap dos dados, e depois agregá-los, ou combiná-los, para formar um preditor, o qual é esperado apresentar um resultado superior quando comparado a um algoritmo tradicional.

A estratégia de bagging proporciona que o algoritmo de Random Forest seja menos suscetível a sobreajuste, assim como viabiliza a redução do viés e da variância do modelo que está sendo construído, uma vez que as árvores de decisão são construídas fazendo-se uso de diferentes subconjuntos das variáveis preditoras.

Com o mesmo intuito de maximizar a acurácia da predição das árvores de decisão, o algoritmo Gradient Boosted Trees (FRIEDMAN, 2001) faz uso da técnica de Boosting.

De acordo com Chambers e Dinsmore (2014), Gradient Boosted Trees é um algoritmo que produz diferentes modelos sequenciais de árvores de decisão, cujos resultados são adicionados para que modelo final seja uma combinação dos anteriores, apresentando uma capacidade de predição muito superior à dos modelos individuais que foram gerados.

A cada iteração do algoritmo, uma árvore é gerada de modo a aprender e minimizar os erros da árvore anterior. A Figura 1.9 ilustra o princípio de funcionamento do algoritmo Gradient Boosted Trees como uma sequência de tacadas de golfe. A primeira tacada (F_0) representa a primeira árvore gerada. Em função da distância da posição bola ($\hat{y}$) até buraco (y), constrói-se uma nova árvore (F_1) com o objetivo de predizer Δ_1, para que $\hat{y} = F_0 + \Delta_1$. Após algumas iterações, um conjunto de árvores foi construído na busca por estimar os erros das árvores anteriores. Dessa forma, para o exemplo em questão, a predição final será dada por: $\hat{y} = F_0 + \Delta_1 + \Delta_2 + \Delta_3 + \Delta_4$, onde F_0 é a árvore inicial, e $\Delta_1, \Delta_2, \Delta_3$ e Δ_4 são valores preditos por árvores geradas para aprender e minimizar os erros das árvores respectivamente anteriores.

1.4.6. Redes Neurais Artificiais

Modelos computacionais de redes neurais artificiais (RNA) são inspirados na estrutura neural de organismos inteligentes, os quais adquirem conhecimento por meio da experiência. As redes possuem unidades de processamento denominadas neurônios artificiais, os quais, por meio de uma rede de interligação, processam os sinais de entrada, na busca pela predição da variável de interesse, que é possível por meio do encaminhamento do sinal de entrada pela rede, simulando um processo de sinapse.

Para Haykin (1999) uma RNA é um sistema massivamente paralelo e distribuído, composto por unidades de processamento simples que possuem a capacidade de armazenar e utilizar conhecimento. As RNAs são uma alternativa para a solução de diversos problemas complexos, uma vez que a representação interna e o paralelismo inerente à sua arquitetura possibilitam um desempenho superior aos modelos convencionais para alguns tipos de problemas (BRAGA; FERREIRA; LUDERMIR, 2007).

Nas RNAs o processamento ocorre nos neurônios artificiais, que estão interconectados. O sinal de entrada é transmitido por meio das conexões entre os neurônios. Para cada conexão, de modo de representar a eficiência da sinapse, um peso associado é armazenado. Dessa forma, o processo de aprendizado de uma RNA se dá por meio

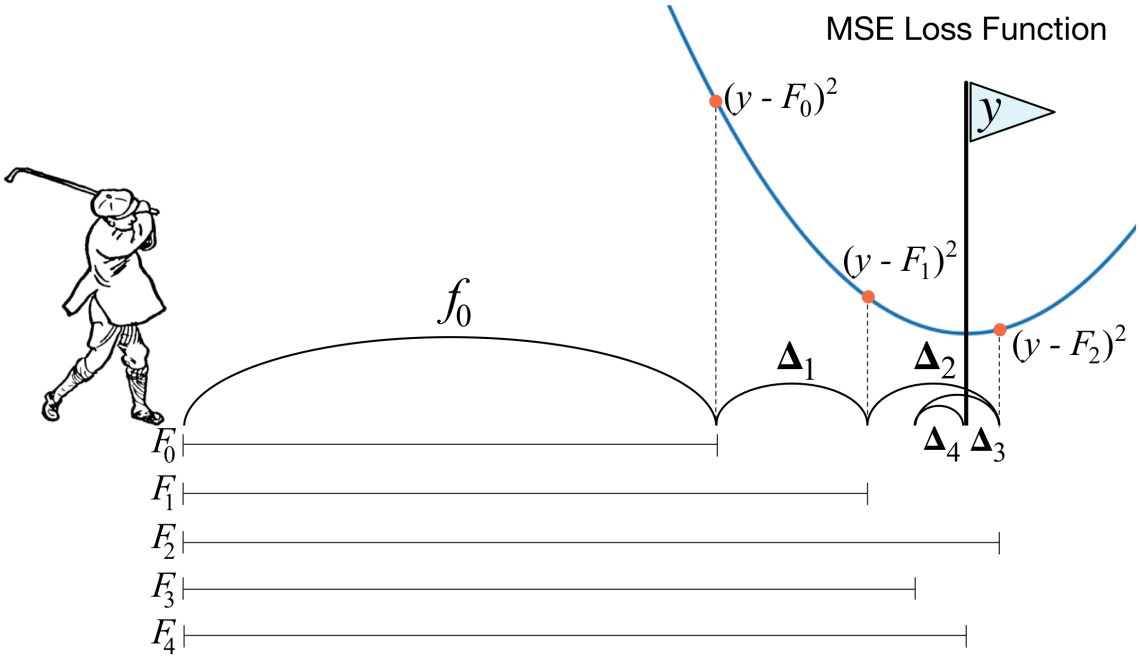


Figura 1.9. Princípio de funcionamento do algoritmo Gradient Boosted Trees (PARR; HOWARD, 2018).

do ajuste dos pesos das conexões entre os neurônios. O treinamento de uma rede neural artificial, portanto, consiste em identificar um conjunto apropriado de pesos de forma que a rede neural se comporte como o desejado, maximizando a sua capacidade de predição e generalização.

Ao longo da construção de modelos de RNAs, diversos pontos devem ser estabelecidos, como (i) a determinação do conjunto de neurônios artificiais, (ii) a escolha da arquitetura da rede neural, isto é, o padrão de conectividade entre os neurônios e as suas respectivas funções de ativação, e (iii) a escolha do método de determinação dos parâmetros da rede, conhecido como algoritmo de aprendizagem.

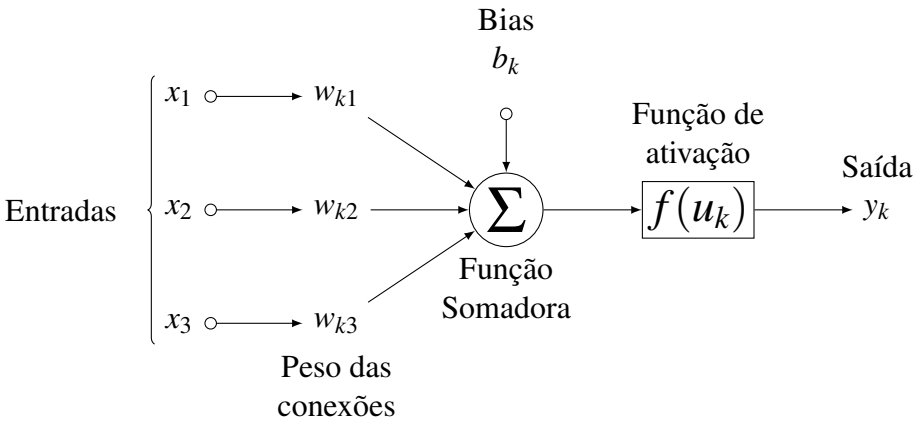


Figura 1.10. Arquitetura de um neurônio artificial.

A Figura 1.10 ilustra a arquitetura de um neurônio artificial. Nela, um neurônio artificial k recebe uma entrada de dados $x_1, x_2, \ldots, x_m$. As conexões recebem valores referentes aos pesos sinápticos, de forma que w_{ki} é o peso sináptico da conexão entre k a entrada x_i.

O neurônio realiza um somatório de todos os sinais sinápticos ponderados pelos respectivos pesos de suas conexões com k. O neurônio também conta um com valor limiar b_k que tem o papel de aumentar ou diminuir a influência do valor das entradas para a ativação do neurônio.

Uma vez calculado o produto interno do vetor de entradas x pelo vetor de pesos w_k, e considerando-se o limiar b_k, tal resultado (u_k) é submetido a uma função de ativação $f(u_k)$, a qual limita a saída do neurônio enquanto introduz não linearidade no modelo. Assim, o valor de saída do neurônio k, denominado y_k, é dado por:

$$y_k = f(u_k) = f\left(\sum_{j=1}^{m} w_{kj} x_j + b_k\right) \tag{12}$$

A notação pode ser simplificada, considerando-se o bias simplesmente como um sinal de entrada de valor $x_0 = 1$ e com peso associado $w_{k0} = b_k$:

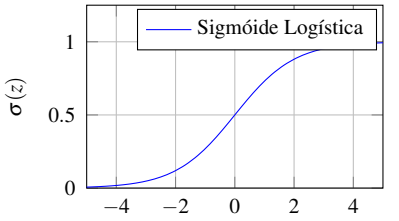
$$y_k = f(u_k) = f\left(\sum_{j=0}^{m} w_{kj} x_j\right) = f(w^T x) \tag{13}$$

A escolha adequada da função de ativação $f(u_k)$ e de suas propriedades contribuem para melhorar o processo de convergência da rede durante seu treinamento. Existem diversas funções matemáticas que são utilizadas como função de ativação. As funções de ativação mais comumente usadas são: função sigmóide logística e a função tangente hiperbólica. Com o advento das redes neurais profundas, a função de ativação retificadora linear (ReLU) também tem sido bastante utilizada. Estas funções estão ilustradas na Figura 1.11.

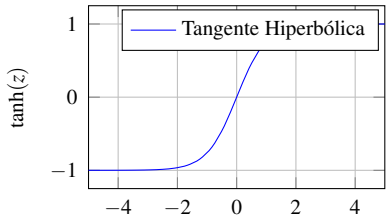
A RNA passa a ser conhecida como um algoritmo de deep learning (aprendizado profundo) quando sua arquitetura é formada por múltiplas camadas. As camadas internas de uma rede neural são denominadas camadas ocultas. A Figura 1.12 ilustra um exemplo de uma rede multicamada *feedforward*, na qual cada neurônio artificial recebe apenas entradas de unidades situadas na camada imediatamente anterior. Devido ao conjunto de conexões sinápticas e da riqueza de interações neurais, as camadas ocultas são capazes de extrair características complexas dos dados (HAYKIN, 1999).

1.5. Implementando Modelos de Machine Learning em Python

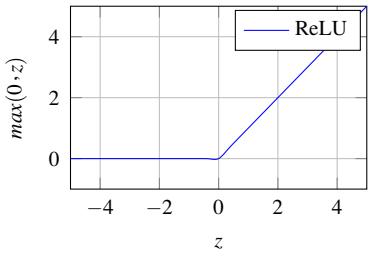
É crescente o aumento da popularidade da linguagem Python em aplicações científicas. Boa parte deste aumento se dá pela diversidade de bibliotecas científicas como Numpy e Scipy. Na área de Machine Learning, a linguagem Python apresenta diversas bibliotecas que permitem a execução de modelos de ML, com destaque para a biblioteca Scikit-Learn (BUITINCK et al., 2013).



[Sigmóide Logística]



[Tangente Hiperbólica]



[Retificadora Linear]

Figura 1.11. Exemplos de Funções de ativação para Redes Neurais Artificiais.

A proposta da biblioteca Scikit-Learn consiste em fornecer um conjunto de funcionalidades padronizadas, de modo a permitir que especialistas das mais diversas áreas possam construir modelos de Machine Learning (BUITINCK et al., 2013).

Além de bem documentados, os modelos de ML implementados na biblioteca Scikit-Learn são padronizados quando ao input de dados e aos métodos disponíveis para a sua execução. Todos os modelos disponíveis na biblioteca aceitam entrada de dados na forma de arrays bidimensionais (observações x características).

Há três componentes básicos: Transformer, Estimator e Predictor. Estes são apresentados a seguir, em conjunto com a Figura 1.13, a qual apresenta como estes componentes estão relacionados entre si e como são aplicados aos conjuntos de treinamento e de teste na busca pelo ajuste de modelos de Machine Learning. Pressupõe-se que o conjunto de dados foi dividido em duas partes: treinamento e teste, para o qual a biblioteca Scikit-Learn dispõe do método `train_test_split`, que pode ser utilizado como segue:

```
1 from sklearn.model_selection import train_test_split
2
3 #70% treino, 30% teste
4 X_train, X_test, y_train, y_test = train_test_split(X, y,
    ↪ train_size=0.70, random_state=42)
```

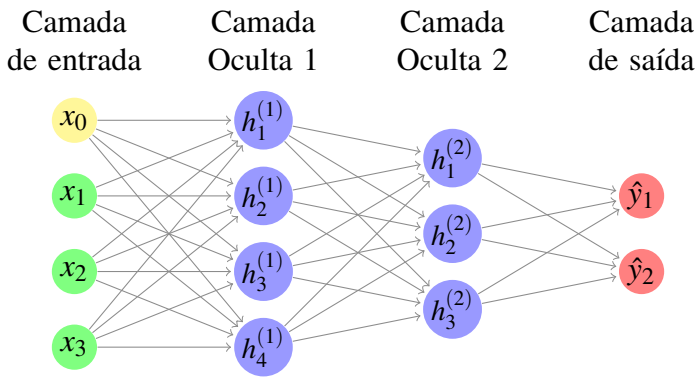



Figura 1.12. Exemplo de rede neural multicamadas.

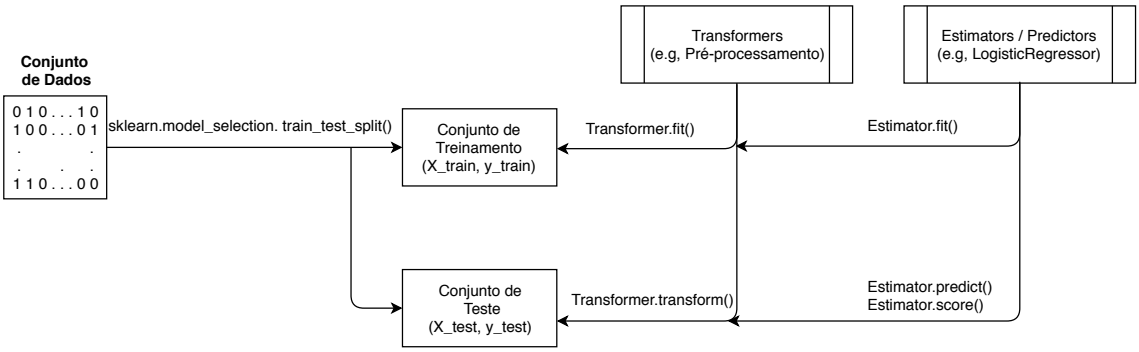


Figura 1.13. Componentes básicos da biblioteca Scikit-Learn e seu uso na construção de modelos de Machine Learning.

A linha ❹ executa a divisão do conjunto de dados em duas partes (treino, teste), de modo que o teste tenha 30% dos dados. Para garantir a reprodutibilidade dos resultados, a semente de aleatoriamente (random_state) precisa ser definida com algum valor (no caso o número que nos dá o significado da vida do universo e tudo mais, 42).

- **Transformer**: De modo a contemplar as necessidades de modificação e/ou filtro de dados antes de introduzi-los em um algoritmo de ML, a biblioteca Scikit-Learn disponibiliza uma interface denominada Transformer. Atividades de pré-processamento, seleção de características, extração de características e algoritmos de redução da dimensão são exemplos de transformadores disponibilizados pela biblioteca. Por exemplo, para a necessidade de padronizar os dados (média = 0 e desvio-padrão = 1), pode-se fazer uso do Transformer `StandardScaler`, por meio do método `fit`:

```
1 from sklearn.preprocessing import StandardScaler
2
3 scaler = StandardScaler()
4 scaler.fit(X_train)
5 X_train = scaler.transform(X_train)
```


Dessa forma, o objeto `scaler` armazena a média e o desvio-padrão do conjunto de treinamento, obtidos com a invocação do método `fit()` (❹), para que quando da invocação do método `transform` (❺) estes valores sejam utilizados. Esta característica é muito importante para a padronização do conjunto de teste, o qual deve ser padronizado fazendo-se uso da média e desvio-padrão que foram aprendidos a partir do conjunto de treinamento. A transformação do conjunto de teste se daria pelo seguinte código:

```
X_test = scaler.transform(X_test)
```

- **Estimator**: trata-se de uma interface para a implementação de algoritmos de ML. Todo algoritmo de ML disponibilizado na biblioteca Scikit-Learn implementa esta interface. Não apenas os algoritmos de ML são implementações da interface Estimator, mas também outras funcionalidades (como seleção de características, imputação de dados, etc.).

  A interface Estimator disponibiliza o método `fit` para que um modelo seja ajustado a partir de um conjunto de dados. Na inicialização de um algoritmo disponibilizado pela biblioteca é possível definir o conjunto de hiperparâmetros que será utilizado no momento da execução do método `fit`.

  A biblioteca também conta com um conjunto de valores padronizados para cada possível hiperparâmetro. Ao longo da execução do método `fit`, o algoritmo, fazendo-se uso dos dados de treinamento, busca determinar os parâmetros específicos do modelo que está sendo ajustado. Por exemplo, ao longo de uma regressão linear, o algoritmo irá determinar os valores dos coeficientes da regressão a partir dos dados fornecidos ao método `fit`. Por convenção, todo parâmetro aprendido pelo modelo recebe um subtraço como sufixo. No caso da regressão, os coeficientes determinados pelo modelo são armazenados na variável `coef_`.

  A seguir é apresentado um trecho de código-fonte para a execução de uma regressão logística. Nesse caso, o estimator é `LogisticRegression`, cujo hiperparâmetro `penalty` recebe o valor `l1`, reproduzindo assim uma regularização l_1. Outros hiperparâmetros não especificados (como o parâmetro de regularização C) irão receber o valor padrão. Uma vez que o método `fit` é invocado, o modelo será construído a partir dos dados de treinamento fornecidos.

```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(penalty="l1")
clf.fit(X_train, y_train)
```

Para conhecer todos os hiperparâmetros disponíveis para os ajustes de um modelo preditivo, pode-se fazer uso do método `get_params()`, conforme a seguir:


```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(penalty="l1")
#obtendo hiperparametros disponiveis:
clf.get_params()
>>{'C': 1.0,
 'class_weight': None,
 'dual': False,
 'fit_intercept': True,
 'intercept_scaling': 1,
 'max_iter': 100,
 'multi_class': 'warn',
 'n_jobs': None,
 'penalty': 'l1',
 'random_state': None,
 'solver': 'warn',
 'tol': 0.0001,
 'verbose': 0,
 'warm_start': False}
```

- **Predictor**: A interface Predictor estende a interface Estimator por meio da adição de um método `predict`. A partir dos exemplos anteriores, é possível invocar o método `predict` para a regressão logística, obtendo assim as classes preditas pelo modelo. Normalmente faz-se uso do conjunto de teste (neste caso, `X_test`) como os dados para os quais queremos fazer a predição.

```
y_pred = clf.predict(X_test}
```

Alguns algoritmos implementados na biblioteca Scikit-Learn disponibilizam o método `predict_proba`, o qual retorna a probabilidade de pertinência de uma instância para cada classe candidata. A interface Predictor também disponibiliza um método denominado `score` voltado para a avaliação do desempenho do modelo. Em modelos de aprendizado supervisionado, este método normalmente recebe como parâmetros o conjunto de dados teste e os rótulos de teste (`X_test` e `y_test`, respectivamente).

1.5.1. Conhecendo o conjunto de dados

As doenças cardiovasculares são uma das principais causas de morte nos Estados Unidos, sendo contabilizadas em mais de 30% das ocorrências anuais. Quando detectadas precocemente é possível reduzir os infartos em 80%, assim como a mortalidade em 45% (LADERAS et al., 2018).

Na busca por fomentar o acesso a base de dados clínicos que possam auxiliar no desenvolvimento de modelos preditivos para essas doenças, Laderas et al. (2018) cria-

ram um conjunto de dados sintéticos de modo a possibilitar o desenvolvimento de modelos preditivos para doenças cardiovasculares. Para a elaboração do conjunto de dados, buscou-se incluir dependências realísticas entre as variáveis (por exemplo, pacientes com alto índice de massa corpórea (IMC) têm maior prevalência de diabetes tipo 2), bem como buscou-se apresentar baixa prevalência da doença para determinadas coortes (pessoas com menos de 40 anos, por exemplo). O conjunto de dados sintéticos possui 425.195 observações e as variáveis disponíveis estão apresentadas na Tabela 1.2.

Todos os códigos exibidos neste capítulo foram executados na plataforma Google Colaboratory e estão disponíveis no endereço eletrônico: <https://tinyurl.com/SBCAS2019-ML>.

Para fins de simplificação, alguns comandos e importações de bibliotecas serão omitidos, mas o conteúdo completo dos comandos encontra-se disponível no arquivo disponibilizado.

Os comandos que seguem são executados sob a variável `dataset`, que representa o arquivo de dados lido a partir de seu repositório, conforme segue:

```
1 dataset = pd.read_csv('https://git.io/fjm0H')
```

Tabela 1.2. Variáveis disponíveis no conjunto de dados cvdRiskData.

Variável	Definição	Tipo	Possíveis valores
patientID	Identificador único do paciente	Alfanumérica	Diversos
age	Faixa etária do paciente	Categórica	0-20, 20-40, 40-55, 55-70, 70-90
htn	Identifica se o paciente possui hipertensão	Categórica	Y, N
treat	Identifica se o paciente recebe tratamento para a hipertensão	Categórica	Y, N
smoking	Identifica se o paciente é fumante	Categórica	Y, N
race	Identifica a raça informada pelo paciente	Categórica	AmInd (america indian), Asian/PI (asian/pacif islander), Black/AfAm (black/african american), White
gender	Identifica o sexo do paciente	Categórica	Male, Female NA significa que o paciente não quis registrar essa informação
t2d	Identifica se o paciente possui Diabetes tipo 2	Categórica	Y, N
numAge	Idade do paciente, em anos	Contínua	0-90
bmi	Índice de massa corpórea (IMC)	Contínua	15-36
tchol	Colesterol total	Contínua	155-245
sbp	Pressão arterial sistólica	Contínua	74-226
cvd	Identifica se o paciente possui doença cardiovascular	Categórica	Y, N

1.5.2. Preparação dos dados

Para a predição de risco cardiovascular, optou-se por filtrar do conjunto de dados os registros de indivíduos que possuam idade superior a 55 anos. Além disso, as variáveis preditoras `patientID`, `age` e `treat` serão removidas uma vez que `patientID` é o

ID do paciente, que não tem relacionamento algum com a variável de interesse, `age` já se encontra representada pela variável `numAge` e `treat` só foi respondida por pacientes com hipertensão.

O código abaixo realiza o filtro da idade e remove as variáveis elencadas. Tal funcionalidade é provida na biblioteca `dfply`, a qual tenta reproduzir na linguagem Python as funcionalidades da biblioteca R `dplyr`. Na linha ❷ há a aplicação do filtro de idades acima de 55 anos, enquanto que na linha ❸ ocorre a remoção das variáveis elencadas. Na biblioteca `dfply`, a variável `X` é uma referência ao conjunto de dados que antecede o símbolo », neste caso `dataset`. Assim, `X.numAge` representa a coluna `numAge` do conjunto de dados `dataset`.

```
cvd_data = (dataset >>
                mask(X.numAge > 55) >>
                drop(X.patientID, X.age, X.treat)
               )
```

Uma vez realizado o fitro e exclusão das variáveis, passa-se para o tratamento das variáveis categóricas, as quais devem ser decompostas em um conjunto de variáveis indicadoras (*dummies*), conforme segue:

```
cvd_data = pd.get_dummies(cvd_data, columns=['htn', 'smoking', 't2d',
  ↪  'gender', 'race'])
```

A linha ❶ faz uso da função `get_dummies`, da biblioteca Pandas, que irá converter uma variável categórica em n colunas, onde n é o número de valores distintos da variável.

As variáveis contínuas (`numAge`, `bmi`, `tchol`, `sbp`) serão padronizadas. Para isso, o código a seguir é utilizado.

```
from sklearn.compose import ColumnTransformer

### variáveis contínuas que serão padronizadas
continuous_cols = ['numAge', 'bmi', 'tchol', 'sbp']
def setScaler():
  ct = ColumnTransformer([
        ('scaler', StandardScaler(), continuous_cols)
    ], remainder='passthrough'
  )
  return ct

scaler = setScaler()
```


Como será feita a padronização apenas das variáveis contínuas, será utilizada a funcionalidade `ColumnTransformer` do Scikit-Learn, que permite que um transformador (no caso, `StandardScaler()` ❼) seja aplicado apenas nas colunas definidas em `continuous_cols`.

A padronização das variáveis deve ser feita apenas com base no conjunto de treinamento, sendo posteriormente aplicada ao conjunto de teste. Assim, é necessário dividir o conjunto de dados em conjunto de treinamento e conjunto de teste, como segue:

```
variaveis_preditoras = cvd_data.iloc[:, cvd_data.columns != 'cvd']
classe = cvd_data.iloc[:, cvd_data.columns == 'cvd']
X_train, X_test, y_train, y_test =
  ↪ train_test_split(variaveis_preditoras, classe, train_size = 0.70,
  ↪ random_state = 42)
```

A linha ❶ seleciona as variáveis preditoras (todas aquelas diferentes de `cvd`). A linha ❷ seleciona a variável de interesse (`cvd`) e a armazena em `classe`. A linha 3, por sua vez, faz a divisão do conjunto de dados em 70% para treinamento (`train_size`) e 30% para teste.

Uma vez particionados, pode-se aplicar a padronização das variáveis contínuas.

```
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

Baseando-se no fluxo de aplicação de transformadores exibido na Figura 1.13, na linha ❶ o transformador é ajustado para os dados de treinamento, enquanto que na linha ❷ o conjunto de treinamento (`X_train`) tem suas variáveis contínuas padronizadas, assim como o conjunto de teste na linha seguinte.

Ao término do pré-processamento o conjunto de dados de treinamento se apresenta conforme exibido na Figura 1.14. Variáveis categóricas foram transformadas em *dummies*. A variável `smoking`, por exemplo, apresentava dois possíveis valores (`Y`, `N`), tendo sido, portanto, transformada em duas colunas: `smoking_Y`, e `smoking_N`. As variáveis contínuas, como `bmi`, foram padronizadas.

1.5.3. Implementação dos modelos

A seguir apresenta-se a execução dos algoritmos de Regressão Logística, K-vizinhos mais próximos, Árvore de Decisão, Random Forest, Gradient Boosted Trees e Redes Neurais Artificiais.

Tais algoritmos irão ajustar modelos de classificação para a variável de interesse `cvd`, predizendo assim a ocorrência de doença cardiovascular a partir do conjunto de variáveis preditoras. Após ajuste dos modelos ao conjunto de treinamento, serão obtidas as seguintes métricas sob o conjunto de teste: (i) precisão, (ii) sensibilidade (recall), (iii)

numAge	bmi	tchol	sbp	htn_N	htn_Y	smoking_N	smoking_Y	t2d_N	t2d_Y	gender_F	gender_M	race_AmInd	race_Asian/PI	race_Black/AfAm	race_White
1.273356	-1.347215	-0.859665	1.195618	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0
-0.659311	0.247619	-0.453873	0.531070	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0
-1.196162	-0.435881	0.695870	0.531070	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0
0.736504	-0.663715	-0.487689	-0.734736	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0
-0.122459	-0.208048	1.338374	-1.367639	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0
-1.088792	2.981619	1.879430	1.512070	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0
-0.015088	-1.119381	-0.994929	1.353844	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0
0.521763	-0.208048	0.087183	1.227263	0.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0
0.092282	0.019786	0.154814	-0.798026	1.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0
-0.551940	-1.119381	0.154814	0.784231	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0

Figura 1.14. Amostra do conjunto de dados de treinamento após a etapa de pré-processamento.

F-score e (iv) AUC. Além disso, os modelos terão suas curvas de calibração analisadas e mensuradas pela métrica *brier score*.

Para facilitar a execução dos modelos de classificação, foi construída a função *runModel()* que irá executar os modelos preditivos, definida como segue:

```
def runModel(model, X_train, y_train, X_test, y_test,
  ↪  confusion_matrix=True, normalizeCM=False, roc=True,
  ↪  plot_calibration=True, random_state=42, title="", pos_label=1):

    clf = model
    name = title
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    if hasattr(clf, "predict_proba"):
        prob_pos = clf.predict_proba(X_test)
    else:
        prob_pos = clf.decision_function(X_test)
        prob_pos = \
            (prob_pos - prob_pos.min()) / (prob_pos.max() -
              ↪  prob_pos.min())

    prob_pos = prob_pos[:,1]
    clf_score = brier_score_loss(y_test, prob_pos,
      ↪  pos_label=pos_label)
    print("%s:" % name)
    print("\tBrier: %1.3f" % (clf_score))
    print("\tPrecision: %1.3f" % precision_score(y_test, y_pred))
    print("\tRecall: %1.3f" % recall_score(y_test, y_pred))
    print("\tF1: %1.3f\n" % f1_score(y_test, y_pred))

    if confusion_matrix:
        skplt.metrics.plot_confusion_matrix(y_test, y_pred,
          ↪  normalize=normalizeCM, title=name)
    if roc:
```



```
25          skplt.metrics.plot_roc(y_test, prob_pos, plot_micro=False,
              ↪ plot_macro=False, classes_to_plot=[1],
              ↪ title=name,figsize=(10,10))
26
27      if plot_calibration:
28          fraction_of_positives, mean_predicted_value = \
29                  calibration_curve(y_test, prob_pos, n_bins=10)
30          fig = plt.figure(3, figsize=(10, 10))
31          ax1 = plt.subplot2grid((3, 1), (0, 0), rowspan=2)
32          ax2 = plt.subplot2grid((3, 1), (2, 0))
33          ax1.plot([0, 1], [0, 1], "k:", label="Perfeitamente
              ↪ calibrado")
34          ax1.plot(mean_predicted_value, fraction_of_positives, "s-",
35                       label="%s (%1.3f)" % (name, clf_score))
36
37          ax2.hist(prob_pos, range=(0, 1), bins=10, label=name,
38                       histtype="step", lw=2)
39          ax1.set_ylabel("Fração de positivos")
40          ax1.set_ylim([-0.05, 1.05])
41          ax1.legend(loc="lower right")
42          ax1.set_title('Gráfico de Calibração  (reliability curve)')
43          ax2.set_xlabel("Valor médio predito")
44          ax2.set_ylabel("Quantidade")
45          ax2.legend(loc="upper center", ncol=2)
46          plt.tight_layout()
47          plt.show()
```

A função *runModel* recebe como parâmetro um modelo a ser ajustado, os conjuntos de treinamento e de teste, além de parâmetros sobre quais saídas serão produzidas. Na linha 5 o modelo é ajustado ao conjunto de treinamento, produzido a predição ($\hat{y}$ = y_pred) na linha 6. Como alguns modelos implementados na biblioteca não possuem o método `predict_proba` (a qual transforma a saída da predição na probabilidade da ocorrência da classe de interesse), as linhas 7 - 12 validam a sua existência, caso contrário, fazem uso do método `decision_function`.

As linhas 16 - 20 exibem os valores do *brier score*, precisão, sensibilidade (recall) e o F-score (F1). A partir da linha 22 os parâmetros da função são validados, de modo a gerar: (i) matriz de confusão 22, curva ROC 24 e curva de calibração 27.

1.5.4. Avaliação do Desempenho

A seguir são apresentados os resultados obtidos com o ajuste dos modelos preditivos escolhidos. Os modelos tiveram seus hiperparâmetros otimizados fazendo-se uso da técnica de Grid Search disponibilizada pela biblioteca Scikit-Learn, que busca exaustivamente todas as combinações no espaço de hiperparâmetros e seus possíveis valores.

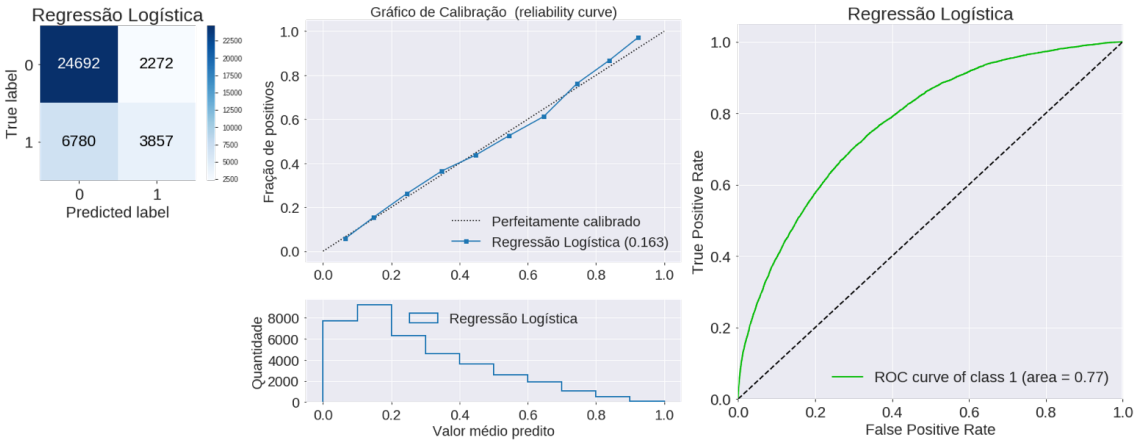


Figura 1.15. Desempenho do modelo de Regressão Logística.

1.5.4.1. Regressão Logística

O código a seguir é responsável pela execução do modelo de regressão logística.

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
runModel(lr, X_train, y_train, X_test, y_test, title="Regressão
  ↪ Logística")
```

O desempenho do algoritmo é apresentado na Figura 1.15. A precisão do algoritmo ficou em 0,629, enquanto a sua sensibilidade foi de 0,363. O valor da AUC do modelo foi de 0,770. Em relação a calibração do modelo, valor do *brier score* foi de 0,163.

1.5.5. KNN

Para o ajuste do modelo de k-vizinhos mais próximos, o seguinte grid foi elaborado:

```
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier()
grid_params = {
        'n_neighbors': np.arange(3,301, 2),
        'weights': ['uniform', 'distance'],
        'metric': ['euclidean', 'manhattan']
}
gs = GridSearchCV(neigh, param_grid=grid_params,
  ↪ scoring='roc_auc',cv=3, n_jobs=-1)
```

Além do valor de *k* (hiperparâmetro n_neighbors), buscou-se verificar os seguintes hiperparâmetros:

- weights: define como se dará o peso das observações de treinamento, no momento da determinação dos vizinhos mais próximos de x. Este peso pode ser uniformemente distribuído, ou inversamente proporcional à distância (quanto menor a distância entre a instância e x, maior será o peso);
- metric: define a métrica de distância a ser utilizada. Neste caso, optou-se por testar a distância euclidiana e a distância de manhattan.

Após a execução do grid, obteu-se que a melhor configuração para o modelo é $k = 59$, weights = uniform, e metric = manhattan. O desempenho do modelo treinado, quando aplicado ao conjunto de teste, é apresentado na Figura 1.16. Em relação ao modelo de regressão logística, k-NN apresentou a mesma AUC (0,770), porém com uma maior precisão (0,633). k-NN também apresentou maior sensibilidade do que o modelo de regressão logística, embora com uma calibração inferior (maior *brier score*). Embora o desempenho seja superior, cabe ressaltar o custo computacional do k-NN em relação ao modelo logístico. Por fazer uso de todas as observações de treinamento para a classificação de uma nova instância, o uso de memória, e consequentemente, o tempo de processamento do k-NN é superior ao modelo de regressão logística.

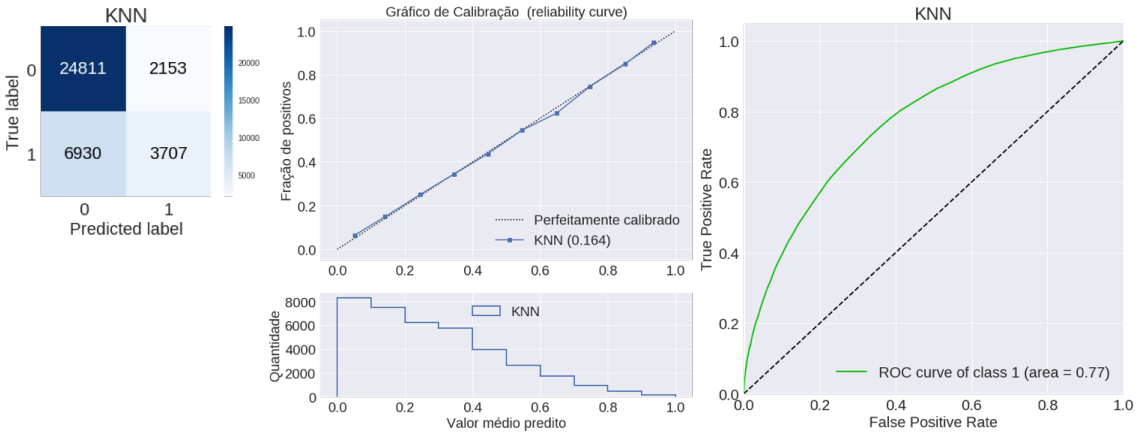


Figura 1.16. Desempenho do modelo de K-vizinhos mais próximos.

1.5.5.1. Árvore de Decisão

O algoritmo de árvore de decisão teve seus hiperparâmetros ajustados pela técnica de Grid Search, exibida no código abaixo.

```
param_grid = {"criterion": ["gini", "entropy"],
              "min_samples_split": [2,5,8,15,20],
              "max_depth": [2,4,6,8,10],
              "min_samples_leaf": [1,2,4,8,10],
              "max_leaf_nodes": [2,4, 7,9, 12, 20],
              }
dt = DecisionTreeClassifier(random_state=42)
```



```
cv_dt = GridSearchCV(dt, cv = 3,
                     param_grid=param_grid,
                     n_jobs = -1)
```

Os seguintes hiperparâmetros foram ajustados:

- criterion: função utilizada para avaliar o atributo mais adequado para subdividir a árvore de decisão. No caso, serão testadas duas opções: entropia e índice de gini;
- min_samples_split: número mínimo de observações que um nó deve possuir na árvore para que seja possível uma nova divisão. Serão testados os valores 2, 5, 8, 15 e 20;
- max_depth: limita a profundidade da árvore. Serão testados os valores 2, 4, 6, 8 e 10;
- min_samples_leaf: número mínimo de observações que deve existir em uma folha da árvore. Serão testados os valores 1, 2, 4, 8 e 10;
- max_leaf_nodes: número máximo de folhas que a árvore pode possuir. Serão testados os valores 2, 4, 7, 9, 12 e 20.

Após a execução do grid, onde todas as combinações possíveis dos hiperparâmetros selecionados foram testadas, o modelo que apresentou o melhor resultado foi:

```
Model with rank: 1
Mean validation score: 0.755 (std: 0.001)
Parameters: {'criterion': 'entropy', 'max_depth': 6, 'max_leaf_nodes':
  ↪  20, 'min_samples_leaf': 10, 'min_samples_split': 20}
```

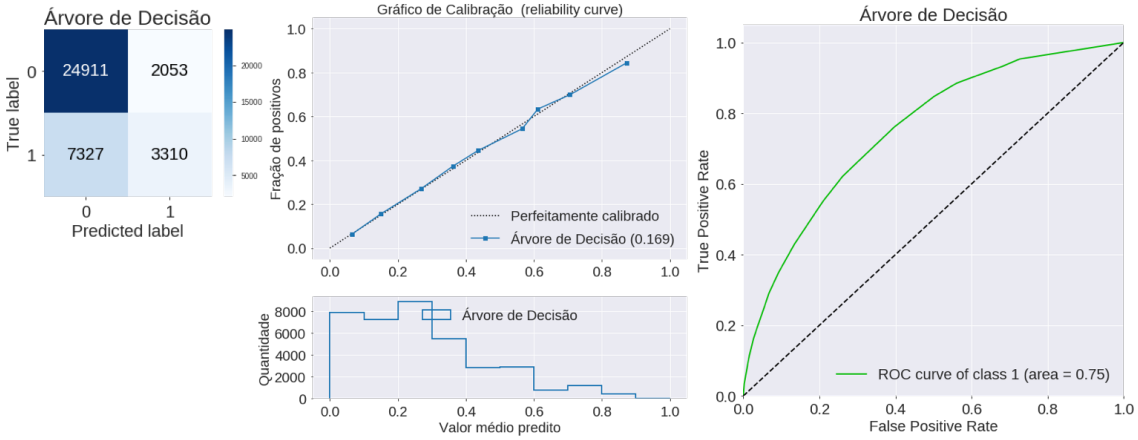


Figura 1.17. Desempenho do modelo de Árvore de Decisão.

Os resultados obtidos estão apresentados na Figura 1.17. O modelo obteve o menor desempenho em relação aos dois modelos anteriores (regressão logística e k-vizinhos mais próximos): AUC de 0,750, precisão de 0,617, sensibilidade de 0,311 e *brier score* de 0,169.

A biblioteca `dtreeviz` permite a visualização da árvore de decisão criada. A Figura 1.18 apresenta o caminho percorrido na árvore de decisão para predizer uma instância aleatoriamente obtida do conjunto de teste. É possível observar que a predição negativa para risco cardiovascular foi realizada a partir dos atributos `sbp`, `gender`, `numAge` e `race`.

1.5.5.2. Random Forest

Para o ajuste do modelo de Random Forest, os seguintes hiperparâmetros foram otimizados:

```
n_estimators = [int(x) for x in np.linspace(start = 100, stop = 1000,
  ↪  num = 5)]
max_features = ['log2', 'sqrt']
max_depth = [int(x) for x in np.linspace(5, 20, num = 5)]
min_samples_split = [2, 5, 10]
min_samples_leaf = [2, 4]
bootstrap = [True, False]

param_grid = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf,
               'bootstrap': bootstrap}

cv_rf = GridSearchCV(fit_rf, cv=3, param_grid=param_grid,
                     n_jobs = -1)
```

- n_estimator: número de árvores a serem construídas no comitê;
- max_features: número de características a serem utilizadas a cada split da árvore;
- max_depth: profundidade da árvore;
- min_samples_split: número mínimo de observações que um nó deve possuir na árvore. para que uma nova divisão seja possível;
- min_samples_leaf: número mínimo de observações que deve existir em uma folha da árvore;

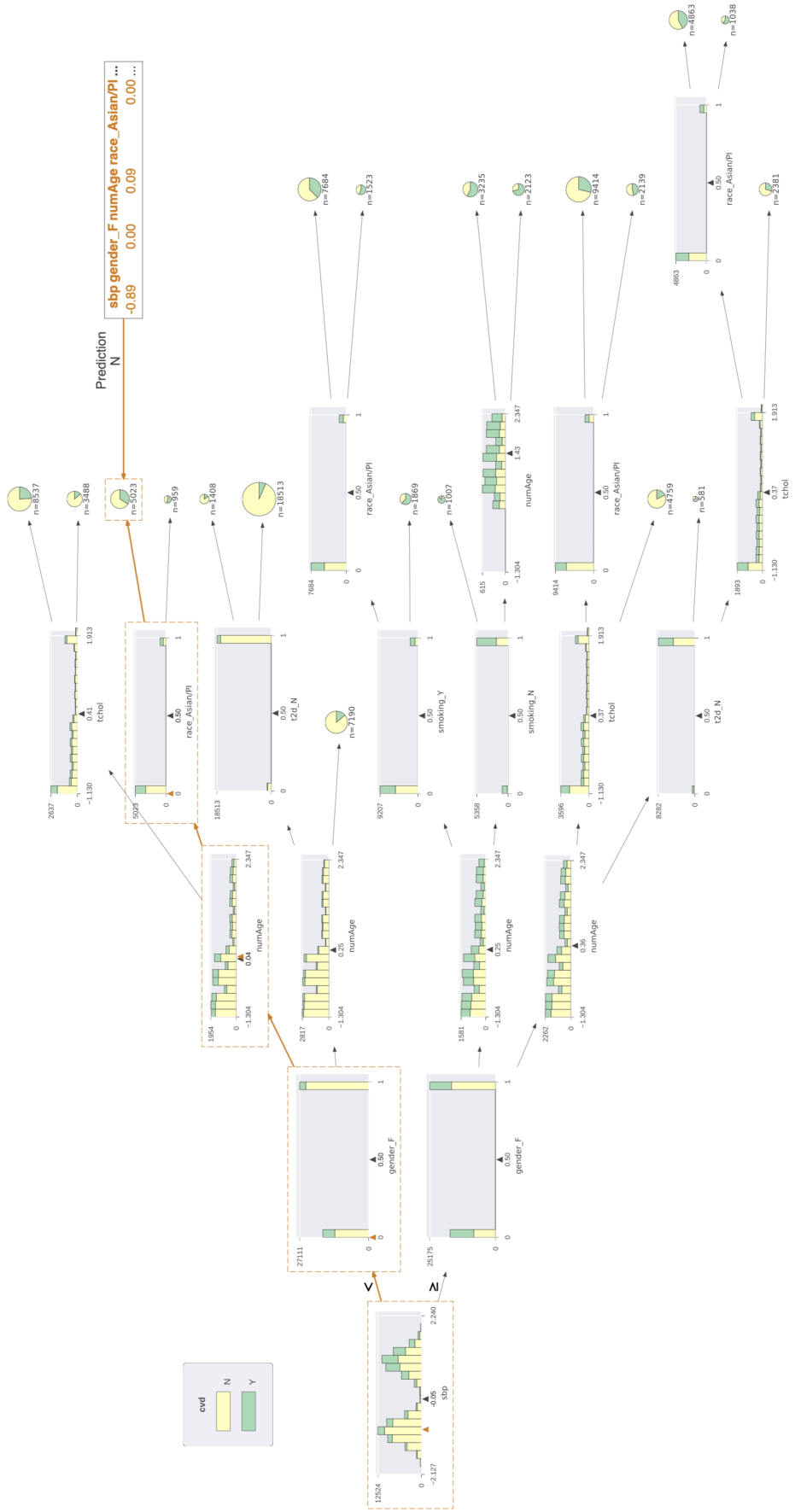


Figura 1.18. Árvore de decisão para a predição de risco cardiovascular. Caminho percorrido para predição de uma instância aleatória obtida do conjunto de teste.

- bootstrap: indica se a técnica de bootstrap será utilizada.

Após a execução do grid, o valores dos hiperparâmetros foram os seguintes:

```
Model with rank: 1 Mean validation score: 0.763 (std: 0.001)
  ↪  Parameters: {'bootstrap': True, 'max_features': 'sqrt',
  ↪  'n_estimators': 775, 'max_depth': 10, 'min_samples_split': 2,
  ↪  'min_samples_leaf': 2}
```

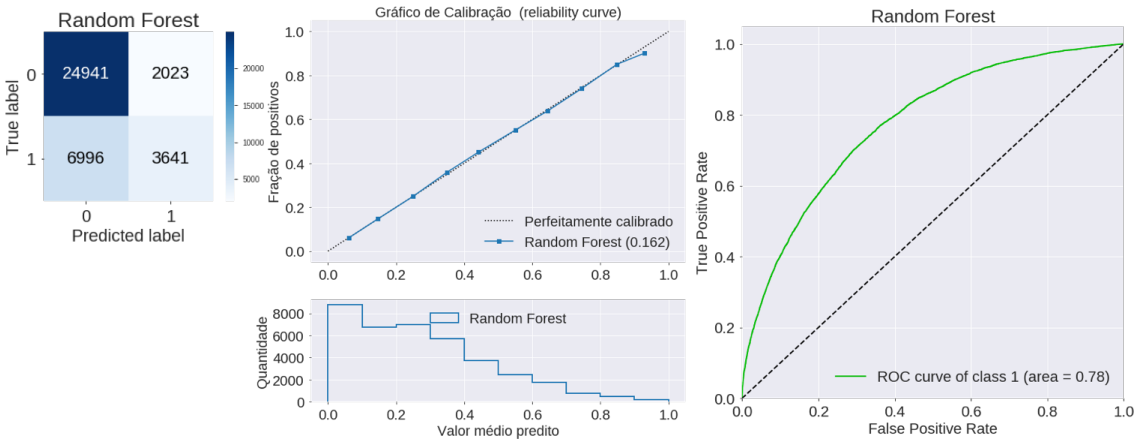


Figura 1.19. Desempenho do modelo de Random Forest.

A Figura 1.19 apresenta o resultado da execução do algoritmo sob o conjunto de teste. O modelo supera os três modelos anteriormente elaborados, com uma AUC de 0,780, e valores superiores de precisão (0,643), embora apresentando menor sensibilidade (0,342) quando comparado à regressão logística (0,363) e K-vizinhos mais próximos (0,349), o que vai ao encontro da relação precisão-sensibilidade existente nos modelos preditivos (DAVIS; GOADRICH, 2006), onde um aumento da precisão leva a uma redução da sensibilidade, ao passo em que um aumento da sensibilidade leva a uma redução da precisão. Em relação a sua eficiência como classificador probabilístico, o modelo apresentou um baixo *brier score* (0,162), superando os modelos anteriormente desenvolvidos.

1.5.5.3. Gradient Boosted Trees

Para o ajuste dos dados ao modelo de Gradient Boosted Trees será feito uso da implementação XGBoost (Extreme Gradient Boosting). Trata-se de uma melhoria do algoritmo Gradient Boosting proposto por Friedman (2001), cujas modificações estão descritas em Chen e Guestrin (2016). XGBoost vem apresentando bons resultados em competições de machine learning, de modo que a teoria para seu funcionamento ainda se encontra em consolidação pela comunidade científica.

Os seguintes hiperparâmetros foram considerados para otimização:


```
params_xgb = {
        'learning_rate': [0.02, 0.03,0.04],
        'n_estimators': [500,700,800],
        'min_child_weight': [1, 5, 10],
        'gamma': [0.5, 1, 1.5, 2, 5],
        'subsample': [0.6, 0.8, 1.0],
        'colsample_bytree': [0.6, 0.8, 1.0],
        'max_depth': [5,10,20,40]
        }
xgb = XGBClassifier(objective='binary:logistic',silent=True,
  ↪  nthread=-1)
gridXGB = GridSearchCV(xgb, param_grid=params_xgb,
                               scoring='AUC', n_jobs=-1, cv=3,
                                 ↪  verbose=1)
```

- learning_rate: utilizada para evitar uma rápida convergência do modelo, ponderando a força do ajuste a ser feito nas novas árvores;
- n_estimators: número máximo de árvores a serem utilizadas pelo modelo;
- min_child_weight: número mínimo de observações em uma folha;
- gamma: parâmetro de regularização do modelo, na busca por prevenir sobreajuste;
- subsample: controla o número de amostras que serão fornecidas ao modelo;
- colsample_bytree: controla o número de variáveis preditoras que serão oferecidas ao modelo;
- max_depth: controla a profundidade máxima da árvore.

Após a execução do grid, a melhor combinação de hiperparâmetros foi a seguinte:

```
XGBClassifier(gamma=0.0, learning_rate=0.1, max_depth=5,
  ↪  min_child_weight=5,
       n_estimators=60,  subsample=0.6)
```

O desempenho do modelo pode ser observado na Figura 1.20. Em relação aos modelos anteriormente elaborados, o XGBoost apresentou AUC (0,780) equivalente ao modelo de Random Forest, porém com valores superiores de precisão (0,647) e sensibilidade (0,344), e com a mesma calibração (0,162).

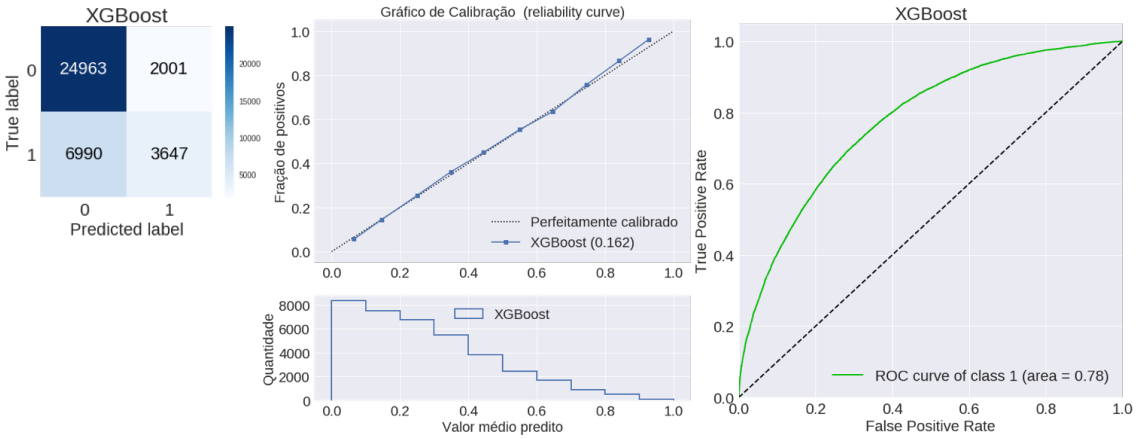


Figura 1.20. Desempenho do modelo XGBoost.

1.5.5.4. Redes Neurais Artificiais

Para a construção da RNA, os seguintes hiperparâmetros foram considerados:

```
parameters_rna = {'activation':['relu', 'tanh'],
                  'solver': ['adam', 'lbfgs', 'sgd'],
                  'max_iter': [500,1000],
                  'alpha': 10.0 ** -np.arange(1, 3),
                  'hidden_layer_sizes':[2,3,5,7],
                  'batch_size':np.arange(5,15,32),
                  'learning_rate_init':[0.01, 0.03, 0.1] }

rna = neural_network.MLPClassifier(verbose=True)
```

- activation: indica qual função de ativação será utilizada pelos neurônios artificiais;
- solver: método utilizado para ajuste dos pesos;
- max_iter: número de épocas de treinamento da rede neural;
- alpha: parâmetro de regularização do modelo;
- hidden_layer_sizes: número de neurônios na camada oculta;
- batch_size: número de exemplos de treinamento usados em uma iteração. Após uma iteração, os pesos da rede são ajustados;
- learning_rate_init: valores iniciais da taxa de aprendizado.

Após a execução do grid, a seguinte combinação de hiperparâmetros obteve o melhor resultado no conjunto de treinamento:

```
Model with rank: 1
Mean validation score: 0.779 (std: 0.004)
Parameters: {'activation': 'tanh', 'alpha': 0.1, 'batch_size': 5,
  ↪ 'hidden_layer_sizes': 7, 'learning_rate_init': 0.1, 'max_iter':
  ↪ 1000, 'solver': 'lbfgs'}
```

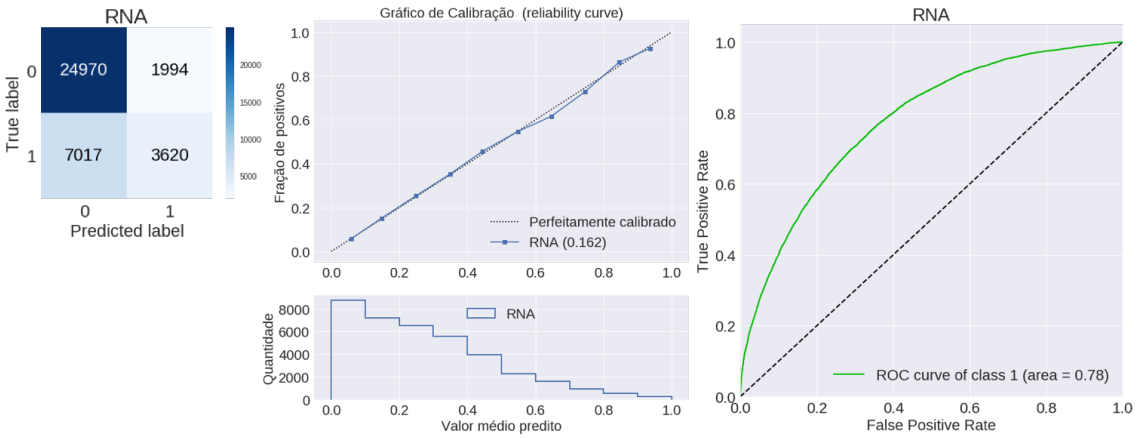


Figura 1.21. Desempenho do modelo de Redes Neurais Artificiais.

A Figura 1.21 apresenta os resultados do modelo quando aplicado ao conjunto de teste. O modelo apresentou a mesma AUC (0,780) que o modelo XGBoost, porém com precisão (0,645) e sensibilidade (0,341) inferiores. Em relação a calibração, o modelo de RNA foi que apresentou a melhor calibração (0,161) dentre todos os modelos elaborados.

A Tabela 1.3 apresenta os resultados consolidados dos modelos elaborados para a predição do risco cardiovascular em pessoas acima de 55 anos. O modelo baseado em um comitê de árvores de decisão e técnica de boosting (Gradient Boosted Trees, implementado pelo XGBoost) apresentou a maior eficiência preditiva dentre os modelos elaborados, seguido pelos modelos de Redes Neurais Artificiais e Random Forest.

Em relação a calibração dos modelos, XGBoost apresentou um bom resultado de calibração. A importância da análise da calibração dos modelos preditivos se dá pela capacidade das probabilidades preditas serem agregadas em níveis de riscos, ajudando a priorizar indivíduos em decisões relacionadas a ações de tratamento ou prevenção.

1.6. Considerações Finais

O aumento expressivo no volume de dados sendo gerados e/ou consumidos por sistemas e aplicações requer a adoção de técnicas voltadas para a análise e a extração de conhecimento, sistemas e metodologias para a gestão dos dados, buscando uma tomada de decisão com base em uma melhor compreensão dos dados.

Conforme enfatizado por Obermeyer e Emanuel (2016), a adoção de algoritmos para a análise e interpretação de conjuntos de dados irá fornecer o ferramental necessário para a compreensão e tomada de decisão. A adoção de técnicas de Machine Learning

Tabela 1.3. Ranking dos modelos preditivos para o risco cardiovascular, ordenados decrescentemente por AUC e Precisão.

Modelo	AUC	Precisão	Sensibilidade	Brier
Gradient Boosted Trees	0,780	0,647	0,344	0,162
Redes Neurais Artificiais	0,780	0,645	0,341	0,161
Random Forest	0,780	0,643	0,342	0,162
K-vizinhos mais próximos	0,770	0,633	0,349	0,164
Regressão Logística	0,770	0,629	0,363	0,163
Árvore de Decisão	0,750	0,617	0,311	0,169

busca dotar os computadores de funções que permitam extrapolar a informação implícita nos dados, não apenas restringindo-se a executar um conjunto de ações pré-determinadas pelo programador.

A área da Saúde apresenta forte potencial para a adoção de Machine Learning. Desde atividades como predição de ocorrência de uma determinada doença, até análises complexas de imagens de ressonâncias 3D, podem se beneficiar da velocidade e capacidade de realizar cálculos n-dimensionais dos algoritmos disponíveis.

Todavia, deixar que os dados falem por si sós não é uma tarefa trivial. Algoritmos podem sofrer sobreajuste para eventuais correlações espúrias presentes nos dados, assim como multicolinearidade das variáveis preditoras pode afetar a capacidade de predição e generalização dos modelos preditivos. Dessa forma, é necessária a adoção de técnicas de pré-processamento que buscam lidar com valores ausentes, *outliers*, dentre outros problemas, de modo a produzir modelos preditivos que apresentem bons resultados diante de novas instâncias a serem classificadas.

A capacidade preditiva de Machine Learning não deve ser confundida com a necessidade de inferência causal (OBERMEYER; EMANUEL, 2016). Modelos de Machine Learning podem apresentar excelentes resultados de predição e destacar quais preditores estão contribuindo significativamente para tal predição. Todavia, a importância de um preditor não necessariamente está ligada com a causa do fenômeno de interesse.

A adoção de Machine Learning nas mais diversas áreas tem sido impulsionada pela disponibilidade e variedade de ferramentas computacionais para o ajuste de modelos preditivos. Este Capítulo utilizou a biblioteca Scikit-Learn (BUITINCK et al., 2013), desenvolvida na linguagem de Programação Python, que apresenta considerável popularidade entre a comunidade científica e profissional.

Foram apresentados modelos preditivos baseados em algoritmos que seguem o paradigma do aprendizado supervisionado. Tais algoritmos possibilitaram o ajuste de modelos preditivos em um conjunto de dados sintéticos sobre o risco cardiovascular, apresentando resultados próximos, embora possuindo princípios de funcionamento diferentes.

Há uma forte tendência da popularização do uso de modelos preditivos de Machine Learning na área da Saúde, assim como nas mais diversas áreas do conhecimento. Conhecer os princípios de funcionamento dos algoritmos disponíveis, as técnicas de pré-processamento que se fazem necessárias antes do ajuste dos modelos, e como interpretar

os resultados obtidos será cada vez mais uma tarefa exigida dos profissionais, independente da sua área de formação e atuação.

Referências

BORBOUDAKIS, G.; STERGIANNAKOS, T.; FRYSALI, M.; KLONTZAS, E.; TSAMARDINOS, I.; FROUDAKIS, G. E. Chemically intuited, large-scale screening of mofs by machine learning techniques. *npj Computational Materials*, Nature Publishing Group, v. 3, n. 1, p. 40, 2017.

BRAGA, A. de P.; FERREIRA, A. C. P. de L.; LUDERMIR, T. B. *Redes neurais artificiais: teoria e aplicações*. [S.l.]: LTC Editora Rio de Janeiro, Brazil:, 2007.

BUITINCK, L.; LOUPPE, G.; BLONDEL, M.; PEDREGOSA, F.; MUELLER, A.; GRISEL, O.; NICULAE, V.; PRETTENHOFER, P.; GRAMFORT, A.; GROBLER, J. et al. Api design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238*, 2013.

CHAMBERS, M.; DINSMORE, T. W. *Advanced analytics methodologies: Driving business value with analytics*. [S.l.]: Pearson Education, 2014.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: ACM. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. [S.l.], 2016. p. 785–794.

CHIAVEGATTO FILHO, A. D. P. Uso de big data em saúde no brasil: perspectivas para um futuro próximo. *Epidemiologia e Serviços de Saúde*, SciELO Public Health, v. 24, p. 325–332, 2015.

DAVIS, J.; GOADRICH, M. The relationship between precision-recall and roc curves. In: ACM. *Proceedings of the 23rd international conference on Machine learning*. [S.l.], 2006. p. 233–240.

FACELI, K.; LORENA, A. C.; GAMA, J.; CARVALHO, A. C. P. d. L. F. C. *Inteligência Artificial: Uma abordagem de aprendizado de máquina*. [S.l.]: Grupo Gen-LTC, 2011.

FAVERO, L. P. L.; BELFIORE, P. P.; SILVA, F. L. d.; CHAN, B. L. *Análise de dados: modelagem multivariada para tomada de decisões*. [S.l.]: Elsevier, 2009.

FRIEDMAN, J. H. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, JSTOR, p. 1189–1232, 2001.

GÉRON, A. *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. [S.l.]: "O'Reilly Media, Inc.", 2017.

GREEN, M. A. Use of machine learning approaches to compare the contribution of different types of data for predicting an individual's risk of ill health: an observational study. *The Lancet*, Elsevier, v. 392, p. S40, 2018.

HAYKIN, S. *Neural networks: a comprehensive foundation*. [S.l.]: Prentice Hall PTR, 1999.

JIANG, L.; CAI, Z.; WANG, D.; JIANG, S. Survey of improving k-nearest-neighbor for classification. In: IEEE. *Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)*. [S.l.], 2007. v. 1, p. 679–683.

KAUFMAN, S.; ROSSET, S.; PERLICH, C.; STITELMAN, O. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, ACM, v. 6, n. 4, p. 15, 2012.

KUHN, M.; JOHNSON, K. *Applied predictive modeling*. [S.l.]: Springer, 2013. v. 26.

LADERAS, T.; VASILEVSKY, N.; PEDERSON, B.; HAENDEL, M.; MCWEENEY, S.; DORR, D. Teaching data science fundamentals through realistic synthetic clinical cardiovascular data. *bioRxiv*, Cold Spring Harbor Laboratory, 2018.

MAGALHÃES, M. N.; LIMA, A. C. P. de. *Noções de probabilidade e estatística*. [S.l.]: Editora da Universidade de São Paulo, 2002. v. 5.

MATTMANN, C. a. Computing: A vision for data science. *Nature*, v. 493, n. 7433, p. 473–475, 2013. ISSN 1476-4687. Disponível em: <http://www.nature.com/nature/journal/v493/n7433/full/493473a.html>.

MAYRINK, V. T. d. M. et al. *Avaliação do algoritmo Gradient Boosting em aplicações de previsão de carga elétrica a curto prazo*. Dissertação (Mestrado) — Universidade Federal de Juiz de Fora (UFJF), 2016.

OBERMEYER, Z.; EMANUEL, E. J. Predicting the future—big data, machine learning, and clinical medicine. *The New England journal of medicine*, NIH Public Access, v. 375, n. 13, p. 1216, 2016.

OBERMEYER, Z.; LEE, T. H. Lost in thought—the limits of the human mind and the future of medicine. *New England Journal of Medicine*, Mass Medical Soc, v. 377, n. 13, p. 1209–1211, 2017.

OLIVERA, A. R.; ROESLER, V.; IOCHPE, C.; SCHMIDT, M. I.; VIGO, Á.; BARRETO, S. M.; DUNCAN, B. B. Comparison of machine-learning algorithms to build a predictive model for detecting undiagnosed diabetes-elsa-brasil: accuracy study. *Sao Paulo Medical Journal*, SciELO Brasil, v. 135, n. 3, p. 234–246, 2017.

OLSON, R. S.; CAVA, W. L.; MUSTAHSAN, Z.; VARIK, A.; MOORE, J. H. Data-driven advice for applying machine learning to bioinformatics problems. *arXiv preprint arXiv:1708.05070*, World Scientific, 2017.

PARR, T.; HOWARD, J. *How to explain gradient boosting*. 2018. Disponível em: <https://explained.ai/gradient-boosting/descent.html>.

RASCHKA, S.; MIRJALILI, V. *Python machine learning*. [S.l.]: Packt Publishing Ltd, 2017.

SAKR, S.; ELSHAWI, R.; AHMED, A. M.; QURESHI, W. T.; BRAWNER, C. A.; KETEYIAN, S. J.; BLAHA, M. J.; AL-MALLAH, M. H. Comparison of machine learning techniques to predict all-cause mortality using fitness data: The Henry Ford exercIse testing (FIT) project. *BMC Medical Informatics and Decision Making*, BMC Medical Informatics and Decision Making, v. 17, n. 1, p. 1–15, 2017. ISSN 14726947.

SELTZER, M. L.; ZHANG, L. The Data Deluge: Challenges and Opportunities of unlimited data in statistical signal processing. p. 3701–3704, 2009.

WENG, S. F.; REPS, J.; KAI, J.; GARIBALDI, J. M.; QURESHI, N. Can machine-learning improve cardiovascular risk prediction using routine clinical data? *PloS one*, Public Library of Science, v. 12, n. 4, p. e0174944, 2017.

Capítulo 2

Blockchain e Aplicações em Saúde

Arlindo F. da Conceição[1],
Vladimir Moreira Rocha[2] e
Ricardo Felipe de Paula[1]

Abstract

Blockchain technology enables reliable, secure, distributed, and fault-tolerant data storage. With the emergence of smart contracts (programs that run above the Blockchain), the technology is no longer used only for financial purposes but also for more complex applications. This chapter aims to present the fundamental characteristics of Blockchain and how it works. We show the main use cases of Blockchain and a survey on their use focused on the health area. Also, we discuss the application scenarios and the challenges to be overcome for the implementation and deployment of the technology. Finally, we present two of the key platforms for building Blockchain applications: Ethereum and Hyperledger Fabric.

Resumo

A tecnologia Blockchain permite o registro de dados de forma confiável, segura, distribuída e tolerante a falhas. Com o surgimento dos contratos inteligentes (programas que são executados de forma autônoma sobre a tecnologia), a tecnologia deixou de ser usada somente para fins monetários e passou a ser utilizada para aplicações mais complexas. Este capítulo visa apresentar as características fundamentais da Blockchain e como ela funciona. São apresentados os principais casos de uso da Blockchain e um levantamento sobre a sua utilização na área da Saúde. Os cenários de aplicação e os desafios para implantação da tecnologia também são discutidos. Por fim, duas das principais tecnologias para criação de aplicações Blockchain serão apresentadas: Ethereum e Hyperledger Fabric.

[1]Universidade Federal de São Paulo (UNIFESP)
[2]Universidade Federal do ABC (UFABC)

2.1. Introdução

A tecnologia Blockchain, uma nova tecnologia para registro confiável e consenso distribuído [1, 2], oferece alternativas para a criação de sistemas interoperáveis, auditáveis e seguros. A tecnologia foi popularizada em 2008 com a criação da criptomoeda Bitcoin por Satoshi Nakamoto [3], Blockchain tem sido utilizada principalmente para realizar transações financeiras de forma anônima, auditável, confiável e segura, evitando que terceiros (por exemplo, os bancos) intermedeiem essas transações.

O conceito utilizado na Blockchain é o de *distributed ledger* e consiste, basicamente, em uma cadeia ordenada e consistente de transações, distribuída em diversos nós de uma rede *peer-to-peer*. Após o sucesso do Bitcoin, outras tecnologias foram integradas à versão inicial proposta por Nakamoto a fim de otimizar o desempenho da solução [4, 5, 6, 7, 8]. Entre elas, os arcabouços Ethereum [9] e Hyperledger [10] fazem uso dos contratos inteligentes, pequenos programas independentes armazenados na própria Blockchain, que permitem realizar operações na cadeia de registros da Blockchain. Por simplicidade, pode-se pensar nos contratos inteligentes, como sendo similares às *store procedures* existentes nos bancos de dados relacionais [2].

O uso de contratos inteligentes expande o poder da Blockchain, que além de armazenar estados (*e.g.*, saldo de uma conta no contexto financeiro), passa a poder armazenar comportamentos (*e.g.*, enviar mensagens de saldo insuficiente). Com o uso da Blockchain e de contratos inteligentes podemos atender outros contextos mais gerais, por exemplo: verificar a consistência da identificação única de usuários, mediar a interoperabilidade de dados em tempo real, garantir a privacidade das informações e tornar auditável todas as ações de acesso a esses dados.

Estima-se que a Blockchain terá um impacto relevante nos próximos anos [11], com aplicações em: registro da cadeia de fornecimento de insumos e produtos, aplicações de governança digital [12], Internet das Coisas (*Internet of Things* ou IoT) [13], Saúde [14, 15], gestão financeira, registros de imóveis, controle de ativos, registros de certidões (nascimento, casamento, óbito), entre outras categorias de aplicações.

Na área de Saúde, a Blockchain pode ser aplicada no controle de acesso e distribuição de informações sensíveis, na transparência e auditabilidade de prestação de serviços e na interoperabilidade de dados, entre outras situações. Pesquisas recentes, contudo, apontam que apesar do seu potencial, Blockchain é uma ferramenta que não pode ser aplicada com sucesso em todos os casos [16]. Desse modo, os profissionais de Saúde devem aprender a discernir os casos de uso viáveis nos quais a tecnologia realmente faça a diferença.

O objetivo desse minicurso é prover aos profissionais de Saúde, estudantes e pesquisadores o entendimento necessário para analisar a viabilidade de aplicação da tecnologia e os primeiros passos a serem dados na implantação de novos projetos envolvendo Blockchain. Para atingir esse objetivo, este texto apresenta a seguinte estrutura: a Seção 2.2 descreve o funcionamento básico de uma Blockchain. A Seção 2.3 mostra alguns conceitos computacionais importantes para o entendimento da tecnologia Blockchain, estes conceitos são pré-requisitos para a correta compreensão do potencial da tecnologia. A Seção 2.4 apresenta um quadro histórico do desenvolvimento da tecnologia. A seguir, a

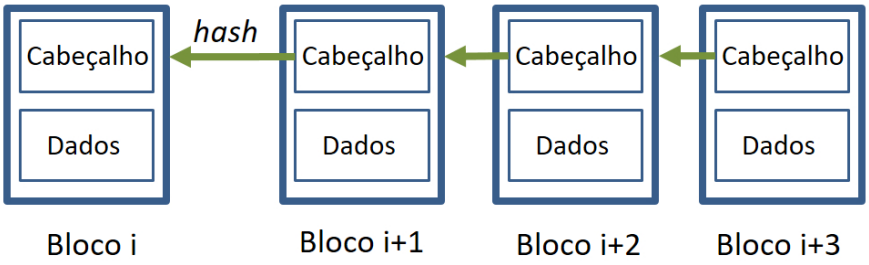


Figura 2.1. Estrutura básica de uma Blockchain

Seção 2.5 explica, passo a passo, como a tecnologia funciona. A Seção 2.6 lista alguns desafios que devem ser enfrentados na utilização de uma Blockchain. A Seção 2.7 apresenta algumas vulnerabilidades de segurança da Blockchain. A Seção 2.8 descreve em quais cenários há a necessidade de utilizar Blockchain. A Seção 2.9, que é o foco principal deste minicurso, apresenta os possíveis cenários de Blockchain aplicados à Saúde. A Seção 2.10, apresenta as tecnologias Ethereum e Hyperledger Fabric, que são as duas ferramentas/ambientes mais utilizados atualmente (destaco: abril de 2019) para a criação de novas aplicações baseadas em Blockchain. Então, a Seção 2.11 lista alguns recursos e leituras para um maior contato com a tecnologia. Por fim, na Seção 2.12, apresentamos nossas considerações finais e perspectivas sobre o futuro da tecnologia.

2.2. Estrutura básica de uma Blockchain

Blockchain implementa algo similar a um livro razão distribuído (*distributed ledger*) e consiste, basicamente, em uma cadeia ordenada e consistente de blocos; por isso o nome *Blockchain*. A Figura 2.1 ilustra a estrutura de blocos encadeados.

Outra característica importante é a de que a estrutura de blocos é replicada em uma rede *peer-to-peer*. Sempre que um novo bloco é criado, ele é enviado para todos os nós da rede. Cada nó verifica os dados do bloco antes dele ser efetivamente incorporado na cadeia de blocos.

As próximas seções explicam o funcionamento geral da estrutura de um bloco e sobre o processo de replicação. Na Seção 2.5 serão dados mais detalhes sobre o funcionamento da Blockchain.

2.2.1. Estrutura de Dados do Bloco

O bloco de uma Blockchain, em geral, pode ser dividido em duas partes: cabeçalho e dados.

O cabeçalho possui as informações responsáveis por identificar o bloco. Uma destas informações é o campo **identificador**, que consiste em um valor único e sequencial. Cada novo bloco inserido na cadeia terá como identificador o valor do identificador do bloco anterior incrementado em uma unidade. Outro campo é o ***timestamp***, que armazena as informações de data e hora aproximada da criação do bloco. Normalmente existe também um campo de assinatura, responsável por identificar e validar o criador do bloco. Por fim, o cabeçalho pode conter metadados sobre o conteúdo do bloco.

Na parte de dados ficam armazenadas as transações pertencentes ao bloco. Estas transações podem representar qualquer tipo de dados ou atividades: em uma aplicação financeira, por exemplo, podem representar a transferência de valores; em uma aplicação de saúde, podem ser um documento ou o link para um documento contendo dados médicos. Cada transação também possui seu identificador, além da informação de quem a criou, entre outras informações conforme a aplicação.

Cabe chamar a atenção para o fato de que uma transação não necessariamente carrega em si um significado financeiro. Na Blockchain, a transação é uma unidade coerente de informação que será replicada e validada por vários nós do sistema. Se o dado carregado por uma transação não for validável, talvez esse dado não precisasse ser armazenado em uma Blockchain, talvez o dado pudesse ser armazenado em um sistema de informação comum.

2.2.2. Replicação em vários nós: redes *peer-to-peer*

Desde os primórdios da Internet, o fornecimento de recursos (como páginas Web ou arquivos) é dado por uma arquitetura denominada cliente-servidor, onde um computador servidor é responsável por fornecê-los e os computadores clientes por requisitá-los. Essa arquitetura segue vigente até hoje, por exemplo nos sítios Web, onde um servidor fornece a página e seu navegador, atuando como cliente, a requisita.

A rede *peer-to-peer* (ou P2P) pode ser entendida como uma rede conectada de computadores onde cada um deles, denominado *peer*, pode atuar tanto como cliente quanto como servidor. Ela nasce como uma alternativa à arquitetura cliente-servidor, e cujos principais objetivos são: (i) aumentar a disponibilidade do recurso e (ii) aumentar a largura de banda de *upload* do sistema [17].

No primeiro ponto, assim que um *peer* P_A requisita (baixa) um arquivo de um *peer* P_B, este arquivo será armazenado por P_A e disponível para que outro *peer* P_C possa baixá-lo, repetindo o processo. Em outras palavras, o mesmo arquivo estará replicado tanto em P_A quanto em P_B. Note que se o *peer* P_B sair da rede, o *peer* P_C poderá baixar o arquivo de P_A, aumentando assim disponibilidade do mesmo.

O segundo ponto está relacionado ao primeiro. Dado que o arquivo está replicado, este poderá ser baixado de todos os *peers* que o armazenam. Note que na arquitetura cliente-servidor, somente um servidor é responsável por enviar o arquivo aos clientes (limitado à largura de banda de *upload* do servidor). Já na rede *peer-to-peer*, supondo que N *peers* tenham baixado o arquivo, a largura de banda de *upload* para enviar o arquivo será N vezes maior a da cliente-servidor, haja vista que cada *peer* se comporta como um servidor.

As redes *peer-to-peer* atualmente são utilizadas em diversas aplicações e sistemas. Exemplos do uso deste tipo de redes são o aplicativo BitTorrent [18] de compartilhamento de arquivos e o Skype [19] de vídeo-áudio conferência.

2.2.3. Confiança: a principal inovação

O uso da tecnologia Blockchain resolve alguns problemas técnicos que mostraremos mais adiante, mas a sua principal inovação é prover um mecanismo de **confiança**. O fato de

todos os blocos e transações serem validados por todos os nós dificulta que registros incorretos sejam inseridos na Blockchain. Se a maioria dos nós do sistema estiverem trabalhando para o bem da rede, então apenas registros corretos serão inseridos. Assim, a confiança não está em um nó, mas na rede de nós como um todo; a confiança está no comportamento coletivo. Por isso é possível criar aplicações sem uma entidade central confiável (*trusted third party* ou TTP); pois a confiança é depositada na rede e não em TTPs.

Mesmo que um usuário (através de um nó) tente enviar dados falsos ou incorretos na rede, os demais nós podem detectar esse comportamento e não inserir o dado na Blockchain. Podem, inclusive, banir o nó suspeito. O comportamento coletivo da rede é o que importa; enquanto houver interesse da maioria dos nós em que a rede continue apenas com dados corretos, pode-se considerar os registros confiáveis. Nesse ponto, note que os nós que sustentam a rede devem ter interesse em que a rede permaneça confiável; esse interesse pode variar de aplicação para aplicação, mas o interesse é um requisito importante para que a rede possa ser considerada confiável.

2.3. Fundamentos

A tecnologia Blockchain tem conquistado seu espaço pelas características que apresenta no desenvolvimento de aplicações, tais como descentralização, disponibilidade, integridade, auditabilidade e privacidade. A seguir serão analisadas as principais características desta tecnologia.

- **Descentralização da informação**

  A descentralização da informação refere-se à dispersão da informação, evitando que uma entidade central ou absoluta tenha o poder sobre ela. Na Blockchain, a descentralização deve ser observada por dois lados: (i) quem detém o poder de realizar alguma ação em uma informação; (ii) quem possui fisicamente a informação.

  No primeiro caso, se a informação está centralizada em uma instituição, organização ou pessoa, esta tem o poder de realizar qualquer ação sobre a informação. Note que diversas instituições no nosso dia a dia funcionam dessa forma. Por exemplo, no banco, você é dono de uma conta, mas as suas informações pessoais podem ser atualizadas em qualquer momento por algum gerente do banco. Nesse sentido, é o banco quem detém o poder da informação.

  No segundo caso, se a informação está armazenada somente nos computadores da instituição, organização ou pessoa, esta possui fisicamente a informação. Seguindo a linha do exemplo do banco, estes armazenam as informações em infra-estruturas próprias, onde pessoas externas não têm acesso.

  A Blockchain visa que as informações não estejam centralizadas nem da perspectiva do poder para realizar alguma ação, nem da perspectiva do armazenamento físico. No primeiro caso, são os participantes que decidem, em conjunto, que informações podem ser modificadas ou inseridas. Para isso, os participantes precisarão chegar a um acordo (denominado de consenso) se essa informação é válida. No segundo caso, a informação é armazenada nos computadores dos participantes, evitando a centralização física da mesma.

- **Disponibilidade**

  A disponibilidade da informação está muito relacionada à descentralização física. Do momento que a informação encontra-se armazenada nos computadores dos participantes, ela torna-se disponível para ser utilizada independente de se um deles sair do sistema. Por exemplo, se existirem dez computadores que permitem o acesso à mesma informação, nove deles poderiam sair do sistema e a informação ainda estaria acessível pelo computador que restou. Por outro lado, se a informação está centralizada em somente um computador, caso esse computador tenha alguma falha ou saia, ninguém mais poderá ter acesso à informação.

  O conceito utilizado para fornecer a disponibilidade é de replicação da informação. Nesse sentido, a mesma informação precisa estar replicada e distribuída em vários computadores. Como mencionado anteriormente, a Blockchain realiza a replicação utilizando a rede *peer-to-peer*. Além da replicação, caso novas informações sejam inseridas, é necessário que todas as réplicas sejam sincronizadas. Para isso, são utilizadas técnicas de consenso distribuído, que serão explicadas na Seção 2.5.8.

- **Privacidade**

  A privacidade permite que todas as operações na Blockchain, denominadas de transações, possam ser realizadas de forma anônima, evitando que terceiros conheçam exatamente que pessoa (ou instituição) a realizou. Para isso, são utilizadas técnicas de criptografia, que permitem que uma pessoa (no mundo real) possa ser identificada (na Blockchain) somente através de um número. Nesse sentido, um terceiro, mesmo tendo acesso a toda a Blockchain, somente visualizará as operações feitas identificadas por números, sem saber quem é a pessoa ou instituição por trás deles.

  Um ponto importante a mencionar é sobre a obtenção desse número, que corresponde à pessoa ou instituição. Basicamente existem duas abordagens para obtê-lo. A primeira, utilizando uma autoridade certificadora, quem verifica que a pessoa ou instituição, no mundo real, realmente existe. Essa abordagem é utilizada nas Blockchains denominadas privadas, como o Hyperledger, detalhada na Seção 2.10, onde somente algumas pessoas têm acesso ao sistema. A segunda, utilizando uma autoridade certificadora quem não verifica se você realmente existe. Esse conceito pode parecer abstrato, mas pense que, no mundo virtual, as pessoas podem se passar por outras ou até por entes imaginários. Essa abordagem é utilizada nas Blockchains denominadas públicas, como o Ethereum, detalhada na Seção 2.10, onde qualquer pessoa tem acesso ao sistema.

- **Integridade**

  Integridade é um ponto importante dentro da Blockchain. Note que, como a informação pode estar distribuída (replicada) em vários computadores, é necessário confiar em que essa informação é íntegra, ou seja, que não foi alterada por ninguém. Mas, como confiar nas informações que alguém está apresentando se, baseado no ponto anterior da privacidade, você não necessariamente conhece a pessoa ou instituição na vida real?

  Nesse sentido, a Blockchain utiliza o conceito de cadeia, que permite criar um enlace entre as informações. Por quê? Fazendo uma analogia com uma corrente da

vida real, uma pessoa consegue identificar facilmente que houve uma quebra se um elo for rompido; isso significa que a corrente não está íntegra.

Agora, como criar enlaces de informações? Na Blockchain, cada elo corresponde a um bloco de informações. Esse bloco será criado com um identificador único entre todos os blocos que já existem ou que serão criados posteriormente. Dentro desse bloco, serão inseridas as operações (transações) realizadas pelos usuários. O elo entre dois blocos é realizado fazendo com que um bloco contenha um identificador do bloco anterior, formando assim o enlace. Note que se alguém quiser quebrar a integridade de uma cadeia de blocos X-Y-Z (por exemplo quebrando Y), deverá criar um novo bloco W que aponte para o bloco X e fazer com que o bloco Z aponte para o novo bloco W. O problema dessa abordagem é que a criação de blocos custa muito tempo, poder computacional ou energia elétrica, o que evita na prática que possa ser realizado em um tempo adequado, como será explicado na Seção 2.5.8.2.

- **Imutabilidade**

  A imutabilidade na Blockchain refere-se a que as informações (sejam estas as transações contidas em um bloco, ou as informações do cabeçalho do bloco) não poderão ser alteradas a partir do momento que forem inseridas na cadeia.

  Agora, como é possível realizar a imutabilidade se as informações variam? Por exemplo, o saldo de uma pessoa pode variar no dia (pelas operações de crédito e débito de um determinado montante), o prontuário de uma pessoa pode variar no tempo, etc.

  Para permitir a alteração de uma informação, a Blockchain cria um novo bloco, inserindo essa alteração como uma nova transação. Note que com isso, a cadeia conterá todas as modificações realizadas na informação, como se fosse o histórico completo, e não somente o último estado dela.

  Para o exemplo do saldo de uma pessoa, imagine que uma pessoa denominada A acabou de entrar no sistema. Nesse momento é criada uma transação de ingresso. A seguir, uma pessoa B envia 10 unidades para A. Nesse momento é criada uma nova transação onde A recebe 10 unidades. Finalmente, A envia para B 3 unidades. Nesse momento é criada uma nova transação onde a A envia 3 unidades. Note que todas as operações realizadas são inseridas na Blockchain, bem diferente a ter somente o último estado do saldo final de A, que seriam 7 unidades.

  A imutabilidade, que não permite a alteração ou remoção das informações, nem sempre é algo desejado. Por exemplo, suponha um sistema de saúde baseado em Blockchain, que adiciona dados aos prontuários dos pacientes. Em um determinado momento, um paciente pede (amparado pela lei) para remover todas suas informações. Note que a imutabilidade evita que isso aconteça, levando à clínica a ter possíveis problemas legais se as mantêm.

- **Auditabilidade**

  A auditabilidade na Blockchain permite verificar, por qualquer um que possua a cadeia, que todas as informações contidas nela são válidas. A auditabilidade faz uso das características apresentadas acima, como privacidade, integridade, entre outras.

Para verificar se as informações são válidas, é necessário verificar que tanto os blocos quanto as transações são válidas. No primeiro caso, é necessário validar que cada bloco aponte para o bloco anterior, até chegar ao primeiro bloco gerado, seguindo o enlace explicado na integridade de blocos e que será detalhado na Seção 2.5.

Já no segundo caso, é necessário verificar se todas as transações de uma determinada pessoa ou instituição (pelo anonimato, somente será mostrado um identificador) apresentam coerência no contexto em que está sendo utilizada a Blockchain.

Por exemplo, a coerência no contexto de operações financeiras está relacionada com os fundos para realizar compras de uma determinada pessoa. Quem realizar a auditoria pode verificar se, em um determinado momento, a pessoa tinha saldo suficiente, utilizando a condição de imutabilidade apresentada anteriormente.

2.4. Revisão histórica sobre o desenvolvimento de Blockchain

A tecnologia de Blockchain foi popularizada com a criação da criptomoeda Bitcoin. Assim, a tecnologia denominada de primeira geração, nasce como base tecnológica para realizar transações financeiras de débito e crédito de moedas virtuais entre pessoas. Seguindo o crescimento do mercado de moedas digitais, a Blockchain evolui como suporte para realizar transações em qualquer contexto, não somente financeiros. Nesse sentido, a tecnologia denominada de segunda geração, permite a inserção de funcionalidades (contratos inteligentes), que visam o cumprimento das normas de negócios a serem aplicadas nessas transações. A seguir serão detalhadas as duas gerações.

2.4.1. O mercado de moedas digitais - 1ª geração

Os primeiros estudos sobres moedas digitais datam do início da década de 90 com o movimento *Cypherpunk* [20]. Criptógrafos eram os principais integrantes deste movimento que lutou pela liberdade de ação dentro da internet desde seu início. Neste período surgiram os primeiros projetos de moedas digitais utilizando criptografia como base, entretanto estes projetos não conseguiram atingir um grande público e foram descontinuados por problemas de segurança.

Em 2008, Nakamoto [3] propõe o uso da estrutura Blockchain como base para um sistema de intercâmbio de dinheiro eletrônico, assim nascia o Bitcoin. Criado em 2009, o Bitcoin foi a primeira moeda digital a utilizar Blockchain para fins monetários em larga escala. Utilizado com o objetivo de manter a confiança entre as partes, a principal função da Blockchain no Bitcoin é armazenar, validar e distribuir as transações de valores realizada entre as mesmas.

Um desafio que as moedas digitais enfrentaram foi o problema conhecido na computação como gasto duplo. Este problema aborda a dificuldade de se garantir que um certo dado digital seja multiplicado de maneira indiscriminada; em outras palavras, evita que uma mesma transação seja utilizada mais de uma vez. Servindo como base para transações financeiras, a Blockchain do Bitcoin conseguiu resolver este problema empregando os conceitos mencionados anteriormente no capítulo de fundamentos e explorados na Seção 2.5.8. Este foi um dos principais motivos da criptomoeda conseguir ganhar a confiança dos usuários e assim expandir sua utilização ao redor do mundo.

A moeda digital Bitcoin pode ser considerada uma aplicação que funciona sobre a Blockchain. Em seu protocolo está descrito todas as funcionalidades e regras, entre elas o número máximo de moedas a ser criado: 21 milhões de unidades. Para entender como são criadas estas moedas é necessário falar sobre o procedimento denominado mineração.

Definido por Nakamoto como *Proof of Work* (PoW ou prova de trabalho), a mineração se dá pela resolução de um problema matemático com alto custo de processamento. O computador responsável pela resolução deste problema (denominado minerador) recebe moedas como recompensa, além de ganhar o direito de criar um novo bloco, que contém as transações. Além da criação, também é uma função do minerador a de validar estas transações e verificar se todos os usuários possuem os saldos transferidos.

Em seu início, a recompensa financeira pela resolução da prova de trabalho era de 50 Bitcoins, cujo valor cai pela metade a cada 4 anos até se esgotar a quantidade de total de moedas proposta. Estima-se que esta quantidade máxima seja atingida no ano de 2140.

No Bitcoin, para armazenar e movimentar um saldo é necessário que os usuários utilizem uma carteira digital (denominada *wallet*). As carteiras digitais podem ser *softwares* para computadores, aplicativos para dispositivos móveis ou *hardware* e sua principal função é armazenar uma chave privada que será utilizada para assinar as transações. Além da chave privada a carteira também armazena chaves públicas, estas chaves são o endereço de recebimento que o usuário deverá utilizar para solicitar uma transação a outro usuário.

Antes de ser validada e inserida na Blockchain, uma transação em Bitcoin fica aguardando a resolução da prova de trabalho para que um novo bloco seja criado. Definido em seu protocolo, este tempo é aproximadamente 10 minutos. Após uma transação ser inserida na Blockchain, a cada novo bloco inserido esta transação recebe novas confirmações. No Bitcoin, quanto mais confirmações uma transação receber maior é a segurança de que esta não será revertida. Um dos motivos para uma transação ser revertida é quando dois ou mais blocos válidos são criados simultaneamente, mas apenas um destes blocos fará parte da cadeia de blocos. A importância da confirmação da transação será abordada na Seção 2.5.8.2.

2.4.2. Contratos inteligentes - 2ª geração

A Blockchain de primeira geração foi aplicada principalmente para executar transações financeiras, onde um valor é transferido de uma pessoa para outra. Nesse sentido, os computadores responsáveis por manter a coerência da Blockchain, isto é, de inserir informações nela, cuidam de que a pessoa que está transferindo o dinheiro realmente possua o saldo suficiente para realizar a transação. Um ponto importante a destacar é que essa funcionalidade de verificação está inserida de forma estática dentro de cada um dos computadores que cuidam da Blockchain.

Aproximadamente em 2015, surgem no cenário da Blockchain algumas ferramentas, como Ethereum [9] e Hyperledger [10], que permitem inserir novas funcionalidades de forma dinâmica. O interessante dessa abordagem é que as funcionalidades não precisam ser somente para verificar um saldo (no contexto financeiro), mas podem ser para qualquer funcionalidade de negócio, por exemplo, verificar se um lote de remédios atin-

giu a data de vencimento e lançar um alerta. Nasce assim a segunda geração da Blockchain, onde o foco está nas funcionalidades, também denominadas de contratos inteligentes (*smart contracts*, em inglês, proposto conceitualmente por Szabo em 1996 [21]).

O contrato inteligente é análogo a um contrato em papel firmado por pessoas. No contrato em papel, são definidas as regras que estabelecem as responsabilidades e comunicação entre as partes que a assinaram. Na Blockchain, a diferença é que o contrato é digital, porém são mantidos os mesmos preceitos do contrato em papel.

No contexto da computação, as regras que estabelecem as responsabilidades que devem ser realizadas pelas partes são denominadas de regras de negócios, ou funcionalidades do sistema. Uma regra de negócio, nesse contexto, conterá uma sequência lógica de passos que serão transformados e implementados em um código executável utilizando alguma linguagem de programação.

Para dar um exemplo mais concreto, suponha que, no contexto de um software de gerenciamento de um hospital, o diretor pede para criar uma funcionalidade que verifique se o lote de um determinado medicamento está vencido, alertando-o dessa situação. A sequência lógica de passos para essa funcionalidade seria:

1. Obter os lotes gerenciados pelo hospital;
2. Recuperar as informações do lote, dentre elas o nome do medicamento, a data de vencimento e o email do diretor;
3. Verificar se a data de vencimento é maior que a data atual;
4. Se for maior, enviar um email para o diretor e gerentes, avisando da situação.

Para alguém que trabalha com sistemas informatizados, a funcionalidade descrita acima será desenvolvida em alguma linguagem de programação (por exemplo, Python, Java, etc.) e implantada no software de gerenciamento do hospital. Então, imagine que o desenvolvedor, por alguma razão, modifica o passo 4 da funcionalidade com a seguinte regra: "*se a data de vencimento for maior, não envie o email ao diretor*". Após a modificação, a regra é implantada no sistema. Note que o diretor (no passo 4) nunca ficará sabendo do vencimento do lote, mesmo que inicialmente foi ele quem solicitou a funcionalidade. Nesse sentido, o contrato foi quebrado sem uma das partes, no caso, o diretor, ter ideia disso.

Eis onde entra o contrato inteligente. De acordo ao mencionado na Seção 2.3, uma das características de Blockchain é a imutabilidade, que permite que uma informação, uma vez inserida na Blockchain, não possa ser alterada. Nesse sentido, agora imagine que a funcionalidade de vencimento foi inicialmente implementada e inserida na Blockchain (ou seja, como se ela fosse uma transação). Note que a funcionalidade não poderá ser alterada a não ser que uma nova transação (com o novo código modificado) seja inserida na Blockchain. O importante é que o diretor poderá observar tanto a funcionalidade inicial quanto a modificada, podendo realizar a auditoria desta. Nesse sentido, a funcionalidade fica transparente para todas as partes que a utilizam.

Atualmente, existem mais de 2100 criptomoedas virtuais parecidas ao Bitcoin, onde dado o interesse pelo investimento, centenas delas apareceram em questão de meses [22].

2.5. Como funciona a Blockchain?

Nesta seção serão apresentadas as diferentes tecnologias que compõem a Blockchain e como estas interagem para permitir seu bom funcionamento. Para isso, primeiro será dada uma visão geral de como funciona. A seguir, será explorado o conceito de bloco, transação e de como a cadeia formada por estes formam a base da Blockchain. A partir daí, será explicado alguns conceitos de criptografia e *hash*, que permitiram aumentar a segurança e eficiência da Blockchain. Finalmente, será descrito o conceito de consenso, ou seja, de como os computadores que fazem parte da Blockchain podem chegar a um acordo de quais blocos são os válidos.

2.5.1. Visão geral

Nesta seção será construído passo a passo o funcionamento da Blockchain através de dois cenários, o primeiro no contexto de uma compra de um bem entre duas pessoas e o segundo no contexto da área da saúde.

No primeiro cenário, imagine uma pessoa que compra um determinado bem de outra através de um meio eletrônico, como um sítio Web. Para que ambas tenham certeza de que a transação financeira ocorreu, ou seja, que o dinheiro foi enviado por uma e recebido pela outra, será necessário que algum computador armazene essa transação.

No segundo cenário, imagine uma médica que atende um paciente e registra essa consulta utilizando um prontuário eletrônico. Para que ambas as pessoas possam visualizar essa informação, também será necessário que algum computador armazene o prontuário.

Na Figura 2.2 é possível visualizar conceitualmente um bloco, que dentro contém um cabeçalho e a transação. Note que o conceito de transação é bem geral, podendo ser tanto as informações de um movimento financeiro quanto às informações de um prontuário do paciente.

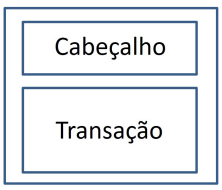


Figura 2.2. Bloco

Agora, o paciente se consulta uma segunda vez com a mesma médica. Após o atendimento, ela registra as informações no prontuário do paciente, gerando um novo bloco, que por sua vez contém uma nova transação. Como mostra a Figura 2.3, nesse novo bloco, a transação 2 fará parte do prontuário do paciente, com a alteração realizada pela nova consulta. Note que o segundo bloco aponta para o primeiro, criando realmente

uma cadeia interligada de blocos.

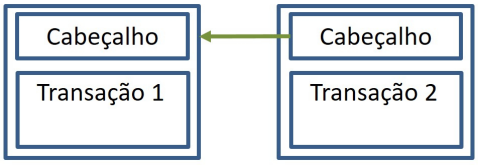


Figura 2.3. Cadeia de Blocos

Nesse momento é necessário fazer duas perguntas: (1) o que acontece se o computador que registrou as transações não tivesse mais acesso à internet ou se seu disco rígido onde estavam armazenadas as transações queimasse? (2) o que acontece que se alguma pessoa, que não tem permissão, modifica as informações do prontuário?

No primeiro ponto, há um problema de disponibilidade da informação. Ou seja, pode ser que as transações se percam e não consigam ser recuperadas. No segundo ponto, a informação está disponível, porém não está íntegra, ou seja, não é confiável.

Para resolver essas questões, uma das alternativas seria replicar a cadeia de blocos em diversos computadores. Assim, se um computador ficar indisponível, outro computador poderia tomar seu lugar. Já no caso de uma informação modificada, as outras réplicas poderiam verificar se houve alguma fraude e tentar chegar a um consenso. Esses casos serão analisados na Seção 2.5.8.

Assim que um computador tiver os blocos, será necessário que este seja capaz de analisar se os blocos e as transações contidas nos blocos, são válidos ou não. Nesse sentido, quais seriam as informações que cada bloco deveria ter para realizar a validação?

2.5.2. Bloco e cadeia de blocos

Como mencionado, a Blockchain é composta por uma cadeia interligada de blocos, que por sua vez contém uma ou mais transações. Agora, faz-se necessário entender quais são as informações que compõem o bloco.

O bloco é uma estrutura composta por dois módulos: cabeçalho e a lista de transações. O cabeçalho consiste em diversos metadados que identificam unicamente o bloco. Já a lista de transações identificam as transações realizadas e contidas nesse bloco. Por simplicidade, nesse texto, representamos a lista de transações sempre com apenas uma transação, mas lembramos que as listas podem conter dezenas ou centenas de transações.

Na Tabela 2.1 pode-se observar os principais campos que determinam o cabeçalho de um bloco junto com sua funcionalidade. Os campos *Merkle Root*, *Difficulty Target* e *Nonce* serão explicados com mais detalhes nas Seções 2.5.5, 2.5.6 e 2.5.8, respectivamente.

Com as informações da tabela, vamos criar os blocos do segundo cenário. Imagine que no primeiro atendimento, realizado às 8.30, um bloco 1 foi criado com uma transação (no caso, o prontuário eletrônico do primeiro atendimento). No segundo atendimento, realizado às 19.30 do mesmo dia, um novo bloco foi criado com a segunda transação (no caso, a adição de um novo prontuário eletrônico). Na Figura 2.4 pode-se observar

Tabela 2.1. Cabeçalho do Bloco.

Nome	Funcionalidade
Previous Block Hash	Apontador para o cabeçalho do bloco anterior.
Merkle Root	Número único que determina as transações que existem no bloco.
Timestamp	Data e hora aproximada da criação do bloco
Difficulty Target	Nível de dificuldade na criação do bloco
Nonce	Número que determina como foi criado o bloco

algumas informações do cabeçalho que o sistema gerou para esse segundo bloco 2.

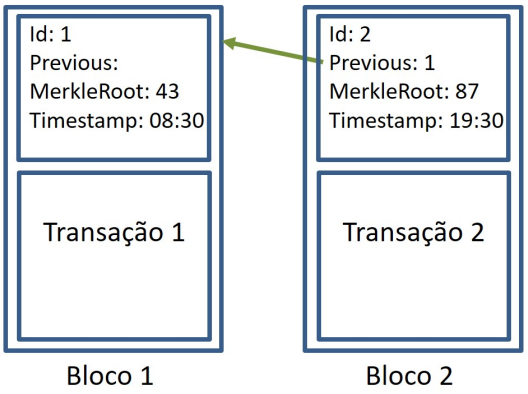


Figura 2.4. Cadeia de Blocos com cabeçalho

No bloco 2, o valor do *Previous Block Hash* corresponderá ao valor que identifica unicamente o bloco 1 (na Seção 2.5.5 será visto como é criado esse valor). Além disso, o sistema gerará um valor que determinará as transações que existem no bloco (Merkle Root) e a hora em que foi criado o bloco, isto é às 19.30 (timestamp).

Nesse momento deve surgir uma questão. Para quem aponta o bloco 1 no seu campo *Previous Block Hash*? Intuitivamente, deve apontar para um bloco anterior. Porém, deve existir algum bloco que não aponte para um bloco anterior, representando assim o começo da cadeia. Esse bloco inicial é chamado de bloco gênesis.

Em um sistema Blockchain, o primeiro bloco da cadeia é denominado de bloco gênesis e todos os computadores que façam uso da Blockchain devem conhecê-lo. Nesse sentido, se a partir de qualquer bloco se retrocede para o anterior, e deste para o anterior, utilizando o valor do campo *Previous Block Hash*, finalmente chegará ao bloco gênesis.

Até aqui, o exemplo do segundo cenário poderá ser complementado com o bloco gênesis, como mostra a Figura 2.5. Mais informações técnicas sobre o bloco podem ser encontradas em [23].

2.5.3. Transação e cadeia de transações

Dentro de cada bloco estão inseridas as transações realizadas pelos usuários do sistema. Um transação permite mostrar que um determinado item (seja este um valor monetário,

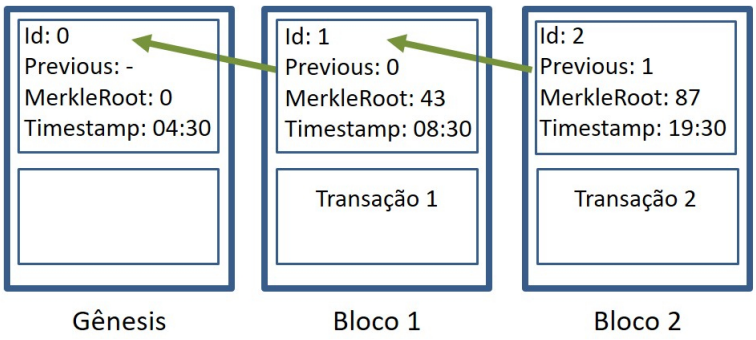


Figura 2.5. Cadeia de Blocos com gênesis

um documento, uma permissão de acesso, etc.) foi autorizado por um certo dono e disponibilizado para outro.

Conceitualmente, uma transação pode ser enxergada como um evento que ocorreu no sistema. Nesse sentido, no primeiro cenário, o evento seria uma transação financeira entre duas pessoas. Já no segundo cenário, o evento seria a inserção do registro eletrônico do paciente e a permissão de visualização da médica para o paciente.

De forma geral, uma transação é composta por um identificador único da transação e dois módulos: entradas e saídas. A entrada representa o identificador do dono que está realizando a transação e a saída representa o identificador do dono que está recebendo a transação. A Figura 2.6 abaixo mostra como seria a transação de uma transferência financeira de 5 unidades da pessoa com identificador ID1 para uma com identificador ID2.

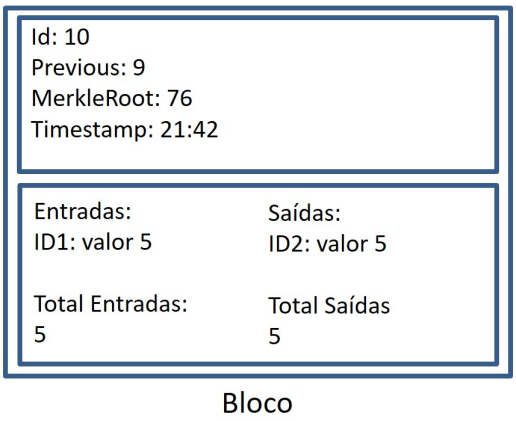


Figura 2.6. Transação

Entretanto, como saber se ID1 realmente tinha posse do item transferido (no exemplo acima, as 5 unidades)? Nesse sentido, ao igual que os blocos, as transações precisam apontar para uma transação válida (isto é, já existente em algum bloco da cadeia de blocos). Com isso, caso alguém queira verificar se ID1 tem ou não o item, basta analisar todas as transações realizadas por esse identificador, utilizando para isso os apontadores das transações. Mais informações técnicas sobre as transações podem ser encontradas

em [24].

2.5.4. Chave pública e privada

Até aqui, sabemos que a Blockchain é composta por blocos interligados, que por sua vez é composto por uma ou mais transações também interligadas. Também foi mencionado que a transação contém identificadores de usuários do sistema que a autorizaram e efetivaram. Por outro lado, como um computador pode validar que o identificador realmente é da pessoa que fez a transação e não de alguém se passando por ela?

Para explicar essa validação, será necessário entender antes alguns conceitos advindos da área de criptografia, cuja responsabilidade, entre outras, é a de segurança da informação. A seguir serão explicados esses conceitos através de um exemplo.

Imagine que você tem um texto que precisa enviar para alguém usando algum meio eletrônico, como um e-mail ou uma aplicação de bate papo. No envio, esse texto poderá passar por diversos computadores intermediários até chegar ao destino (é o que acontece normalmente na Internet). Para evitar intromissões, será necessário codificá-lo na origem de alguma forma, para que os intermediários não possam saber o que diz a mensagem, e decodificá-lo no destino, para recuperar a mensagem.

Na criptografia, a codificação na origem é denominada encriptar e a decodificação no destino é denominada de decriptar. Já o texto codificado, que não possui nenhum significado para quem não souber decodificá-lo, é denominado de "texto cifrado". A Figura 2.7 mostra os conceitos aplicados ao texto "Paciente: João. Tuberculose: negativo".

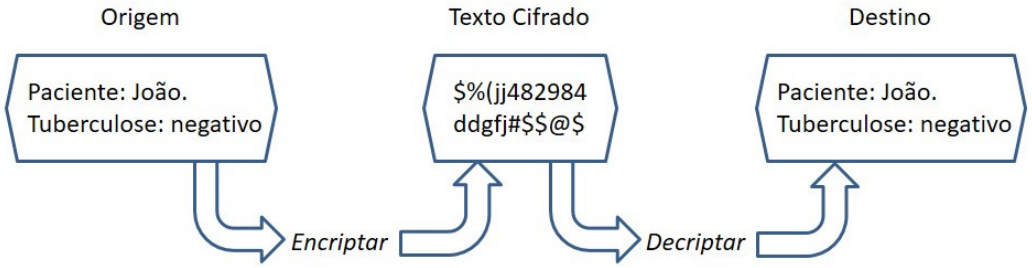


Figura 2.7. Aplicação da Criptografia

Agora, como é possível codificar o texto? Para isso, a criptografia utiliza um código secreto, denominado de chave (pense nela como se fosse o PIN do seu cartão de crédito) e o uso de um mecanismo de codificação. Dentro do mecanismo, existem duas alternativas a simétrica e a assimétrica. Na simétrica, se utiliza o mesmo código tanto para a codificação quanto para a decodificação, assim, origem e destino precisarão conhecer o mesmo código. Na assimétrica, se utilizam dois códigos diferentes, um para a codificação e outro para a decodificação. A seguir veremos o segundo caso, que é a alternativa utilizada pela maioria das Blockchains.

Na criptografia assimétrica existem duas chaves, a pública e a privada, que estão relacionadas entre si. O mais interessante dessa abordagem é a propriedade de que o texto encriptado por uma chave, somente pode ser decriptado pela outra. Como exemplo, veja o fluxo mencionado na Figura 2.8. Na figura, o texto "Paciente: João. Tuberculose:

negativo" é encriptado com uma chave privada (cor verde), transformando-o em um texto cifrado. A seguir, o texto cifrado é decriptado por uma chave pública (chave vermelha), recuperando o texto inicial.

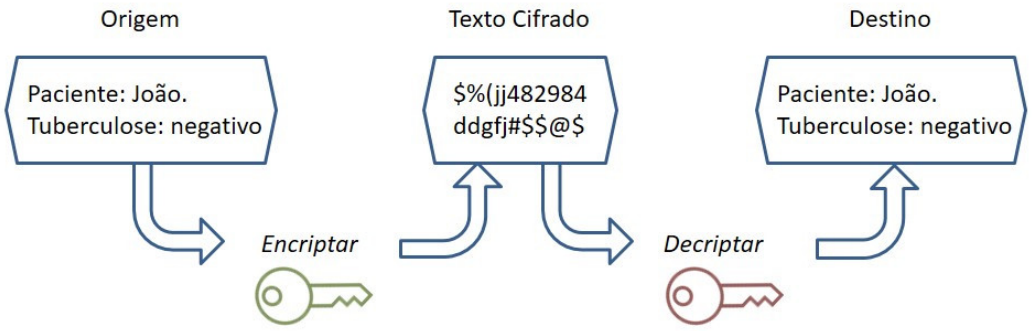


Figura 2.8. Fluxo de criptografia assimétrica

Na Figura 2.9 pode-se observar o fluxo inverso, onde o texto "Paciente: João. Tuberculose: negativo" é encriptado com uma chave pública (cor vermelha), transformando-o em um texto cifrado. A seguir, o texto cifrado é decriptado por uma chave privada (cor verde), recuperando o texto inicial.

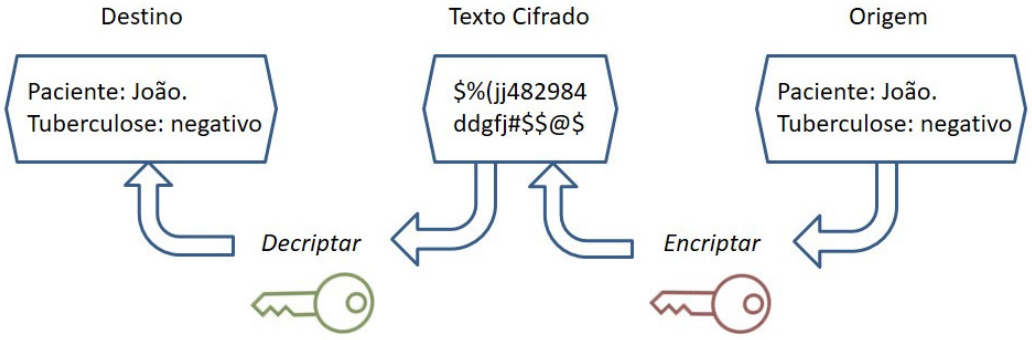


Figura 2.9. Fluxo inverso de criptografia assimétrica

Um ponto importante é que a chave privada permite identificar unicamente o dono emissor da mensagem, já que é a única pessoa que deveria ter posse dela (nesse sentido, essa chave não deveria ser compartilhada com ninguém). Do outro lado, o receptor da mensagem poderá identificar que essa mensagem realmente corresponde ao dono, dado que a única forma de decriptá-lo é utilizando a chave pública relacionada com essa chave privada. Mais informações técnicas sobre a criptografia podem ser encontradas em [25].

Finalmente, cabe mencionar que a geração da chave privada, e sua correspondente chave pública, deverão ser realizadas por uma entidade reconhecida, com a qual os usuários do sistema tenham confiança. Na vida real, por exemplo, diversas empresas realizam essa atividade, como a Verisign, Digicert, entre outros. Já na Blockchain, existe um componente de software que realiza essa ação, como veremos mais à frente.

2.5.5. Função de Hash

Como mencionado até agora, um usuário pode criar uma transação utilizando sua chave privada, inseri-la dentro de um bloco, concatená-la com um bloco anterior e replicá-la em outros computadores. Os outros computadores, por sua vez, obterão esses blocos e verificarão se a informação contida é válida ou não.

Entretanto, note que quanto mais transações hajam, mais informação precisará ser validada, chegando a um ponto em que precisará-se muito tempo para realizar essa ação. Não haveria uma forma de validar se a informação está correta somente comparando dois números, independente do tamanho da informação?

Novamente a criptografia entra em ação, provendo um mecanismo de compactação da informação que possui as seguintes características: determinístico, evita conflitos e de uma via [26].

Para entender essas características, imagine que precisamos compactar inicialmente o texto "Bloco 101 com 1 transação advinda do usuário João". Vamos supor que o mecanismo compactou o texto para "B101-1-J". O determinismo está relacionado com o fato de que a compactação do texto inicial sempre terá como resultado o mesmo valor "B101-1-J".

Já o evite de conflitos, denominado de colisão, está relacionado com que diferentes textos não deveriam ter como resultado o mesmo valor. Assim, por exemplo o texto "Bloco 101 com 1 transação advinda do usuário Joã" (sem a vogal 'o' no final) deveria criar um texto compactado completamente diferente.

Finalmente, a característica de uma via, permite que não seja possível recuperar o texto original a partir das informações do texto compactado. Nesse sentido, "B101-1-J" não cumpriria essa condição, pois alguém poderia entender que 'B101' corresponde ao número do bloco, 1 corresponde à quantidade de transações e 'J' corresponde ao nome de algum usuário.

O mecanismo geralmente utilizado pela Blockchain para compactar a informação é denominado de *hash*, que permite compactar um texto de qualquer tamanho em outro de tamanho fixo. Existem diversas implementações que realizam a compactação, como por exemplo o MD5, SHA1, entre outras. A seguir será dado um exemplo com MD5, para entender as características mencionadas anteriormente.

Para o texto "Bloco 101 com 1 transação advinda do usuário João" (sem aspas), o MD5 o compacta no seguinte código: 0F1D18186FD5E1C9072627CC9677446E. Independente de quantas vezes aplicar o MD5 no texto, será obtido o mesmo código anterior, corroborando o determinismo. Agora, para o texto "Bloco 101 com 1 transação advinda do usuário Joã" (sem aspas e sem a vogal 'o'), o código obtido será bem diferente: 8899FA17AE7A802024D96E101C85B0FC. Veja que, mesmo com um pequena alteração, ele é completamente diferente do anterior, corroborando a ideia de evitar conflitos ou colisões. Finalmente, note que para o código obtido, não há sequer uma dica de como obter o texto antes de ser compactado, corroborando a característica de uma via. Mais informações técnicas sobre as funções de *hash* podem ser encontradas em [27, 25].

2.5.6. Árvores de Merkle

Uma árvore na computação pode ser descrita como uma estrutura composta por um conjunto de nós ligados, que começam em um nó (denominado raiz) e terminam em um ou mais nós (denominados folhas). O mais interessante dessa estrutura é que para chegar do nó raiz até uma folha, somente haverá um caminho possível de ser percorrido. Além disso, cada nó não folha (denominado pai) poderá ter um ou mais nós (denominados filhos) A Figura 2.10(a) mostra uma árvore numerada desde o nó raiz (com o número 1), onde cada nó tem dois possíveis filhos (direita e esquerda) até os 4 nós folha (com os números 4 ao 7).

(a) Árvore. (b) Árvore de Merkle.

Figura 2.10. Exemplos de Árvores.

A árvore de Merkle [28] é uma árvore bastante utilizada na Blockchain para conferir que as transações inseridas dentro de um bloco são válidas [3]. Antes de entender como a Blockchain usa a árvore de Merkle, vamos analisar como esta é construída. Imagine que um bloco possui 4 transações, cada um com seu identificador único. Como mostra a Figura 2.10(b), primeiro, de cada identificador da transação será obtido o valor *hash* e adicionado à folha correspondente. Assim, a folha 4 terá um valor v_4 calculado com a função H(ID1). Já a folha 5 terá um valor v_5 calculado com H(ID2), onde H é uma função *hash*, como MD5. A seguir, cria-se o valor *hash* da concatenação de pares de transações e insere-se esse valor no nó pai destas. Para o exemplo, o nó 2 terá um valor v_2 calculado com a função $H(v_4v_5)$. O mesmo para o caso do nó 3 que terá um valor v_3, calculado com a função H para as transações ID3 e ID4. Finalmente, o nó raiz terá o valor v_1 através do *hash* da concatenação dos valores nos nós 2 e 3 (*i.e.*, $H(v_2v_3)$), formando assim a árvore de Merkle.

Como mencionado na seção anterior, sabemos que o cálculo do *hash* possui certas propriedades que permite que o cálculo do valor, usando a função, seja determinístico. No caso da árvore, sempre teremos o mesmo valor da raiz se começarmos com os mesmos valores dos identificadores nas folhas. Note que se qualquer transação das folhas tiver outro identificador, o nó raiz também terá outro valor.

Agora, onde a Blockchain utiliza a árvore de Merkle? Como mencionado na Tabela 2.1, um dos campos do cabeçalho que deve ser preenchido na criação do bloco é o Merkle Root, que determina as transações que existem no bloco. Note que o valor v_1 do

nó raiz da Figura 2.10(b), representa todas as transações, por ser uma combinação delas. Assim, na criação do bloco, o Merkle Root do cabeçalho será preenchido com o valor existente na raiz da árvore de Merkle.

2.5.7. Tipos de Blockchain

Segundo Buterin [29], criador do Ethereum, existem dois tipos de Blockchains: permissionadas e não permissionadas. Nas Blockchain não permissionadas, qualquer membro pode realizar modificações e auditar a cadeia. Já nas Blockchain permissionadas, somente membros autorizados podem realizar operações na cadeia. Além disso, é comum associar o termo Blockchain não permissionada a Blockchain pública e Blockhain permissionada a instâncias privadas, federadas ou em consórcio.

Na Blockchain não permissionada, qualquer entidade pode entrar e sair do sistema em qualquer momento. Ao entrar, a entidade transforma-se em um membro que poderá realizar modificações e auditorias na cadeia inteira de blocos. Como é possível que potencialmente cada membro possua a cadeia de blocos, nesse tipo de Blockchain há uma total descentralização da informação. Exemplos deles são Bitcoin [3] e Ethereum [9].

Na Blockchain permissionada, somente algumas entidades serão transformadas em membros do sistema e terão permissão para realizar operações na cadeia. Assim, algumas poderão ler os blocos, outras poderão escrever e outras poderão auditar. Para permitir a identificação e autorização dos membros, será necessário criar responsáveis (confiáveis) por gerenciar as permissões. Nesse tipo de Blockchain existem entidades que realizam o papel de autorizadores. Exemplos de Blockchain permissionada são as plataformas Hyperledger Fabric [30] e Corda [31].

Wüst e Gervais [32] propuseram uma subdivisão da Blockchain permissionada entre pública e privada. A divisão somente considera a auditabilidade, onde a *"Blockchain permissionada pública"* permite que qualquer membro possa verificar os dados da cadeia e na *"Blockchain permissionada privada"* somente é permitida a verificação para um conjunto bem definido e autorizado de membros.

As aplicações Blockchain que requerem identificação de usuários tendem a ser construídas usando infraestrutura permissionada. Por outro lado, estruturas não permissionadas tendem a oferecer maior anonimato. Outra questão importante para a escolha de tipo de Blockchain é a criação e manutenção da infraestrutura que suporta a rede de nós. Uma Blockchain privada normalmente é de responsabilidade de uma instituição que a mantém operante; nesse sentido, as Blockchains públicas ou federadas concentram menos o poder de decisão sobre a rede.

2.5.8. Consenso

Lembrando que uma cadeia de blocos é replicada em diferentes computadores por questões de disponibilidade (caso um dos computadores saia do sistema, outros poderão tomar seu lugar), veja a seguinte situação que pode acontecer.

Imagine que em um primeiro momento os três computadores A, B e C da Figura 2.11 possuem a mesma cadeia de blocos 0 e 1.

Como mostra a Figura 2.12(a), em um segundo momento, o computador D envia

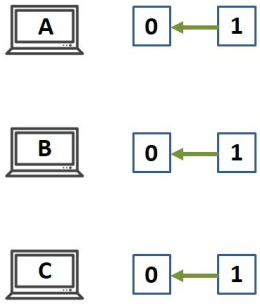
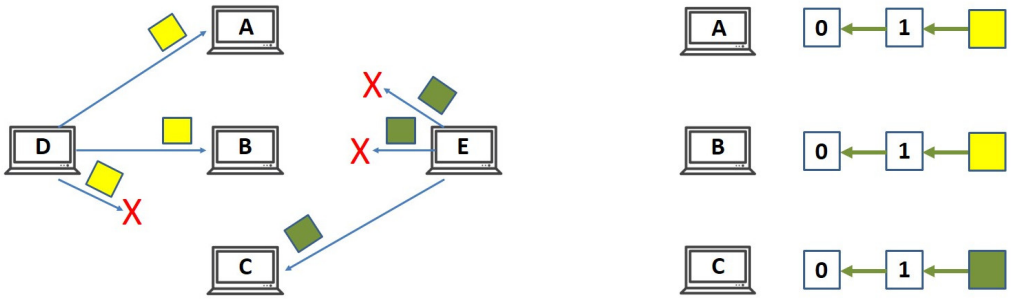


Figura 2.11. Consenso inicial

uma mensagem para que A, B e C adicionem o bloco amarelo, porém somente A e B a recebem. Em seguida, o computador E também envia a mensagem para que A, B e C adicionem o bloco verde, porém somente C a recebe. Uma pergunta que pode surgir é, porque algumas mensagens não foram recebidas? Na Internet, muitas vezes as mensagens são perdidas, principalmente, por questões de congestionamento nos roteadores por onde passa a mensagem até chegar a seu destino. No final do segundo momento, pode-se observar na Figura 2.12(b) que os computadores A e B possuem o bloco amarelo e o computador C possui o bloco verde, ambos apontando para o bloco 1 (portanto, tanto o bloco verde quanto o amarelo são válidos).



(a) Recebendo blocos. (b) Adicionando blocos.

Figura 2.12. Consenso intermediário.

O consenso tem como objetivo que os computadores cheguem a um acordo sobre um determinado valor [27]. No caso da Blockchain, eles devem chegar a um acordo de qual bloco (verde ou amarelo) deverá ser adicionado no final da cadeia, ou seja após o bloco 1. No final do acordo, todos os computadores (no exemplo, A, B e C) deverão ter a mesma cadeia. A seguir veremos duas estratégias empregadas pela Blockchain para chegar ao consenso.

2.5.8.1. Processo de consenso via Paxos

No consenso via Paxos [33], somente alguns poucos computadores são os encarregados por realizar o processo de acordo, geralmente algumas dezenas destes. A escolha desses

computadores pode-se basear em diferentes características, tais como, poder computacional, tempo sem interrupções, largura de banda, entre outros. Para o exemplo, vamos supor que foram escolhidos 7 computadores.

Tendo os computadores que farão o consenso, o primeiro passo do processo é a escolha, dentre eles, de um líder, ou seja, um computador responsável por dirimir qual será o bloco a ser inserido no final da cadeia. Existem diversas alternativas para realizar a escolha, por exemplo, cada computador pode ter um número identificador único e o menor destes será escolhido como o líder. Caso os outros seis computadores do consenso enxerguem que o líder não está mais disponível (*e.g.* caiu), o computador com o segundo menor identificador será escolhido como o novo líder, e assim sucessivamente.

Tendo o líder, cada computador que não faz parte do consenso (denominado de cliente) pode propor um novo bloco para ser inserido na cadeia. O cliente pode propor a inserção de um bloco a qualquer dos sete do consenso, porém somente o líder poderá inseri-lo na cadeia. Nesse sentido, o computador do consenso que recebeu o bloco, redirecionará o pedido para o líder, caso não o seja.

O líder, por sua vez, receberá os pedidos, direto dos clientes ou dos redirecionamento, e dará uma ordem neles (geralmente, o primeiro pedido que chega é o primeiro pedido a ser atendido). Após a ordenação dos pedidos, adicionará os blocos nessa ordem, e replicará essa informação para os outros seis computadores do consenso.

Quando os computadores do consenso receberem a ordem dos blocos, inserirão nas suas respectivas cadeias, respondendo ao líder que conseguiram realizar a inserção. Finalmente, assim que o líder obtiver a maioria das respostas (por maioria, entenda-se à metade mais um, ou seja, quatro respostas), responderá ao computador cliente que seu bloco foi adicionado com sucesso.

Os detalhes do funcionamento do processo serão analisados na Seção 2.10.1 sobre o Hyperledger.

2.5.8.2. Processo de consenso via Proof of Work (POW)

Neste tipo de consenso, todos os computadores podem ser passíveis por realizarem o processo de consenso, diferente do Paxos onde somente alguns são os escolhidos. Para comportar os possíveis milhares de computadores, a escolha de um líder não é a mais adequada, haja vista que esse líder talvez não será capaz de lidar com todas as mensagens advindas de todos os computadores. Assim, será necessário criar um processo que não dependa somente de um computador.

A Blockchain utilizada no Bitcoin foi uma das primeiras técnicas, aplicadas em um sistema real, em possibilitar o consenso em milhares de computadores. O funcionamento dele é dado a seguir.

O processo começa com um computador (denominado minerador) obtendo de outro computador a cadeia de todos os blocos, com suas respectivas transações, que existe até esse momento. Para se ter uma ideia, o histórico em 2019 possui milhares de blocos, com um tamanho aproximado de 150 gigabytes. Para o exemplo, imagine que a cadeia

tem somente 100 blocos.

Em posse desse histórico, o minerador, que será denominado de M1, deve verificar que cada uma das transações é válida (utilizando o conceito de cadeia de transações mencionado na Seção 2.5.3) e que cada bloco é válido (utilizando o conceito de cadeia de blocos mencionado na Seção 2.5.2).

Tendo realizado as validações, o minerador M1 obterá novas transações que foram realizadas pelos clientes (e que não existem em nenhum bloco anterior), criando um novo bloco com essas transações.

Na criação do bloco é necessário preencher as informações do cabeçalho do mesmo. Primeiro, o campo 'timestamp' corresponde à hora do computador do minerador, por exemplo, 12/02/2019 13:30. A seguir, o campo '*Previous Block Hash*' deverá apontar para o bloco 100, calculado através do *hash* desse bloco 100, por exemplo usando o MD5 explicado na Seção 2.5.5. '*Difficulty Target*' e 'nonce' são números que o minerador deverá utilizar para provar aos demais mineradores que realmente foi realizado um trabalho computacional para criar o bloco.

O trabalho é realizado da seguinte maneira: '*Difficulty Target*' é um número calculado pelo sistema e gerado aproximadamente a cada duas semanas. Esse número, que permite que cada duas semanas sejam criados no máximo 2016 blocos, geralmente começa com uma quantidade de zeros, por exemplo: 000101827749837 (começando com 3 zeros). A seguir, o minerador precisará encontrar um número menor que o '*Difficulty Target*', obtido através do *hash* do bloco que está sendo criado. Porém, pense o seguinte, só com as informações do cabeçalho (timestamp, previous block *hash* e *Difficulty Target*) pode ser que o MD5 não consiga gerar um valor menor que o *Difficulty Target*'. Por exemplo, vamos supor que o MD5 do texto '12/02/2019 13:30 bloco100 000101827749837' dê o valor 100405682589837. Note que esse valor é maior ao 0001018277498372371. Eis onde entra o nonce. O nonce é um atributo do cabeçalho que permite ser modificado para que o *hash* seja menor ao *Difficulty Target*. Veja o exemplo na Tabela 2.2 abaixo para diferentes nonces, aplicados ao cabeçalho anterior.

Tabela 2.2. Aplicação de diferentes Nonces.

Nonce	Texto a ser usado no MD5	Resultado
0	'12/02/2019 13:30 bloco100 000101827749837 0'	100405682589837
1	'12/02/2019 13:30 bloco100 000101827749837 1'	320106680549244
2	'12/02/2019 13:30 bloco100 000101827749837 2'	000047479763563

Note que o computador teve que realizar três cálculos (trabalho computacional com os nonces 0, 1 e 2) para encontrar um número menor que o *Difficulty Target*. Nos sistemas reais de Blockchain, como Bitcoin, o computador realiza milhões ou bilhões de cálculos, dai o nome *Proof of Work*, ou prova de trabalho.

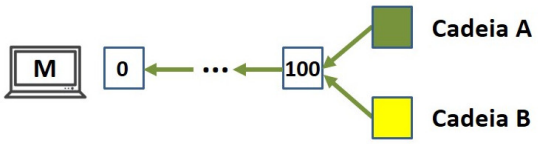
Após encontrar o nonce adequado, o minerador M1 o insere no cabeçalho e cria o bloco (denominemos esse bloco de amarelo, para efeitos ilustrativos). Após a criação do bloco amarelo, o minerador M1 dissemina essa informação a outros mineradores, denominado M. O consenso acontecerá em dois casos: (1) o minerador M que recebeu o bloco

amarelo de M1 também estava tentando criá-lo, e (2) o minerador M já tinha recebido um bloco verde, de outro minerador M2, que apontava para o bloco 100.

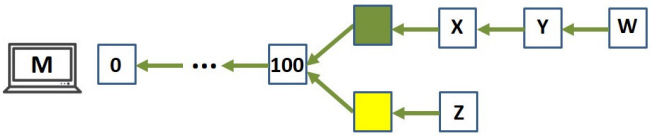
Para o primeiro caso, assim que o minerador M receber o bloco amarelo, imediatamente deverá verificar que o bloco recebido é válido (olhando que aponta para o bloco 100, por exemplo). Caso seja válido, M parará de criar seu bloco e adicionará o bloco amarelo no final da sua cadeia, começando a criação de um novo bloco, apontando para o amarelo.

Para o segundo caso, o minerador M tinha recebido um bloco verde válido que apontava para o bloco 100. Mas, como pode ter acontecido isso se o minerador M1 criou o bloco amarelo, também válido, nesse instante? Note que a criação de um bloco verde pode ter acontecido por um outro minerador M2 (nada impede isso) e ter sido disseminado antes que o bloco amarelo de M1.

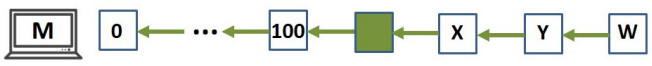
Agora, como mostra a Figura 2.13(a), M terá dois blocos válidos, um verde e um amarelo, ambos apontando para o bloco 100. O que fazer? A regra para chegar ao consenso será esperar a chegada de novos blocos e, depois de um certo tempo, escolher aquele que tenha a maior cadeia a partir do bloco 100. Imagine o seguinte caso, após um certo tempo, M recebe três novos blocos que contém o bloco verde, denominada cadeia A, e somente um novo bloco que contém o bloco amarelo, denominada cadeia B, como mostra a Figura 2.13(b). Finalmente, como a cadeia A é maior que a cadeia B, o minerador M descartará a cadeia B (que contém o bloco amarelo e o bloco Z), como mostra a Figura 2.13(c). Note que o consenso ocorrerá dado que a regra de somente continuar com a maior cadeia será seguida por qualquer minerador (inclusive M1).



(a) Duas cadeias.



(b) Recebendo blocos.



(c) Escolhendo a cadeia mais longa.

Figura 2.13. Consenso de qual bloco inserir.

Os detalhes do funcionamento do processo serão analisados na Seção 2.10.2.

2.6. Desafios para o uso da Blockchain

Zhang *et al.* [15] descrevem alguns desafios para a implantação de sistemas de informação baseados em Blockchain: *i.* capacidade de evoluir, *ii.* capacidade de armazenamento, *iii.* privacidade e *iv.* escalabilidade. Além desses, serão tratados alguns outros desafios como a interoperabilidade, a compatibilidade legal dos sistemas e o gasto de energia.

- A evolução é algo natural em sistemas de informação. Tanto a informação quanto as funcionalidades que as tratam sofrem alterações durante a vida do sistema, sejam por questões do projeto em si ou até por questões normativas (leis) que devem ser seguidas. Por exemplo, imagine que em um determinado momento todos os prontuários dos pacientes do SUS, armazenados na Blockchain, sejam identificados pelo CPF. Anos depois, cria-se uma lei que obriga aos sistemas a usarem, como identificador único, o número do cartão do SUS.

  A capacidade de evoluir refere-se ao suporte que entrega a ferramenta ou tecnologia utilizada para facilitar a evolução dos sistemas, diminuindo ao máximo as mudanças que devem ser realizadas. Note que no caso da Blockchain, todas as transações antes da aplicação da lei foram feitas usando o CPF. Nesse sentido, se quisermos adequar o sistema à nova lei, ou mantém-se na Blockchain a mistura de informações com CPF e o nº do SUS (lidando com possíveis inconsistências) ou elimina-se a Blockchain inteira e inserem-se novamente todas as transações que tinham o CPF, mas agora com o nº do SUS. Nesse sentido, é necessário ter extremo cuidado na escolha dos dados que vão compor a transação, para que seja flexível o suficiente para lidar com mudanças.

- O armazenamento permite a persistência da Blockchain nos computadores. Um sistema de saúde deve considerar que a Blockchain conterá informações sobre pacientes, médicos, prontuários, pagamentos, medicamentos, etc. Nesse sentido, grandes volumes de dados deverão ser armazenados nos computadores. Por outro lado, além do armazenamento, será necessário também considerar a infraestrutura de hardware (computadores, acesso à Internet, etc) e de software (sistema operacional, memória RAM, etc) que permitirá o acesso à Blockchain.

  A capacidade de armazenamento refere-se ao suporte que entrega a ferramenta ou tecnologia utilizada para facilitar o armazenamento da Blockchain. Por exemplo, usando o Ethereum (analisada na Seção 2.10.2) é possível abstrair todos os problemas, já que a rede oficial dessa tecnologia provê a infraestrutura e acesso. Entretanto, para usá-la, é necessário comprar GAS (unidade de medida similar ao kilowatt/hora da eletricidade) para executar as transações, incorrendo em gastos que é necessário considerar. Já usando o Hyperledger (analisada na Seção 2.10.1) é possível criar a rede entre os participantes do sistema, evitando os gastos por transação do Ethereum. No entanto, a criação e manutenção da rede também incorrerá em gastos que devem ser considerados.

- A privacidade da informação é um ponto sensível em qualquer sistema. Para o caso dos sistemas de saúde, alguns requisitos em privacidade incluem: autenticação dos participantes; armazenamento seguro, baseado em princípios de criptografia; controle de acesso às informações. Mesmo para sistemas considerados seguros, a cada

dia novos ataques tentam encontrar suas vulnerabilidades. Na Blockchain existe um risco maior. Primeiro, as informações são replicadas em todos os computadores que fazem parte da rede. Nesse sentido, caso no futuro o processo de encriptação seja comprometido, potencialmente todos os registros poderão ser lidos por uma pessoa (note que não há como evitar isso, haja vista que a pessoa, que faz parte da rede, poderá ter armazenado a Blockchain completa). Segundo, as funcionalidades (contratos inteligentes) implementadas e implantadas são abertas e possíveis de serem auditadas. No entanto, isso também gera um problema, caso uma pessoa descubra um erro de segurança na programação da funcionalidade, realizando ataques em todos os computadores que a usem.

- A escalabilidade, no contexto da Blockchain para saúde, refere-se tanto à quantidade de transações que o sistema permite realizar quanto às pesquisas que consegue responder dentro de um período de tempo. Como mencionado até agora, transações podem ser operações financeiras, registro de prontuários eletrônicos, cadastro de pacientes, entre outros. Nesse sentido, a ferramenta ou tecnologia deve ser capaz de suportar possivelmente dezenas de milhares de transações ou pesquisas em um curto espaço de tempo. Das duas ferramentas analisadas na Seção 2.10, Ethereum conseguiu realizar 1.349.890 transações no dia 4 de janeiro de 2018 (aproximadamente 14 transações por segundo). Já no Hyperledger foi possível realizar 1.7 bilhão de transações por dia (aproximadamente 20 mil por segundo) com modificações na arquitetura [34], provendo a escalabilidade necessária para atender os requisitos dos sistemas de saúde.

- O tempo de confirmação de uma transação pode variar entre as diferentes tecnologias Blockchain. Este tempo é o intervalo entre a criação da transação até o momento que a mesma é inserida em um novo bloco e distribuída entre os participantes. Na Blockchain utilizada pelo Bitcoin este intervalo é de aproximadamente 10 minutos. Com as melhorias implementadas na Blockchain do Ethereum e Hyperledger Fabric, este tempo foi reduzido para aproximadamente 15 segundos e 1 segundo, respectivamente. Este tempo deve ser observado com muito cuidado visto que, dependendo do contexto a ser utilizado, esta demora pode inviabilizar a sua aplicação.

- A interoperabilidade, refere-se à habilidade dos diferentes sistemas de informação de se comunicarem, transferirem e usarem informações [35]. A interoperabilidade é dividida em dois níveis: funcional, focada na interação entre sistemas usando regras de negócios; semântica, focada na compreensão do significado dos conceitos envolvidos na transferência.

  Por exemplo, imagine que um paciente se atende em um hospital do estado onde mora, mas em uma viagem a outro estado se atende em uma clínica privada. A interoperabilidade funcional proverá que o sistema da clínica possa ter acesso ao prontuário armazenado no sistema do hospital, desde que cumpridos os requisitos de privacidade da informação. Já a interoperabilidade semântica permitiria à clínica entender um significado específico de uma frase utilizada no contexto do hospital.

  Atualmente, alguns padrões de interoperabilidade de dados médicos (como HL7 e

FHIR [36]) provem as bases para intercambiar informações entre sistemas, entretanto, a implementação desses padrões ainda não é amplamente utilizada.

- A compatibilidade legal refere-se à capacidade do sistema se adequar aos regulamentos e leis existentes ou que possam ser criados no futuro [37]. Note que esse desafio está muito interligado à capacidade de evolução do sistema. Um exemplo para esse desafio é o artigo 17 do Regulamento Geral sobre a Proteção de Dados na Europa, que define o direito ao apagamento dos dados (o "direito a ser esquecido"). Entretanto, a tecnologia Blockchain não permite se adequar a essa lei de forma fácil, haja vista que as informações não podem ser eliminadas.

- Manter uma rede Blockchain em funcionamento requer uma quantidade de participantes *online* para validar e distribuir as transações. O gasto de energia para manter estes computadores ligados é um desafio a ser analisado. Em geral, as Blockchains públicas e não permissionadas possuem uma grande quantidade de participantes e o consumo de energia pode não ser sustentável (a rede Bitcoin[3] e Ethereum[4] possuem cerca de 8 mil participantes). Bitcoin, por ser baseado em solução de problemas matemáticos por força bruta, tem a sua sustentabilidade energética contestada [38]: em 2018, a energia elétrica utilizada para mineração foi superior ao consumo da Irlanda. Nas Blockchains privadas, por exemplo o Hyperledger Fabric, o consumo de energia ainda existirá, mas não é uma preocupação haja vista que normalmente utilizam algoritmos de consenso bizantino (que são mais baratos computacionalmente) e são poucos os computadores responsáveis por ele.

2.7. Vulnerabilidades de segurança da Blockchain

Todo sistema de informação está sujeito a ataques de segurança. Entre os ataques mais comuns estão a tentativa de quebra de senhas (quebra de segredos criptográficos) ou ataques de disponibilidade (DDOS, *Distributed Denial of Service*, em inglês). Blockchain, por sua natureza distribuída, pode estar exposto a ataques adicionais [39]. Na Blockchain, os três problemas de segurança mais discutidos são:

- **Ataque de 51%**. Apesar de conhecido pelo nome de 51%, na verdade, esse ataque é caracterizado quando uma única entidade (ou um arranjo de membros atuando como uma única entidade) detém uma fatia expressiva, ou a maioria, do poder computacional. Em uma rede Bitcoin, por exemplo, se uma mesma entidade detivesse a maioria do poder computacional, essa entidade poderia influenciar ou manipular a formação da cadeia de blocos. Em outras palavras, poderia influenciar a formação da cadeia mais longa de blocos para permitir, maliciosamente, o cancelamento de transações ou de decisões de consenso.

- **Gasto Duplo**. A tecnologia não permite a existência de gasto duplo; mas é estatisticamente possível que uma transação seja cancelada uma hora após ter sido registrada em um bloco. Em vendas no varejo, por exemplo, o cancelamento tardio de uma transação pode levar, na prática, ao não pagamento de uma transação.

[3] Vide https://bitnodes.earn.com/

[4] Vide https://ethstats.net/

- **Perda da Chave Privada**. Um recurso digital na Blockchain só pode ser alterado ou transferido usando a chave privada que o gerou. A perda dessa chave privada leva a perda permanente do recurso. Na rede Bitcoin, por exemplo, se um usuário perde a sua chave privada, suas moedas não podem mais ser movimentadas; esses recursos estarão perdidos para sempre.

Além dessas três preocupações mais comuns colocadas acima, um sistema Blockchain pode estar sujeito a ataques realizados para revelar a identidade de usuários e dados sensíveis. Esses ataques podem ser baseados em estratégias em engenharia social, que cruzem, por exemplo, registros Blockchain e hábitos individuais. Outro ataque relevante em aplicações que envolvem criptomoedas é o roubo de moedas por meio da exploração de falhas dos programas.

Por fim, uma preocupação cada vez mais importante é a corretude dos contratos inteligentes. Cabe lembrar que assim como uma transação, um contrato inteligente não pode ser modificado após sua escrita nos blocos. Encontrar um erro em um contrato inteligente significa: perder os valores gastos para sua publicação original, a necessidade de lançar uma nova aplicação e a respectiva migração de usuários. A corretude dos contratos inteligentes passa não apenas pela verificação de que os requisitos funcionais estão corretamente implementados, mas também que os usuários não possam explorar brechas que comprometam a rede (por exemplo, laços infinitos[5]) ou que permitam o roubo de dados e de chaves criptográficas.

O teste e validação de programas sempre foi importante para a ciência da computação, mas com o advento de contratos inteligentes essa importância ficou ainda maior. Em alguns casos, a não corretude de um programa pode ser diretamente associado a perda de recursos econômicos.

2.8. Será que preciso usar a Blockchain?

Apesar da Blockchain ter se tornado uma tecnologia muito popular nas criptomoedas, cabe perguntar-se se ela é aplicável a qualquer cenário. Segundo uma pesquisa realizada em 2018 pela PricewaterhouseCoopers, 84% de 600 executivos de empresas das mais diversas áreas estão de alguma forma envolvidos ativamente com a Blockchain [40]. No entanto, será que realmente essas empresas precisam utilizar a Blockchain ou poderiam utilizar tecnologias já consolidadas (como banco de dados) para realizar as mesmas funcionalidades de forma mais simples e até mais eficiente?

A seguir, será analisada uma metodologia, baseada no estudo de Wust [32], que permite identificar se faz sentido utilizar a Blockchain. Caso seja identificada a necessidade, será mostrado qual é o tipo da Blockchain (permissionada ou não) que deveria ser utilizada.

A Figura 2.14 mostra o fluxograma da metodologia, onde a circunferência representa uma pergunta a ser respondida, cada linha representa a resposta (sim S, não N) à

[5] A presença de laços infinitos em contratos inteligentes obviamente comprometem a capacidade computacional das redes. Na rede Ethereum, por exemplo, esse problema é contornado por meio do uso de GAS. Como todo tipo de execução computacional custa um certo volume de GAS, um laço só será executado enquanto houver GAS. Todo laço infinito, portanto, será executado apenas até o fim do GAS.

pergunta e o retângulo representa a resposta final (o tipo da Blockchain ou se não deveria ser utilizada). Para cada pergunta será dado um exemplo no contexto dos sistemas de informação na área de saúde.

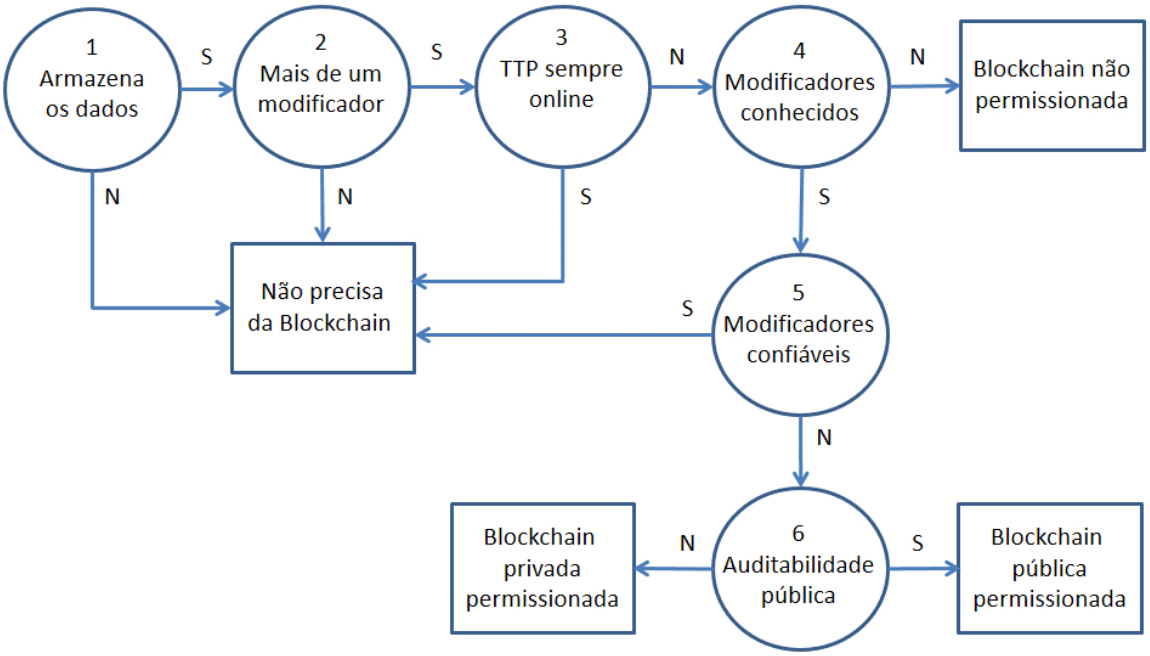


Figura 2.14. Fluxograma da Metodologia

A primeira pergunta a ser respondida é: *Precisa armazenar os dados*? A resposta pode ser sim ou não. No primeiro caso, o sistema armazena todas as mudanças nos prontuários dos pacientes. No segundo caso, o sistema captura a temperatura do paciente, talvez de algum aparelho, e envia um alerta se passar de um certo valor, mas sem armazenar a temperatura. Se responder não à pergunta, não é necessário utilizar a Blockchain. Se responder sim, passe à segunda pergunta.

A segunda pergunta é: *Há mais de um modificador*?. Entenda-se por modificador a entidade que realiza mudanças nas informações, seja inserindo-as, modificando-as ou removendo-as. A resposta pode ser sim ou não. No primeiro caso, diversas entidades (*e.g.*, cada sistema de informação das clínicas e hospitais) realizam modificações nos prontuários dos pacientes. No segundo caso, somente uma entidade é a responsável pela informação (vide centralização da informação, explicada na Seção 2.3). Se responder não à pergunta, não é necessário utilizar a Blockchain e talvez um banco de dados tradicional resolveria. Se responder sim, passe à terceira pergunta.

A terceira pergunta é: *Os TTP estão sempre online*? Entenda-se por TTP (*Trusted Third Party*, em inglês) às entidades às quais sempre é possível confiar, tanto na procedência quanto nas informações que entrega. Exemplos de um TTP seriam sistemas de informação do ministério da saúde ou do conselho federal de medicina. A resposta pode ser sim ou não. Se responder sim, talvez esse sistema (por ser confiável e permanecer sempre online) poderia ser responsável por armazenar as informações, não sendo necessário o uso da Blockchain. Se responder não, talvez esse sistema (por ser confiável, mas nem

sempre permanecer online) poderia ser usado como um certificador dos modificadores. Assim sendo, passe à quarta pergunta.

A quarta pergunta é: *Os modificadores são conhecidos*? A resposta pode ser sim ou não. Nesse caso, a pergunta refere-se a se os modificadores podem ser certificados e portanto responsabilizados em algum momento caso haja um comportamento malicioso. Por exemplo, conhecer quem é o responsável por um sistema de informação de um determinado hospital. Se responder não, cabe o uso de uma Blockchain do tipo não permissionada. Se responder sim, passe à quinta pergunta.

A quinta pergunta é: *Os modificadores são confiáveis*? A resposta pode ser sim ou não. No primeiro caso, quer dizer que os modificadores confiam que não existirão comportamentos maliciosos entre eles (*e.g.*, modificações indevidas, acessos sem permissão, etc). No segundo caso, não é possível garantir a confiança de todos (talvez há uma entidade terceira, *e.g.*, um plano de saúde, com problemas na justiça). Se responder sim à pergunta, não é necessário utilizar a Blockchain e talvez um banco de dados tradicional resolveria. Se responder não, passe à sexta e última pergunta.

A sexta pergunta é: *A auditabilidade é pública*? A resposta pode ser sim ou não. No primeiro caso, qualquer participante (podendo ser entidades ou até sistemas externos ou pessoas) tem permissão para validar todas as informações armazenadas. No segundo caso, somente um conjunto restrito e bem definido de participantes terá acesso às informações. Se responder sim, cabe o uso de uma Blockchain do tipo permissionada e pública. Se responder não, cabe o uso de uma Blockchain do tipo permissionada e privada.

2.9. Aplicações da Blockchain na área de Saúde

A Seção 2.3 apresentou que o uso da Blockchain permite a construção de sistemas descentralizados e auditáveis, com a característica de oferecer interoperabilidade e privacidade de dados. Esta seção explora como a Blockchain pode ser colocada a serviço dos sistemas de informação em Saúde. Para isso, primeiro, apresenta-se um levantamento sobre os usos da Blockchain em Saúde. A seguir, discorre-se sobre o potencial de algumas serem aplicadas no Brasil; para isso, são enumerados alguns cenários e casos de uso.

2.9.1. Levantamento sobre aplicações da Blockchain no contexto da Saúde

Um estudo bibliométrico recente [41] demonstrou que o número de artigos sobre Blockchain publicados em revistas da área de saúde ainda é pequeno se comparado com outras áreas. Dabbargh *et al.* [41] analisaram 995 artigos sobre Blockchain (de 2013 a 2018) indexados pelo portal Web of Science; destes, apenas 30 ($0,3\%$) foram publicados em revistas da área de saúde. O estudo também demonstrou um salto no número de artigos de 2016 para 2017, quando a atenção deixou o Bitcoin e passou a focar em usos gerais da Blockchain e de contratos inteligentes.

Apesar de não haver muitos artigos sobre Blockchain publicados em revistas da área de saúde, a literatura internacional (principalmente de Ciência da Computação e Engenharia) tem apresentado inúmeras proposta de utilização da Blockchain em saúde. A seguir apresenta-se uma lista de trabalhos relevantes publicados recentemente.

Casino *et al.* [42] realizaram uma revisão sistemática sobre os desafios e as diferentes áreas de aplicação da Blockchain. Fez uma análise qualitativa de 260 artigos e 54 relatórios (manuais, *white papers*, etc.) publicados entre 2014 e abril de 2018. O trabalho classificou as aplicações da Blockchain em 9 grandes áreas, a saber: financeira, negócios e indústria, gerenciamento de dados, verificação de integridade, governança, Internet das Coisas, privacidade e segurança, educação e saúde. Na área de saúde, considerou que o registro eletrônico de saúde provavelmente será a aplicação de maior impacto. Outras aplicações em saúde que poderiam se beneficiar da tecnologia Blockchain e, sobretudo, de contratos inteligentes seriam: transparência de recursos públicos aplicados em saúde, obtenção de dados para estudos longitudinais, arbitragem de processos (por exemplo, liberação automática de exames), acesso *online* de pacientes aos seus dados, compartilhamento de dados de saúde, controle de medicamentos e de ensaios clínicos e medicina de precisão.

Alonso *et al.* [43] fizeram uma revisão sistemática especificamente sobre os usos da Blockchain em Saúde, em artigos publicados entre 2010 e 2018 em algumas das principais bases de artigos (IEEE Xplore, Google Scholar, PubMed, Science Direct, Web of Science e ResearchGate). O trabalho encontrou 18 referências relevantes, sendo uma de 2016 [44], quatro de 2017 e treze de 2018, o que aponta para um crescimento da importância dessa temática. Segundo a revisão, os principais obstáculos são a escalabilidade e a implementação de controles de acesso. A principal vantagem seria o acesso a uma grande quantidade de informação anonimizada, que poderia ser utilizada para: desenvolvimento e aprimoramento de tratamentos personalizados, racionalização dos custos de ações de saúde e melhorias nas políticas de saúde. Alonso *et al.* [43] destacam ainda que a tecnologia Blockchain, associada ao avanço dos sistemas de informação e a maior participação e envolvimento dos pacientes, poderia levar a uma nova era do cuidado com a Saúde.

De forma geral, os trabalhos analisados por Alonso *et al.* [43] demonstram a viabilidade do uso da Blockchain para:

- Gerenciamento de identidade de pacientes.
- Registro de informações médicas com segurança e privacidade, permitindo a verificação de autenticidade de registros e preservando a identidade de pacientes e de profissionais de saúde [45].
- Rastreabilidade das ações médicas.
- Redução do tempo para interoperabilidade de dados [46]

Outra revisão sistemática sobre o uso da Blockchain em Saúde, realizada por Vazirani *et al.* [47], analisou qualitativamente 71 trabalhos, extraídos das seguintes bases: PubMed, Scopus, EMBASE, MEDLINE, ProQuest, CINAHL, AMED, Global Health, Books@Ovid e Cochrane Library. Este trabalho chamou a atenção para um detalhe importante não apontado em Alonso *et al.* [43], a compatibilidade legal. Citou o exemplo europeu, cujo Regulamento Geral sobre a Proteção de Dados[6] [48], no artigo 17, define

[6]Disponível em https://eur-lex.europa.eu/legal-content/PT/TXT/PDF/?uri=CELEX:02016R0679-20160504from=EN. Visitado em março de 2019.

o direito ao apagamento dos dados, também chamado "direito a ser esquecido". Desse modo, os dados devem poder ser apagados; entretanto, os dados gravados em uma estrutura Blockchain usual não podem ser apagados. No Brasil, a lei nº 13.709, de 14 de agosto de 2018, classifica os dados de saúde como dados sensíveis e estabelece o direito a ser esquecido nos artigos 7, 8, 15 e 16 desta lei. Desse modo, o projeto de armazenamento de dados de Saúde na Blockchain deve ter como requisito a funcionalidade de permitir apagar os dados de um paciente ou de revogar a sua autorização de acesso a estes dados. Voltaremos mais adiante a falar sobre este ponto.

Outro fator apontado por Vazirani *et al.* [47] foram os custos associados ao desenvolvimento de sistemas de informação, o que pode ser uma barreira para a adoção de sistemas baseados em Blockchain.

Park *et al.* [49] tentaram demonstrar experimentalmente os limites de uso da Blockchain para o compartilhamento de dados médicos. Para isso, criaram uma rede privada baseada na tecnologia Ethereum com um hospital e 300 pacientes. Ao final do experimento concluíram que era fundamental: minimizar a quantidade de dados efetivamente gravada na Blockchain, melhorar a privacidade dos dados e considerar os custos das transações. O trabalho foi baseado na rede Ethereum, onde cada transação tem um custo medido em *GAS*. Esse custo é proporcional ao tamanho do dado armazenado e inversamente proporcional a prioridade de gravação. Se fosse usada a rede oficial da Ethereum, o custo de uma transação, atualmente, seria de no mínimo cerca de 10 centavos de dólar. Parece pouco, mas quem absorve estes custos? Clientes ou unidades de saúde? Cabe ainda chamar a atenção para a imprevisível flutuação do valor da moeda digital Ethereum (ETH), que pode, eventualmente, tornar estes custos mais relevantes. O trabalho também chamou a atenção para o fato de que dados de saúde são dados sensíveis e que não podem ficar abertos nas redes Blockchain, devem ser protegidos por mecanismos criptográficos.

O trabalho de McGhin *et al.* [50] faz um resumo dos desafios tecnológicos para a implantação real de sistemas de informação em Saúde baseados em Blockchain. O texto traz dois pontos interessantes: *i*) apresenta uma compilação de projetos de software que dão suporte a aplicações de saúde usando Blockchain e *ii*) enumera vulnerabilidades agregadas aos sistemas de saúde pelo uso da Blockchain. Este último, em outras palavras, lembra que os sistemas de informação de saúde já estão sujeitos a requisitos severos de segurança e ao se usar Blockchain novas vulnerabilidades devem ser levadas em consideração, como, por exemplo, a susceptibilidade a ataques de 51% [51].

Ao todo, McGhin *et al.* [50] destacam nove iniciativas (MedRec [52], Gem Health Network [53], OmniPHR [54], PSN [55], Virtual Resources [56], Context-driven Data Logging [57], MedShare [58], Trial and Precision Medicine [59] e Healthcare Data Gateways [60]) e chama a atenção para o fato de que todos os trabalhos possuem limitações de escalabilidade e/ou de segurança. Por exemplo, o projeto MedRec é, atualmente, o artigo mais citado aplicações da Blockchain em Saúde [52]. O trabalho trata do uso da Blockchain para o armazenamento auditável do histórico de interações médicas do paciente. A arquitetura, baseada em *Proof of Work*, permite a mediação de permissão de acesso aos dados, mas, segundo McGhin *et al.* [50], o protótipo MedRec não garante satisfatoriamente o anonimato dos dados individuais e torna possível, por meio de técnicas forenses, descobrir a identidade dos pacientes e de prestadores de serviços.

A partir das leituras acima, pode-se concluir que ainda não existem aplicações robustas para o uso da Blockchain em saúde, ou que estas aplicações ainda estão em desenvolvimento. Mas podemos esperar que em breve possamos encontrar aplicações em produção. Segundo Perera *et al.* [37][7], o surgimento de aplicações Blockchain mais complexas pode levar de 5 a 10 anos.

O caminho para o desenvolvimento de aplicações relevantes, seguras e escaláveis, passa pelo refinamento de seus requisitos. A Seção 2.9.2, a seguir, organiza alguns dos principais cenários de aplicações e seus requisitos funcionais e não funcionais.

2.9.2. Cenários e casos de uso da Blockchain em Saúde

O trabalho de Liam Bell [61] argumenta que são quatro as principais potenciais contribuições da Blockchain para a área da saúde, a saber:

- **Compartilhamento de dados** de saúde, principalmente entre médicos e entre instituições de saúde, com a anuência do paciente. Sem dúvida, a principal aplicação nessa categoria são os prontuários eletrônicos.
- **Interoperabilidade de dados** de saúde em contexto nacional, permitindo avanços em desenvolvimento de sistemas para controle epidemiológico e de saúde coletiva. Nessas aplicações são fundamentais as aplicações de técnicas de anonimização dos dados.
- **Rastreamento da cadeia de suprimentos e de dispositivos médicos**, permitindo a auditoria sobre o uso de equipamentos em ambiente hospitalar, a prevenção de subutilização e a análise de fraudes.
- **Rastreamento da cadeia de distribuição de medicamentos**, o que permite verificar a prescrição de medicamentos aos pacientes, identificar lotes de medicamentos com problemas e realizar, se necessário, *recalls* de lotes de medicamentos (vencidos, falsificados ou com problemas no processo de fabricação).

Zhang *et al.* [15], por sua vez, acrescentam mais itens a essa lista. O trabalho aponta seis casos de uso principais, sendo que os três últimos são substancialmente diferentes dos apontados por Liam Bell [61]:

- **Compartilhamento seguro de dados** para, por exemplo, viabilizar telemedicina.
- **Monitoração da prescrição de drogas**. Nesse item, inclui-se a monitoração de drogas controladas e de alto custo.
- **Observação agregada de eventos** (Big Data) com aplicações principalmente em saúde coletiva.
- **Identificação de pacientes** (PKI [62]).
- **Compartilhamento de dados para pesquisa científica**.

[7] Artigo sob avaliação, disponível em https://peerj.com/preprints/27529/.

- **Implementação de estruturas autônomas**, por exemplo, para a gestão e controle de seguros de saúde suplementar.

A identificação de pacientes pode ser considerada uma das partes fundamentais de sistemas de Prontuário Eletrônico do Paciente — e uma das componentes mais complexas. Por exemplo, atualmente no Brasil ainda não há um sistema de informação para identificação única dos pacientes. Um dos primeiros esforços foi a implementação do Cartão SUS e mais recentemente a proposta de um Documento Nacional de Identificação. Por sua importância, a componente de gestão de identidade de saúde deve ser tratada como um caso de uso isolado, separado dos Prontuários, com requisitos próprios de segurança, privacidade e de integração com outros sistemas.

O compartilhamento de dados para pesquisa científica visa a análise de dados de saúde, longitudinais, no curso de longos períodos de tempo, para desenvolvimento de drogas e de tratamentos mais eficientes. Requer, segundo os padrões modernos de proteção de dados individuais, mecanismos para concessão e revogação de direitos de uso dos dados.

A última sugestão de Zhang *et al.* [15] passa pela organização de estruturas autônomas para prestação de serviços médicos. Nesse tipo de aplicação a Blockchain é responsável por, automaticamente, analisar demandas e ofertas de serviços de saúde. Por exemplo, um prestador de serviços pode ser diretamente remunerado utilizando mecanismos de pagamento semelhantes ao Bitcoin. O principal desafio em aplicações como essa é a escrita de contratos inteligentes capazes de tratar toda a complexidade de sistemas de saúde complementar.

A seguir, tratamos os principais requisitos para aplicações de Identificação de Pacientes, Prontuário Eletrônico e Gestão de Medicamentos. Escolhemos estes casos de uso por eles serem de aplicação imediata [37] e relevantes para a área de saúde.

2.9.2.1. Identificação Única de Pacientes

O leitor pode estranhar que o primeiro caso de uso da Blockchain em Saúde a ser apresentado seja a identificação única de pacientes e não os prontuários eletrônicos. Ao longo do texto indicamos que os prontuários devem ser a aplicação de maior impacto, mas para isso é necessário um serviço de identidade e de gerenciamento de credenciais digitais.

O processo de atendimento médico é rico de momentos em que a privacidade de um paciente fica exposta. Dados pessoais ficam expostos quando preenchemos formulários de papel, quando falamos nossos dados pessoais a um atendente e até mesmo quando são transmitidos ou armazenados sem seguir os devidos requisitos de segurança [63].

Essas fragilidades de segurança, quer sejam de processo, quer sejam de sistema, podem ser minimizadas com o uso de sistemas para gerenciamento de identidades. Orman *et al.* [62] fazem reflexões sobre a migração da identidade dos indivíduos do papel para os meios digitais. Aponta que Blockchain é, talvez, a mais forte das candidatas para a criação de identidades digitais.

Um serviço digital de gerenciamento de identidades e de credenciais deve:

- Permitir a identificação única de um usuário por um **ID principal**, que pode ser um número, uma validação biométrica ou uma chave pública, entre outras possibilidades. Cabe observar que este ID principal pode ser mantido privado e que pode não ser possível identificar um indivíduo a partir dele.

- Permitir a criação de ilimitados **IDs secundários**. Se um paciente utiliza o mesmo ID em diferentes instituições, ele expõe a sua identidade; de modo alternativo, cada relação do paciente poderia utilizar um ID diferente e o serviço de gerenciamento de identidades seria responsável em gerenciar esses IDs.

- Armazenar dados pessoais, chaves de acesso a estes dados e validar solicitações de acesso aos dados. Deveria, inclusive, ser capaz de revogar acessos.

- Emitir credenciais de acesso que permitam a um requisitante ter acesso a dados específicos, sem expor totalmente os dados de saúde do paciente. Essas credenciais podem ter atributos adicionais, tais como localização, nível de acesso aos dados e validade da autorização de acesso.

O sistema deve proteger a identidade e gerenciar as credenciais de acesso aos dados pessoais. Diferentemente de como é hoje, um paciente, ao chegar da triagem de um hospital, poderia, ao invés de repassar todos os seus dados pessoais, mostrar um QRCode contendo uma credencial de acesso aos seus dados, protegendo assim sua identidade e privacidade.

No contexto de sistemas públicos de saúde, o sistema pode ainda ser utilizado para garantir a consistência de cadastros de usuários. Atualmente, na prática, não existe um registro único de pacientes. Talvez, por uma questão de privacidade, esse registro único nem devesse existir. Talvez trouxesse mais problemas do que benefícios. Sem um sistema de gestão de identidade, é difícil garantir que não haja vazamentos e maus usos de dados pessoais.

Na busca por uma padronização de cadastros, o Ministério da Saúde propôs a ampla adoção do Cartão Nacional de Saúde (Cartão SUS), regulamentado pela portaria número 940/2011, para construção de cadastros de usuários. Outra iniciativa brasileira para padronização da identificação, esta mais recente, foi a proposta do Documento Nacional de Identificação (DNI), implementado pela lei Nº 13.444/2017 e regulamentado no decreto Nº 9.278/2018.

Entretanto, os esforços em torno do Cartão SUS e do DNI não garantem a criação de uma base consistente de usuários. Mas a existência de uma base consistente e com identificação única (mesmo que não revele a identidade do paciente) é essencial para construção de sistemas de saúde que permitam a interoperabilidade de dados.

A identificação de usuários poderia ser realizada a partir da construção de uma infraestrutura Blockchain, onde o usuário é representado por seu ID principal. Os dados pessoais sensíveis podem ser gravados na Blockchain, ou em um *data lake*, de forma protegida por chaves criptográficas [14]. Contratos inteligentes podem ser implementados para a gestão de credenciais digitais, assim como para verificar requisições de dados.

A implementação de tal rede Blockchain certamente não é trivial. Uma primeira questão é a sua manutenção. Quem deve manter a rede no ar? Nessa aplicação não pode-se contar com os benefícios financeiros para manutenção da rede, como acontece nas redes Bitcoin e Ethereum. Acreditamos que tal rede poderia ser mantida por órgãos públicos (Ministério da Saúde e secretarias estaduais e municipais). A arquitetura do serviço provavelmente seria a de uma Blockchain permissionada.

Dentre os requisitos não funcionais mais importantes estaria o tempo de resposta às requisições de autorização de acesso a dados, que idealmente não deveria ultrapassar a ordem de segundos. A tecnologia atual da Blockchain permissionada permite a realização de transações com tempo de execução na ordem de segundos [10]. Mas como poderia ser a implantação de uma Blockchain permissionada com tamanho e abrangência compatíveis com o SUS?

2.9.2.2. Registro Eletrônico de Dados de Saúde

A maioria dos autores que descrevem os casos de uso da Blockchain [37, 50, 41, 49, 47, 15] colocam as aplicações de Prontuário Eletrônico e de compartilhamento de dados de saúde dentre as de maior impacto na área de saúde. Segundo Zhang*et al.* [15], esse tipo de tecnologia pode ser vista sob dois pontos de vista: o ponto de vista das unidades de saúde, que normalmente se utilizam de abordagens centralizadas para armazenar dados de saúde dos pacientes, e ponto de vista do paciente, que deseja ter uma visão geral dos seus dados de saúde. O primeiro está normalmente associado ao conceito de *Electronic Health Records* (EHR) e o segundo a *Personal Health Records* (PHR).

Imagine, para simplificar a apresentação da ideia, que todos os dados de saúde de um paciente sejam gravados em uma Blockchain. Suponha ainda que essa Blockchain conta com um serviço de identificação única de pacientes, como descrito na seção anterior. O uso da Blockchain nessas categorias de aplicação poderia [14]:

- Simplificar a **interoperabilidade** de dados. Os dados do paciente, coletados em um hospital A, poderiam ser acessados durante uma consulta em um outro hospital B, desde que o acesso fosse autorizado.
- Dar real **controle** ao paciente sobre os dados que dizem respeito a ele. Segundo a legislação atual, um paciente pode solicitar que os seus dados médicos sejam apagados de uma instituição. Na prática isso é muito difícil de ser realizado, o uso de contratos inteligentes poderia permitir a revogação de direitos de acesso a documentos [14].
- Tornar **transparente** os usos e os acessos aos dados dos pacientes. Pacientes poderiam monitorar o uso dos seus dados para a agregação de dados e geração de estatísticas de saúde coletiva. Caso estes dados fossem utilizados em pesquisas e desenvolvimentos de produtos, seria possível reclamar direitos sobre a propriedade intelectual desenvolvida [64]. É interessante observar que o uso correto da Blockchain e de técnicas de anonimização de dados pode permitir a disponibilização de dados sem o comprometimento da identidade dos pacientes.

A implementação de tais redes de compartilhamento de dados depende da migração de sistemas de EHR atuais para um novo padrão. Um padrão que contemple restrições severas de segurança e que especifique interfaces claras para compatibilidade de trocas de dados. Além de implementar restrições de segurança da área de saúde (*e.g.* HIPAA e NGS2), sistemas baseados em Blockchain estão sujeitos a novos ataques, como os ataques de 51% e *Sybil*.

Tal rede provavelmente seria permissionada, mantida por unidades de saúde e financiadores do sistema de saúde. A principal restrição técnica para a implantação seria a escalabilidade em termos de volume de transações. Hoje, por exemplo, o Brasil conta com cerca de 7.522 hospitais [65] — muitos deles mal conectados à Internet.

O leitor pode se perguntar sobre onde deveriam ficar armazenados os dados do paciente. Certamente não devem ficar na Blockchain, mas em *data lakes* dedicados. Em outro trabalho [14], sugerimos o uso de serviços atuais de armazenamento em nuvem, tais como Google Drive e Dropbox. Mas cabe citar que o próprio Governo Federal brasileiro possui uma iniciativa para criação de um *data lake*, que deverá ser utilizado por unidades de saúde tanto públicas quanto privadas: o Conjunto Mínimo de Dados (CMD, vide: https://conjuntominimo.saude.gov.br/#/cmd). Um sistema de EHR baseado em Blockchain poderia gravar os dados nesses *data lakes* genéricos e gravar na Blockchain apenas o link para esses dados.

É interessante pensar que as unidades de saúde podem beneficiar-se da rede, pois a qualidade dos atendimentos tende a aumentar. Para gestores de saúde complementar, por exemplo, a principal vantagem seria econômica, por exemplo, com a não realização de exames que tenham sido recentemente feitos. Entretanto, deve-se ter em mente que os maiores beneficiados seriam os pacientes, quer por ter mais dados, quer por ter maior controle sobre estes dados.

2.9.2.3. Gestão de Medicamentos e de Prescrições (*e-Prescription*)

Uma aplicação mais específica é a gestão de medicamentos e de prescrições médicas (receitas). A aplicação de controle eletrônico de prescrições também pode ser considerada como uma parte dos sistemas de EHR.

No Brasil, quase a totalidade das receitas são em papel. O maior problema com as receitas de papel são os erros de interpretação; mas podemos citar outros ploblemas, como a perda do registro em papel e a facilidade para falsificação.

No mundo, diversos países já têm adotado sistemas de prescrição eletrônica (*e-Prescription* ou *e-Rx*). Na Europa, os sistemas avançaram, sobretudo, devido a necessidade de criar um padrão europeu comum [66]. Em alguns países, como na Estônia, a prescrição digital já contabiliza cerca de 80% do total de prescrições. As vantagens do uso de sistemas de *e-Prescription*, além de eliminar papel e aumentar a qualidade da informação, são:

- Controlar o abuso de uso de medicamentos.
- Permitir a análise de interações medicamentosas.

- Simplificar comércio eletrônico de medicamentos, inclusive de medicamentos controlados.

O uso da Blockchain permitiria aos sistemas de *e-Prescription* melhorar a interoperabilidade entre sistemas, ampliar a capacidade de fazer o *recall* de lotes de medicamentos (se a cadeia de produção for registrada na Blockchain) e aumentar a confiabilidade da informação.

No Brasil, o SUS é responsável por garantir o acesso a medicamentos de alto custo; sabe-se que o sistema de distribuíção é sujeito a falhas, fraudes [8] e até mesmo a furtos [9]. Especialmente no contexto de distribuição de medicamentos de alto custo, um sistema baseado em Blockchain permitiria o rastreamento desses medicamentos, o controle de validade e previsão de consumo.

Sistemas de *e-Prescription* baseados em Blockchain teriam como principais requisitos a existência de identificação única e o volume de transações. Pode-se imaginar que cadeias de distribuição de medicamentos possam ter interesse em manter tais sistemas, quer para melhorar a qualidade do serviço prestado, quer para conhecer hábitos e demandas dos clientes. Uma arquitetura permissionada seria a escolha mais provável para esse tipo de aplicação.

2.10. Ferramentas

Os projetos Hyperledger e Ethereum são principais referências sobre desenvolvimento de contratos inteligentes para o contexto de, respectivamente, Blockchains privadas e públicas. A seguir são apresentadas as características gerais de cada uma dessas soluções.

2.10.1. Hyperledger Fabric

O projeto *Hyperledger*, e seu software *Hyperledger Fabric*, é de código aberto. Nascido em 2015, seu objetivo é prover uma plataforma flexível, que permita usar a Blockchain em diferentes cenários além das criptomoedas. O projeto é mantido pela Linux Foundation e é apoiado por centenas de empresas de tecnologia, como a IBM, Intel, entre outras. As quatro principais característica do *Hyperledger Fabric* são: (1) as redes não são anônimas (*permissioned networks*), estabelecendo assim uma maior confiança na rede; (2) as transações têm critérios de permissão para leitura; (3) possui uma arquitetura flexível; (4) facilidade de uso, permitindo escrever aplicações em diversas linguagens de programação. Além disso, por ser permissionada, o Hyperledger Fabric não exige consumo de energia excessivo, como o Bitcoin.

No Hyperledger Fabric, os componentes são divididos por responsabilidades, sendo eles: os nós autorizadores, armazenadores, coletores, coordenadores e clientes. Além disso, os componentes na arquitetura se comunicam utilizando canais, estruturas criadas especificamente para realizar transações de forma privada e confidencial, isolando dife-

[8] Leia *"SUS joga fora R$ 16 milhões em medicamentos de alto custo"*, por André Shalders. Disponível em https://www.bbc.com/portuguese/brasil-41007650.

[9] Leia *"Ladrões roubam medicamentos de alto custo do Hospital São Paulo"*, por Roberto Paiva e Paula Paiva Paulo. Disponível em https://g1.globo.com/sp/sao-paulo/noticia/2018/08/02/ladroes-roubam-medicamentos-de-alto-custo-do-hospital-sao-paulo-veja-video.ghtml

rentes aplicações. Assim, o *canal* é o meio pelo qual os componentes podem se comunicar de forma segura e confiável com a Blockchain.

Os nós autorizadores (*fabric certificate authorities*) são os responsáveis por duas tarefas: a primeira, em certificar que qualquer componente, seja este um usuário ou um contrato inteligente, que quiser utilizar o sistema seja quem diz ser (em outras palavras, pelo reconhecimento da autenticidade do componente); a segunda, em autenticar o componente e autorizá-lo a usar certas funcionalidades (*e.g.*, realizar transações) ou acessar outros componentes, após sua certificação.

Os nós armazenadores (*committing peer*) são os responsáveis por persistir uma ou mais cadeias de transações, que foram transmitidas através dos canais criados no sistema. Assim, são os nós que armazenam as diversas Blockchains, um por cada canal criado. Essa abordagem, de uma Blockchain por canal, permitirá obter dois benefícios: privacidade e escalabilidade.

Em se tratando da privacidade, um componente não poderá acessar (*i.e.*, visualizar ou modificar) uma Blockchain de um nó armazenador associado a um canal, se não tiver acesso a esse canal. Nesse contexto, a privacidade e confidencialidade das informações fica assegurada para os componentes que obtiveram a permissão.

Já na escalabilidade, note que poderão haver diversas Blockchains, uma por canal, o que permitirá distribuir a quantidade de transações entre diversos nós armazenadores, aumentando a quantidade de requisições que um nó deve atender, consequentemente, aumentando a escalabilidade do sistema.

Os nós coletores (*endorsing peer*) são os responsáveis por duas tarefas: a primeira, coletar as transações advindas dos clientes; a segunda, analisar, utilizando contratos inteligentes, se a transação tem alguma política ou regra associada.

Os nós coordenadores (*ordering peer*) são os responsáveis por duas tarefas: a primeira, receber as transações dos clientes; a segunda, realizar uma ordenação nessas transações para que a Blockchain esteja consistente (*i.e.*, fique igual em todos os nós que armazenarão essas transações). Nesse sentido, todos os nós coordenadores atuando sobre uma determinada Blockchain deverão chegar a um consenso da ordem na qual as transações serão adicionadas na Blockchain pelos nós armazenadores. O recebimento das transações é realizada usando a tecnologia Apache Kafka [67], que permite armazená-las de forma distribuída e com tolerância a folhas. Já o consenso é realizado utilizando a tecnologia Zookeeper [68], que aplica uma versão do Paxos, explicada na Seção 2.5.8.

Os nós clientes (*application*) são encarregados por efetuar transações no sistema, por enviá-las aos nós coletores e repassá-las aos nós ordenadores. No Hyperledger Fabric, um cliente do sistema pode ser uma pessoa física (que utiliza o sistema através de um aplicativo), ou uma pessoa jurídica que utiliza o sistema através de interfaces de comunicação ou aplicativos.

A Figura 2.15 mostra um exemplo de uma arquitetura que pode ser criada para um cenário onde existem três organizações que utilizam diferentes canais de comunicação do Hyperledger. Na figura, a cor representa a que organização o componente está associado.

O primeiro passo para criar a arquitetura é gerar o nó autenticador, responsável

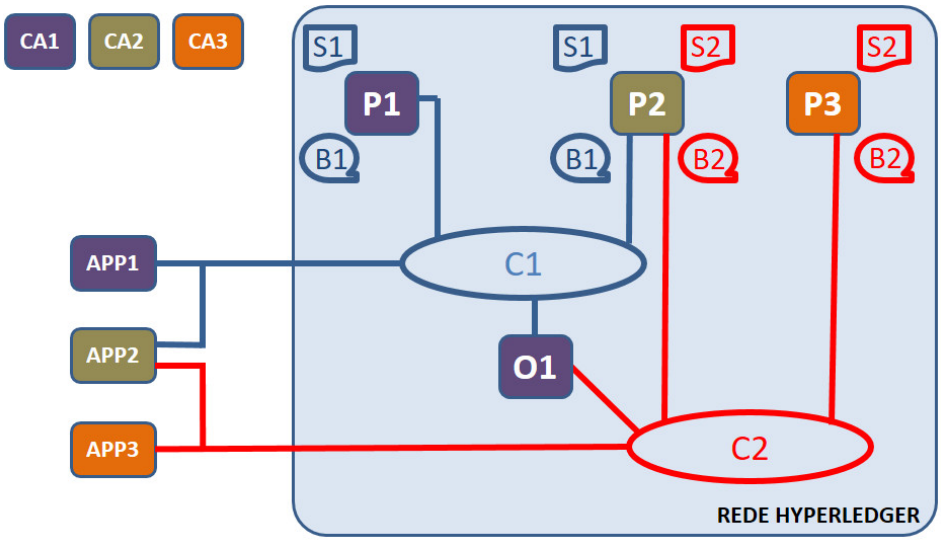


Figura 2.15. Exemplo da Arquitetura Hyperledger

pela autenticação dos componentes do sistema. Assim, tanto a organização 1 quanto a 2 e a 3 geram seus próprios nós autenticadores (CA1, CA2 e CA3).

Após os autenticadores, será necessário criar os nós armazenadores, coletores e ordenadores. Nesse sentido, pode-se observar que a organização 1 designou o nó (P1) para realizar tanto as funções de coletor quanto de armazenador. Note que, por ser um coletor, também terá a responsabilidade de analisar as transações utilizando contratos inteligentes (S1). Além disso, por ser um armazenador, terá a responsabilidade de armazenar a Blockchain (B1) do canal 1 (C1). Da mesma forma, o nó P3, designado pela organização 3, será responsável por analisar as transações utilizando o contrato inteligente S3 e armazenar a Blockchain (B2) do canal 2 (C2). Por outro lado, note que o nó P2 foi designado pela organização 2, porém é responsável por analisar e armazenar as transações em ambas Blockchains B1 e B2.

A seguir, é necessário definir a comunicação entre as organizações, em outras palavras, definir os canais. Note que na arquitetura de exemplo existem dois canais C1 e C2. No canal C1, somente os componentes autorizados pela organização 1 e 2 poderão se comunicar e utilizar a Blockchain B1. Nenhum outro componente (por exemplo, da organização 3) poderá acessar esse canal. Da mesma forma, o canal C2 permite somente a comunicação e utilização da Blockchain B2 pelas organizações 2 e 3.

Finalmente, é necessário definir os nós responsáveis pela ordenação das transações e as aplicações que utilizarão a arquitetura. Na figura, pode-se observar que há somente um nó ordenador O1, designado pela organização 1 e três clientes (cada um autorizado pelo respectivo componente autorizador da organização correspondente). Note que o cliente APP1 ou APP3 somente possuem acesso ao canal 1 ou 2, respectivamente. Já o cliente APP2 possui acesso tanto ao canal 1 quanto ao canal 2 (nesse sentido, o cliente pertence às duas organizações, mas com dois acessos diferentes).

A Figura 2.16 mostra como o fluxo das mensagens transita na arquitetura, criada

no exemplo anterior, quando um cliente quer enviar uma transação. Vamos assumir que a transação requer a modificação de um prontuário eletrônico do cliente e que somente a aplicação APP1 realiza a transação. Para uma melhor visualização, serão eliminadas as conexões e os componentes que fazem parte do canal 2, haja vista que a APP1 somente utilizará o canal 1.

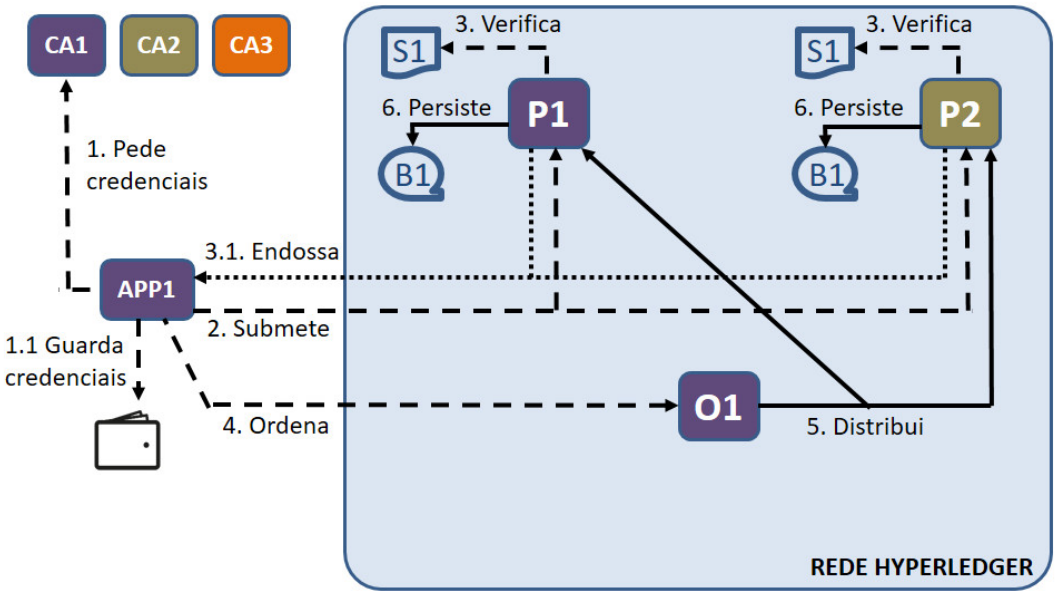


Figura 2.16. Fluxo de Mensagens

1. O cliente APP1 obtém o acesso do autorizador CA1 correspondente a sua organização. Quando o autorizador entrega as credenciais de acesso, o cliente as mantêm em uma estrutura denominada carteira (Wallet) e as armazena no seu disco rígido, pendrive, etc. Note que esse armazenamento permitirá ao cliente APP1 conectar-se novamente sem precisar pedir a autorização ao CA1.

2. O cliente se conecta ao canal C1 e submete a transação para um dos nós coletores desse canal, no caso P1, P2 ou ambos.

3. O peer coletor P1 verifica, utilizando o contrato inteligente S1, se a transação pode ser realizada, devolvendo essa informação ao cliente (em outras palavras, endossando a transação). Note que, dependendo do contrato inteligente, pode ser que seja necessário que o nó coletor P2 também tenha que verificar a transação.

4. O cliente recebe a informação de que a transação pode ser realizada e a envia para o nó ordenador.

5. O nó ordenador recebe a transação, cria um bloco com esta transação, e a distribui para os nós armazenadores (no caso P1 e P2).

6. Os nós armazenadores analisam novamente a transação (validam se houve alguma outra transação anterior que também modificou o prontuário). Se não houver nenhum problema, o bloco (com a transação) é persistido na Blockchain B1, enviando a informação ao cliente. O envio da informação não é mostrado na figura.

Como mencionado, (contratos inteligentes) são programas que permitem a validação de uma transação em termos de regras de negócios. Por exemplo, se o negócio está relacionado com operações financeiras entre pessoas, uma regra de negócio a ser implantada seria a verificação de se a pessoa tem saldo para realizar a transferência.

No Hyperledger, um contrato inteligente pode ser implementado em diversas linguagens de programação, atualmente com versões para Node.js (Javascript), Go e Java. Entretanto, independente da linguagem, o contrato inteligente possui uma estrutura a ser seguida pelo desenvolvedor.

Na estrutura, o desenvolvedor deve definir como mínimo: o modelo (com os ativos e participantes); as transações a serem realizadas; o controle de acesso. Em posse das definições, o Hyperledger criará um arquivo .bna, o qual será implantado nos coletores.

Para entender a estrutura, daremos um exemplo simples de um contrato inteligente que valida se um usuário pode ou não visualizar o prontuário eletrônico armazenado na Blockchain.

No modelo, os ativos (denominados *asset*) serão os prontuários eletrônicos. cada um com um identificador único, o site onde o pdf está armazenado e o cpf do paciente ao qual está associado, definido como:

```
asset Prontuario identified by idProntuario {
   o String idProntuario
   o String urlProntuario
   o String cpfPaciente
}
```

Os participantes (denominados participant) serão todas os pacientes que inseriram seu prontuário eletrônico, definido como:

```
participant Paciente identified by cpfPaciente {
   o String cpfPaciente
}
```

A única transação (denominadas transaction) permitida será a de visualizar o prontuário, desde que o cpf do participante seja o mesmo do cpf inserido no ativo. Assim, a transação será definida como:

```
transaction Visualizar {
   --> Prontuario prontuario
   --> Paciente pac
}
```

Cuja implementação, mostrada abaixo, recebe a transação 'Visualizar' (linha 1), a seguir verifica se os cpf são os mesmos (linha 2), e mostra a url do pdf (linha 3) ou uma mensagem de "sem acesso" (linha 5), dependendo da verificação dos cpf.

```
1 function verProntuario(Visualizar) {
2    if (Visualizar.prontuario.cpfPaciente == Visualizar.pac.cpfPaciente) {
3       return Visualizar.prontuario.urlProntuario
4    } else {
5       return "sem acesso"
6    }
```



```
7 }
```

Finalmente, o controle de acesso (denominado rule) permite a definição das permissões que o sistema dará tanto para a execução das transações, quanto para os participantes. Por exemplo, a regra abaixo permite que todos os participantes possam acessar a função verProntuario.

```
rule AcessoTotal {
  description: "Permissoes para todos os participantes"
  participant: "org.hyperledger.composer.system.Participant"
  operation: ALL
  resource: "org.hyperledger.composer.system.**"
  action: ALLOW
}
```

2.10.2. Ethereum

O projeto Ethereum também é de código aberto. Nascido em 2015, seu objetivo é prover uma plataforma descentralizada, que permita a execução de contrato inteligente para usar a Blockchain em diferentes cenários além das criptomoedas. O projeto é mantido pela Ethereum Foundation e é apoiado pela Enterprise Ethereum Alliance e por centenas de programadores do mundo todo. A principal característica, que a diferencia com a Blockchain do Bitcoin, é a possibilidade de executar os contrato inteligente, permitindo criar regras específicas para cada tipo de negócio.

No Ethereum, os componentes são divididos por responsabilidades, sendo eles: os nós armazenadores, mineradores, coletores e clientes. A diferença do Hyperledger, que permite ter diversas Blockchains no mesmo sistema, no Ethereum existem duas Blockchains: o *testnet*, Blockchain de teste utilizado pelos desenvolvedores na criação de suas aplicações e o *mainnet*, Blockchain oficial com as informações reais de todas as transações da rede.

Os nós armazenadores, denominados no Ethereum de Full nodes, são os encarregados de armazenar a cadeia de blocos e transações validados desde o bloco gênesis. Nesse sentido, esses nós permitem ter a prova da integridade da Blockchain.

Os nós mineradores são nós armazenadores, ou seja, possuem a Blockchain completa, com a capacidade de criar novos blocos, utilizando o consenso via prova de trabalho POW explicada na Seção 2.5.8.

Os nós coletores, denominados no Ethereum de Light-weight nodes, geralmente são nós cujo poder computacional ou de armazenamento é pequeno (por exemplo, dispositivos móveis, plugins em navegadores Web, entre outros). Esses nós são encarregados de verificar certas transações, mas confiando nas informações dos armazenadores, haja vista que não possuem a Blockchain completa. Um exemplo de uso é verificar o saldo de um determinado identificador, olhando somente as transações onde ele aparece.

Os nós clientes, ao igual que no Hyperledger, são encarregados por efetuar transações no sistema e por enviá-las aos nós coletores ou armazenadores (que, no caso, podem ser eles mesmos, dependendo do poder computacional). No Ethereum, os clientes podem ser uma pessoa física ou também um contrato inteligente, já que estes também geram

transações de acordo a certas regras de negócios. Cada cliente possui uma chave pública e outra privada, a chave publica é seu identificador na rede Ethereum e a chave privada é utilizada para efetuar uma transação na rede.

Os contratos inteligentes do Ethereum podem ser considerados programas que funcionam sobre a Blockchain, esses programas são executados em uma máquina virtual chamada *Ethereum Virtual Machine* (EVM). A EVM é responsável por se comunicar com as informações armazenadas na cadeia de blocos e conforme as regras do contrato, pode transferir valores, inserir novas informações ou requisitar chamadas para APIs externas a rede.

Os contratos inteligentes podem ser desenvolvidos em linguagens de programação de alto nível como o java ou python. Há também a linguagem Solidity, linguagem de alto nível semelhante ao JavaScript, esta é a mais utilizada pelos desenvolvedores. Existe algumas IDEs que auxiliam no processo de desenvolvimento, uma destas é o Remix, plataforma que se conecta a rede *testnet* e possibilita criar e testar os contratos inteligentes.

Diferente do Hyperledger, no Ethereum para cada inserção de informação na Blockchain é cobrada uma taxa chamada gás, esta taxa deve ser enviada junto a transação e para seu pagamento é utilizado o éter, moeda digital baseada na rede Ethereum. Este taxa foi criada com a intenção de proteger a rede de ataques e abusos por parte dos participantes.

Para entender melhor um contrato inteligente criado na rede Ethereum, mostraremos um exemplo simples da estrutura de dados utilizada para armazenar as informações do pacientes e uma função criada para validar se um usuário possui acesso para exibir estas informações, neste exemplo foi utilizado a linguagem *Solidity*. A estrutura denominada Paciente possui alguns atributos: *paciente* do tipo *address* é o identificador único do paciente. O tipo *address* é um endereço público da rede Ethereum. O campo *url* armazena o site onde esta armazenado o pdf. O campo *acessos* é um *array* de endereços, será o responsável por armazenar os usuarios que possuem acesso as informações.

```
struct Paciente {
    address paciente;
    string urlProntuario;
    address[] acessos;
    mapping(address => bool) roles;
}
```

Utilizando a estrutura anterior, foi criado uma função para validar se o usuário que está solicitando as informações possui tal acesso. A função *VerificaAcesso* recebe por parâmetro o identificador do paciente *_paciente*. A primeira verificação realizada é se o solicitante é o próprio paciente, caso seja, este poderá visualizar suas informações (Linha 2). A próxima validação é se o identificador do solicitante esta listado na relação de identificadores com permissão de acesso do paciente (linha 3). Caso as duas verificações anteriores sejam falsas, a função não permitirá que as informações do paciente sejam exibidas.

```
1 modifier VerificaAcesso (address _paciente) {
2     if ( msg.sender != _paciente &&
3         !pacientesLista[_paciente].roles[msg.sender]) {
```



```
4             revert();
5         }
6         _;
7     }
```

2.11. Leituras sugeridas

Caberá ao leitor continuar a pesquisa sobre Blockchain e suas ferramentas. Esse mercado está em grande atividade e diariamente surgem novas técnicas, ferramentas e algoritmos. Para maior contato com a área, sugerimos uma série de recursos e leituras:

- Não apenas por uma questão histórica, mas também por sua clareza e para tomar contato com a concepção inicial, sugerimos a leitura do artigo seminal de Satoshi Nakamoto [3] sobre Bitcoin.
- O artigo de Zhang *et al.* [15] oferece um bom panorama sobre os usos de Blockchain em saúde.
- As plataformas Coursera (www.coursera.org), edX (www.edx.org) e Udacity (www.udacity.com) oferecem vários cursos sobre Blockchain, sendo alguns gratuitos.
- A documentação das plataforma Ethereum (http://www.ethdocs.org) e os tutoriais para o Hyperledger Fabric (https://hyperledger-fabric.readthedocs.io/en/latest/tutorials.html) são excelentes fontes de informação. Essa documentação está atualizada e nela pode-se encontrar o *roadmap* para os futuros desenvolvimentos.

  Àqueles que pretendem ensinar Blockchain, sugerimos apresentar a plataforma Ethereum para a escrita dos primeiros contratos inteligentes. A plataforma Hyperledger Fabric seria outra escolha natural para uma experiência didática, mas a instalação da plataforma pode ser uma barreira para os estudantes menos avançados.

Além dos sites e referências acima, pode-se acompanhar o mercado de criptomoedas e de lançamentos de novas ferramentas na mídia especializada (exemplos: https://cointelegraph.com/ e https://www.coindesk.com).

2.12. Conclusões

Este capítulo apresentou as características principais de Blockchain, suas limitações e seus casos de uso principais em Saúde. Esperamos ter oferecido uma visão do potencial e das limitações de Blockchain, tal que os profissionais de Sistemas de Informação possam discernir quando usar Blockchain e quais devem ser as escolhas de projeto. O que deve conter uma transação? Qual o mecanismo de consenso mais apropriado? Entre outras decisões de projeto.

Antes de encerrar, chamamos a atenção para o conceito de confiança. A principal inovação de Blockchain foi deslocar a garantia de confiança das instituições para a arquitetura computacional. Mesmo que essas arquiteturas ainda sejam muito complexas, os próximos anos devem ser prolíficos na redução de complexidade das soluções,

assim como na mitigação dos desafios de escalabilidade, privacidade, mutabilidade, entre outros.

Em poucos anos, as aplicações baseadas em Blockchain poderão ser uma das principais formas para registro confiável de dados. Para que isso aconteça serão necessários desenvolvedores de software capacitados para compreender e dominar a complexidade das implantações de Blockchain e dos contratos inteligentes. Esperamos que esse capítulo possa ajudar nessa caminhada.

Referências

[1] Melanie Swan. *Blockchain: Blueprint for a new economy*. O'Reilly Media, Inc., 2015.

[2] Fabíola Greve, Leobino Sampaio, Jauberth Abijaude, Antonio Coutinho, Ítalo Valcy, and Sílvio Queiroz. Blockchain e a revolução do consenso sob demanda. *Minicursos do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (Minicursos_SBRC)*, 36, 2018.

[3] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf, 2008.

[4] Ralph C Merkle. Method of providing digital signatures (1979). *URL: https://www. google. com/patents/US4309569 (visitado em abril de 2019)*.

[5] Eric Brewer. Cap twelve years later: How the "rules" have changed. *Computer*, 45(2):23–29, 2012.

[6] Takuro Nakagawa and Naohiro Hayashibara. Energy efficient raft consensus algorithm. In *International Conference on Network-Based Information Systems*, pages 719–727. Springer, 2017.

[7] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.

[8] Leslie Lamport. Generalized consensus and paxos. Technical report, Microsoft Research MSR-TR-2005-33, 2005.

[9] G. Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 2014.

[10] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolić, Sharon Weed Cocco, and Jason Yellick. Hyperledger fabric: A distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, pages 30:1–30:15, New York, NY, USA, 2018. ACM.

[11] Don Tapscott and Alex Tapscott. *Blockchain Revolution: How the Technology Behind Bitcoin Is Changing Money, Business, and the World*. Brilliance Audio, 2016.

[12] Lemuria Carter and Jolien Ubacht. Blockchain applications in government. In *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*, dg.o '18, pages 126:1–126:2, New York, NY, USA, 2018. ACM.

[13] Seyoung Huh, Sangrae Cho, and Soohyung Kim. Managing IoT devices using blockchain platform. In *Advanced Communication Technology (ICACT), 2017 19th International Conference on*, pages 464–467. IEEE, 2017.

[14] Arlindo F. da Conceição, Flavio S. Correa da Silva, Vladimir Rocha, Angela Locoro, and João Marcos M. Barguil. Eletronic health records using blockchain technology. *Workshop em Blockchain: Teoria, Tecnologias e Aplicações (WBlockchain, SBRC)*, 1(1/2018), 2018.

[15] Peng Zhang, Douglas C. Schmidt, Jules White, and Gunther Lenz. Blockchain technology use cases in healthcare. Advances in Computers. Elsevier, 2018.

[16] Cécile Monteil. Blockchain and health. In *Digital Medicine*, pages 41–47. Springer, 2019.

[17] A. Oram. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly Media, 2001.

[18] Bram Cohen. The BitTorrent protocol specification, 2008.

[19] Salman Baset and Henning Schulzrinne. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol. *Proceedings - IEEE INFOCOM*, 2005.

[20] Brendan McCallum. Digital dollars, masks, and black markets: The cypherpunk legacy. 2013.

[21] Nick Szabo. Smart contracts: Building blocks for digital markets. http://www.alamut.com/subj/economics/nick_szabo/smartContracts.html, 1996. Acessado: 28 Mar. 2019.

[22] Top 100 Criptomoedas por Capitalização de Mercado. https://coinmarketcap.com. Acessado: 22 Abr. 2019.

[23] I. Bashir. *Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained, 2nd Edition*. Packt Publishing, 2018.

[24] A.M. Antonopoulos. *Mastering Bitcoin: Programming the Open Blockchain*. O'Reilly Media, 2017.

[25] D.R. Stinson. *Cryptography: Theory and Practice, Third Edition*. Discrete Mathematics and Its Applications. Taylor & Francis, 2005.

[26] Ilya Mironov. Hash functions: Theory, attacks, and applications. Technical report, November 2005.

[27] Andrew S. Tanenbaum and Maarten van Steen. *Distributed Systems: Principles and Paradigms (2nd Edition)*.

[28] R. C. Merkle. Protocols for public key cryptosystems. In *1980 IEEE Symposium on Security and Privacy*, pages 122–122, April 1980.

[29] Vitalik Buterin. On public and private blockchains. *Ethereum blog*, 7, 2015.

[30] Elli Androulaki et al. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, pages 30:1–30:15, New York, NY, USA, 2018. ACM.

[31] Brown, Carlyle, Grigg, & Hearn. *Corda: An Introduction*. https://docs.corda.net/_static/corda-introductory-whitepaper.pdf, 2016.

[32] K. Wüst and A. Gervais. Do you Need a Blockchain? In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pages 45–54, June 2018.

[33] Leslie Lamport. Paxos made simple. *ACM SIGACT News (Distributed Computing Column) 32, 4 (Whole Number 121, December 2001)*, pages 51–58, December 2001.

[34] Christian Gorenflo, Stephen Lee, Lukasz Golab, and Srinivasan Keshav. FastFabric: Scaling Hyperledger Fabric to 20,000 Transactions per Second. *CoRR*, abs/1901.00910, 2019.

[35] Brasil. Ministério de Saúde. Política nacional de informação e informática em saúde. http://bvsms.saude.gov.br/bvs/publicacoes/politica_nacional_infor_informatica_saude_2016.pdf, 2016. Acessado: 28 Mar. 2019.

[36] Duane Bender and Kamran Sartipi. Hl7 fhir: An agile and restful approach to healthcare information exchange. In *Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems*, pages 326–331. IEEE, 2013.

[37] Srinath Perera, Frank Leymann, and Paul Fremantle. A use case centric survey of blockchain: *status quo* and future directions. *PeerJ Preprints*, 7:e27529v1, 2019.

[38] The Economist. Why bitcoin uses so much energy. https://www.economist.com/the-economist-explains/2018/07/09/why-bitcoin-uses-so-much-energy. Acessado: 21 de abril de 2019.

[39] Xiaoqi Li, Peng Jiang, Ting Chen, Xiapu Luo, and Qiaoyan Wen. A survey on the security of blockchain systems. *Future Generation Computer Systems*, 2017.

[40] Steve Davies. blockchain is here. what's your next move?

[41] Mohammad Dabbagh, Mehdi Sookhak, and Nader Sohrabi Safa. The evolution of blockchain: A bibliometric study. *IEEE Access*, 2019.

[42] Fran Casino, Thomas K Dasaklis, and Constantinos Patsakis. A systematic literature review of blockchain-based applications: current status, classification and open issues. *Telematics and Informatics*, 2018.

[43] Susel Góngora Alonso, Jon Arambarri, Miguel López-Coronado, and Isabel de la Torre Díez. Proposing new blockchain challenges in ehealth. *Journal of medical systems*, 43(3):64, 2019.

[44] Ariel Ekblaw, Asaph Azaria, John D Halamka, and Andrew Lippman. A case study for blockchain in healthcare:"medrec" prototype for electronic health records and medical research data. In *IEEE Open & Big Data Conference*, volume 13, page 13, 2016.

[45] You Sun, Rui Zhang, Xin Wang, Kaiqiang Gao, and Ling Liu. A decentralizing attribute-based signature for healthcare blockchain. In *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9. IEEE, 2018.

[46] Alevtina Dubovitskaya, Zhigang Xu, Samuel Ryu, Michael Schumacher, and Fusheng Wang. Secure and trustable electronic medical records sharing using blockchain. In *AMIA Annual Symposium Proceedings*, volume 2017, page 650. American Medical Informatics Association, 2017.

[47] Anuraag A Vazirani, Odhran O'Donoghue, David Brindley, and Edward Meinert. Implementing blockchains for efficient health care: Systematic review. *Journal of medical Internet research*, 21(2):e12439, 2019.

[48] Paul Voigt and Axel Von dem Bussche. The EU General Data Protection Regulation (GDPR). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 2017.

[49] Yu Rang Park, Eunsol Lee, Wonjun Na, Sungjun Park, Yura Lee, and Jae-Ho Lee. Is blockchain technology suitable for managing personal health records? mixed-methods study to test feasibility. *Journal of medical Internet research*, 21(2):e12533, 2019.

[50] Thomas McGhin, Kim-Kwang Raymond Choo, Charles Zhechao Liu, and Debiao He. Blockchain in healthcare applications: Research challenges and opportunities. *Journal of Network and Computer Applications*, 2019.

[51] Deepak K Tosh, Sachin Shetty, Xueping Liang, Charles A Kamhoua, Kevin A Kwiat, and Laurent Njilla. Security implications of blockchain cloud with analysis of block withholding attack. In *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 458–467. IEEE Press, 2017.

[52] Asaph Azaria, Ariel Ekblaw, Thiago Vieira, and Andrew Lippman. Medrec: Using blockchain for medical data access and permission management. In *Open and Big Data (OBD), International Conference on*, pages 25–30. IEEE, 2016.

[53] Matthias Mettler. Blockchain technology in healthcare: The revolution starts here. In *2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–3. IEEE, 2016.

[54] Alex Roehrs, Cristiano André da Costa, and Rodrigo da Rosa Righi. Omniphr: A distributed architecture model to integrate personal health records. *Journal of biomedical informatics*, 71:70–81, 2017.

[55] Jie Zhang, Nian Xue, and Xin Huang. A secure system for pervasive social network-based healthcare. *IEEE Access*, 4:9239–9250, 2016.

[56] Mayra Samaniego and Ralph Deters. Hosting virtual iot resources on edge-hosts with blockchain. In *2016 IEEE International Conference on Computer and Information Technology (CIT)*, pages 116–119. IEEE, 2016.

[57] Muhammad Siddiqi, Syed Taha All, and Vijay Sivaraman. Secure lightweight context-driven data logging for bodyworn sensing devices. In *2017 5th International Symposium on Digital Forensic and Security (ISDFS)*, pages 1–6. IEEE, 2017.

[58] QI Xia, Emmanuel Boateng Sifah, Kwame Omono Asamoah, Jianbin Gao, Xiaojiang Du, and Mohsen Guizani. Medshare: Trust-less medical data sharing among cloud service providers via blockchain. *IEEE Access*, 5:14757–14767, 2017.

[59] Zonyin Shae and Jeffrey JP Tsai. On the design of a blockchain platform for clinical trial and precision medicine. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 1972–1980. IEEE, 2017.

[60] Xiao Yue, Huiju Wang, Dawei Jin, Mingqiang Li, and Wei Jiang. Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control. *Journal of medical systems*, 40(10):218, 2016.

[61] Liam Bell, William J Buchanan, Jonathan Cameron, and Owen Lo. Applications of blockchain within healthcare. *Blockchain in Healthcare Today*, 2018.

[62] Hilarie Orman. Blockchain: the emperors new PKI? *IEEE Internet Computing*, 22(2):23–28, 2018.

[63] Maria José Amaral Salomi and Rafael Fabio Maciel. Gestão de documentos e automação de processos em uma instituição de saúde sem papel. *Journal of Health Informatics*, 8(1), 2016.

[64] Mehdi Benchoufi and Philippe Ravaud. Blockchain technology for improving clinical research quality. *Trials*, 18(1):335, 2017.

[65] Alexandre Marinho. A crise do mercado de planos de saúde: devemos apostar nos planos populares ou no SUS? *Planejamento e Políticas Públicas*, (49), 2017.

[66] Patrick Kierkegaard. E-prescription across europe. *Health and Technology*, 3(3):205–219, 2013.

[67] J. Kreps, N. Narkhede, and J. Rao. Kafka: A distributed messaging system for log processing. In *Proceedings of 6th International Workshop on Networking Meets Databases (NetDB), Athens, Greece*, 2011.

[68] Patrick Hunt, Mahadev Konar, Flavio P. Junqueira, and Benjamin Reed. Zookeeper: Wait-free coordination for internet-scale systems. In *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, pages 1–14, 2010.

Capítulo 3

Ontologias biomédicas: teoria e prática

Fernanda Farinelli[1,2] e Mauricio B. Almeida[1,3]

[1] Programa de Pós Graduação em Gestão e Organização do Conhecimento, UFMG

[2]fernanda.farinelli@gmail.com, [3]mba@eci.ufmg.br

Abstract

This chapter is dedicated to the study of ontology, an interdisciplinary research field that examines entities and relationships as a mean for better performing modeling in knowledge domains. There is a growing interest in the application of ontologies for problem solving in several areas, such as Computer Science, Information Science, Philosophy, Artificial Intelligence, Linguistics, Knowledge Management; as well as within knowledge domains, such as Health, Biomedicine, Law, Geography, to name a few. Our approach here is called "Applied Ontology", a field that gathered together new technologies, in particular those developed for the Semantic Web, and solid philosophical foundations. The first part of the chapter presents basic information about ontology and its application in information systems; in the second part, a practical example of the application of ontologies in the field of medicine is developed.

Resumo

O presente capítulo se dedica ao estudo da ontologia, um campo de pesquisa interdisciplinar que estuda entidades e relações fundamentais para a modelagem de domínios do conhecimento. Existe um crescente interesse na aplicação de ontologias para a solução de problemas em diversas áreas como Ciência da Computação, Ciência da Informação, Filosofia, Inteligência Artificial, Linguística, Gestão do Conhecimento; bem como em domínios como Saúde, Biomedicina, Direito, Geografia, para citar alguns. A abordagem aqui é a da "Ontologia Aplicada", um campo que alia novas tecnologias, em particular aquelas desenvolvidas para a Web Semântica, a uma sólida fundação filosófica de cunho metafísico. Na primeira parte do capitulo apresentam-se informações básicas sobre ontologia e sua aplicação em sistemas de informação; na segunda parte, apresenta-se exemplo prático da aplicação de ontologias no domínio da medicina.

3.1. Teoria em ontologias

3.1.1. Fundamentos: ontologias e modelos baseados em ontologias

Ontologia é um tema que tem sido estudado em diferentes campos de pesquisa – Filosofia, Ciência da Computação e Ciência da Informação – e no âmbito de vários domínios do conhecimento – Medicina, Biologia, Direito e Geografia, para citar alguns. Apesar da ampla difusão do termo, não é uma tarefa simples entender o que significa "ontologia". Na Filosofia, ontologia é um ramo da Metafísica que diz respeito a quais categorias de entidades existem e estão relacionadas [Lowe 2007]. Em Ciência da Computação, ontologias são aplicadas à modelagem de sistemas de informação (SIs), em diferentes ramos: sistemas tradicionais baseados em bancos de dados, e sistemas de baseados em conhecimento, assim nomeados no campo da Representação do Conhecimento. Em Ciência da Informação, Vickery [1997, p. 285] conclui que "os problemas com os quais os cientistas da informação vêm lutando há muito tempo são enfrentados agora pela comunidade de engenheiros do conhecimento." Tais problemas dizem respeito questões de modelagem.

Antes de adentrar em detalhes sobre os diversos significados para o termo "ontologia", apresenta-se, para entendimento do contexto, um breve histórico desde os primeiros modelos de dados até os modelos baseados em ontologias.

Modelos são representações simplificadas da realidade que se busca entender. O mundo é complexo e os modelos são produzidos para permitir que a compreensão humana apreenda e organize fatos. Modelos também são entidades importantes e partes integrantes do método científico. Uma das formas de classificar os modelos é considerar a questão semântica, a qual trata das funções da representação. Deste ponto de vista, os modelos podem ser *modelos de fenômenos*, *modelos teóricos* ou *modelos de dados* [Frigg 2006]. Modelos de dados proliferam nas organizações modernas como um meio de representar o que se pretende codificar e processar em SIs.

Os SIs têm papel relevante na consolidação de novas práticas administrativas, pois visam atender às necessidades da corporação. O desenvolvimento de SIs envolve a criação de modelos para representar atividades e processos que ocorrem na corporação. Um modelo de dados corporativo é "[...] uma representação explícita da estrutura, atividades, processos, fluxos, recursos, pessoas, comportamento, objetivos e restrições de uma corporação" [Gandon 2002, p.42].

No desenvolvimento de SI, o estágio em que os modelos são criados para fins de compreensão humana é geralmente referenciado como *modelagem conceitual*. Os modelos conceituais são criados a partir de abstrações de aspectos da realidade, seja da perspectiva de um indivíduo ou de um grupo. As abstrações são um meio de especificar as entidades e as relações entre entidades dentro do domínio de um campo de conhecimento que é de interesse para o sistema em construção. A modelagem conceitual de SIs é resultado de pesquisas realizadas nos últimos 50 anos. As primeiras iniciativas para a especificação de modelos de dados datam do final da década de 1950 [Young, Kent 1958, Bosak *et al.* 1962]. Tais iniciativas objetivavam gerar modelos que fornecessem estruturas de dados para atendimento a necessidades computacionais.

Na década de 1960, a pesquisa em bancos de dados deu origem a três tipos principais de modelos de dados: o *modelo hierárquico*, o *modelo de rede* e o *modelo relacional*. Esses modelos são em geral referenciados como *modelos lógicos*, pois não alcançavam aspectos físicos (de

implementação). No entanto, os modelos lógicos apresentavam problemas que limitavam sua utilização na modelagem conceitual [Mylopoulos 1998]. Por exemplo, no modelo relacional [Codd 1970], um constructo denominado "relacionamento" é usado para representar tanto "entidades" quanto "relações entre entidades" do mundo real [Peckham, Maryanski 1988]. Esse fato gera problemas de comunicação e leva a erros de modelagem.

Os primeiros *modelos semânticos* utilizados na modelagem conceitual surgiram na década de 1970, no escopo do trabalho do Comitê ANSI/X3/SPARC para a padronização de sistemas de gerenciamento de banco de dados. Os mais notáveis são o *modelo de dados semântico* [Abrial 1974], a *arquitetura de três esquemas* [Jardine 1976], o *modelo entidade-relacionamento* (ER) [Chen 1976] e o *modelo ER estendido* [Codd 1979], dentre outros. A principal característica dos modelos semânticos, em comparação aos anteriores, é que eles são de mais fácil compreensão por pessoas.

O modelo ER, por exemplo, removeu a sobrecarga do constructo "relação" que existia nos modelos relacionais, além de fornecer termos adicionais para uso como primitivas de modelagem. A modelagem conceitual surgiu a partir de modelos de dados semânticos desenvolvidos para bancos de dados, mas o Comitê ISO/TC97/SC5 formou um grupo com o objetivo de determinar padrões para linguagens de modelagem conceitual de SIs.

Nos anos 90, propostas para *modelagem orientada a objetos* se tornaram populares. Muitos consideravam a orientação a objeto como uma categoria diferente dos modelos de dados. De fato, tais modelos tinham recursos adicionais em comparação com modelos de dados anteriores, mas ainda assim mantinham semelhanças nos constructos adotados, tais como: *objetos* versus *entidades*, *atributos* versus *propriedades*, *relacionamentos* versus *associações*, *classes* versus *hierarquias* [Milton 2000]. A *Unified Modeling Language* foi uma tentativa de padronizar as notações orientadas a objetos que incluíam diversas outras iniciativas: o Método Booch [Booch 1993], a Técnica de Modelagem de Objetos [Rumbaugh *et al.* 1991], a Engenharia de Software Orientada a Objetos [Jacobson *et al*. 1992], dentre outros.

Ao longo de todos esses anos, a criação de modelos conceituais vem sendo motivada pela busca de formas cada vez melhores de representar a realidade em SIs. A modelagem conceitual é "[...] a atividade de descrever formalmente alguns aspectos do mundo físico e social ao nosso redor para fins de compreensão e comunicação" [Mylopoulos 1992, p. 3]. No entanto, modelos semânticos usados na modelagem conceitual utilizam um conjunto limitado de constructos para a tarefa a que se propõe. A partir dos anos 1990, uma diversidade de iniciativas para o uso de ontologias nas corporações pode ser encontrada na literatura de SIs [Fox 1992, Uschold *et al*. 1998, Fillion *et al*. 1995, Schlenoff 1996, Bernus *et al*. 1996].

Smith e Welty (2001) apontam a inconsistência nas práticas durante os primeiros anos de modelagem conceitual como a principal causa dos problemas de interoperabilidade nos SIs atuais. Uma alternativa para esse tipo de problema são os *modelos baseados em ontologias*: "[...] a provisão, definitiva, de uma ontologia de referência comum e bem fundamentada – uma taxonomia compartilhada de entidades – pode fornecer vantagens significativas sobre os métodos *ad-hoc*, caso a caso, usados até aqui" [Smith, Welty 2001, p. 4].

O termo "ontologia" apareceu pela primeira vez na literatura da Ciência da Computação em 1967, na teoria de dados proposta por Mealy[1] [Smith 2003]. Historicamente, vários autores

[1] Mealy, G. H. (1967). Another Look at Data. Proceedings of AFIPS Conference. 31, 525–534. Washington: Thompson.

foram pioneiros no tema [Genesereth, Nilsson 1987, Gruber 1993, Guarino, 1995, Guarino, Giaretta 1995, Guarino 1998, Smith 2003, Vickery 1997, Wand *et al.* 1999]. Ontologias têm sido estudadas desde a década de 1970 na pesquisa de Inteligência Artificial e desde os anos 80 na modelagem conceitual de SI. Nos anos 90, a pesquisa em Web Semântica aumentou a demanda por ontologias para diversos tipos de aplicativos, tanto para solucionar problemas de interoperabilidade quanto para fornecer uma estrutura unificada de comunicação.

O estudo das ontologias é caracterizado pela coexistência de abordagens interdisciplinares, com pelo menos sete interpretações disponíveis na literatura para o termo ontologia [Guizzardi 2005]: i) uma disciplina filosófica; ii) um sistema conceitual informal; iii) um sistema baseado semântica formal; iv) uma especificação de uma conceitualização. v) uma representação de um sistema conceitual via teoria lógica; vi) um vocabulário usado por uma teoria lógica; vii) uma especificação (meta-nível) de uma teoria lógica. Uma ontologia descreve o significado dos símbolos adotados no SI e representa uma visão específica do mundo. As ontologias são nesse caso classificadas em duas dimensões principais: a dimensão de tempo corresponde à utilização de ontologias em SI, seja em tempo de desenvolvimento ou em tempo de execução; a dimensão estrutural lida com o uso da ontologia como um componente de banco de dados, como a interface do usuário ou como um aplicativo [Guarino 1998].

Em Representação do Conhecimento (RC), um subcampo da Inteligência Artificial (IA), o termo ontologia é usado para se referir a uma estrutura de entidades representados por um vocabulário lógico. Para melhor entendimento, considere-se as atividades e os agentes envolvidos na tarefa de representar conhecimento: *sistemas declarativos* contém declarações que representam *fatos* governados por *regras*. Um exemplo de um fato é "New York é uma cidade nos Estados Unidos" e de uma regra é "todas as pessoas que vivem em New York vivem nos Estados Unidos". Essa combinação de fatos e regras compõem uma *base de conhecimento* do sistema. Uma base de conhecimento é construída e mantida por um *engenheiro do conhecimento*, que tem como tarefa formalizar o conhecimento de um grupo de especialistas. Em muitos casos, a ontologia faz o papel da base de conhecimento.

Para executar tais tarefas, um engenheiro realiza generalizações e abstrações, as quais requerem *insights* metafísicos. Nessa linha de pensamento, uma ontologia é uma teoria representativa dos principais fatos e regras que governam parte da realidade, para fins computacionais. Existem outras abordagens em RC que empregam linguagens de representação mais expressivas para construir modelos para uma teoria [Guarino 1998]. Aderindo a um compromisso ontológico – uma descrição da conceitualização pretendida para uma teoria lógica [Guarino, Giaretta 1995] – emprega-se uma linguagem de representação para gerar um conjunto de modelos representativos da realidade. O papel da ontologia é tornar explícitos axiomas que restringem modelos, de forma a igualar, tanto quanto possível, os modelos que contém o significado pretendido (ver Figura 3.1).

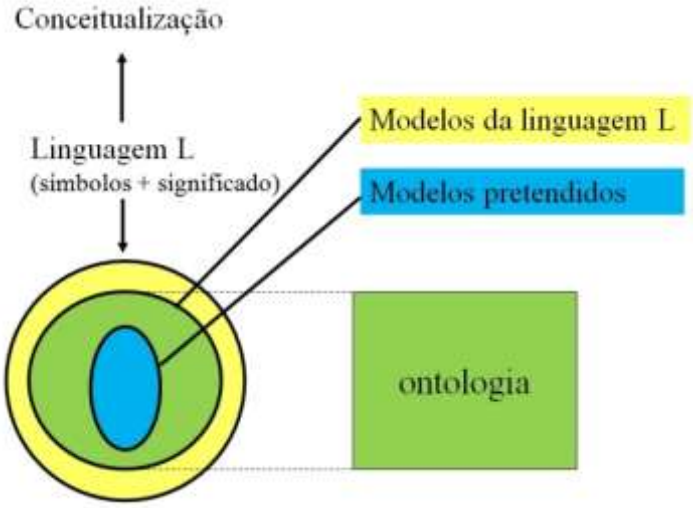


Figura 3.1. Ontologia e linguagem de representação
Fonte: Adaptado de Guarino (1998)

Uma explicação formal da Figura 1 deve incluir uma conceitualização C representada por uma linguagem de representação L aderente a um compromisso ontológico K= <C,I>. Na verdade, L se compromete com o domínio D através de K, do qual C é a conceitualização subjacente. A variável I representa uma função interpretação, a qual mapeia elementos de D para símbolos do vocabulário V. Todos os modelos de L compatíveis com K são modelos pretendidos de L de acordo com K. Nesse contexto, o papel da ontologia é restringir os modelos de L de forma que eles se tornem modelos pretendidos I_kL.

Observam-se então dois significados principais para o termo ontologia em Ciência da Computação.

O primeiro significado diz respeito ao uso de princípios ontológicos para entender e modelar a realidade (ver por ex. Wand, Weber 1990, Wand *et al*. 1999), o qual denominamos princípios da "ontologia como disciplina". O uso do termo nesse caso está alinhado com seu papel original na Filosofia, ou seja, fornecer uma descrição do que existe e caracterizar entidades nas atividades de modelagem.

Para um ilustrar de forma prática em sistemas de informação, a Figura 3.2 apresenta um modelo em *Unified Modeling Language* (UML) real com uma falha de modelagem: uma data é modelada como entidade, quando na verdade é uma propriedade. Esse é um tipo de falha simples que a ontologia, funcionando como um metamodelo, pode aprimorar. Na verdade, diversos outros tipos de falhas similares podem ser corrigidos pelo uso da ontologia como metamodelo. A Figura 3.3 apresenta o mesmo modelo UML da Figura 3.2 depois de corrigido pelo uso da ontologia *Bunge-Wand-Weber* (BWW) [Alturki *et al.* 2013]. O uso da ontologia como disciplina para criar ontologias como artefatos em geral gera modelos ontológicos bem fundamentados.

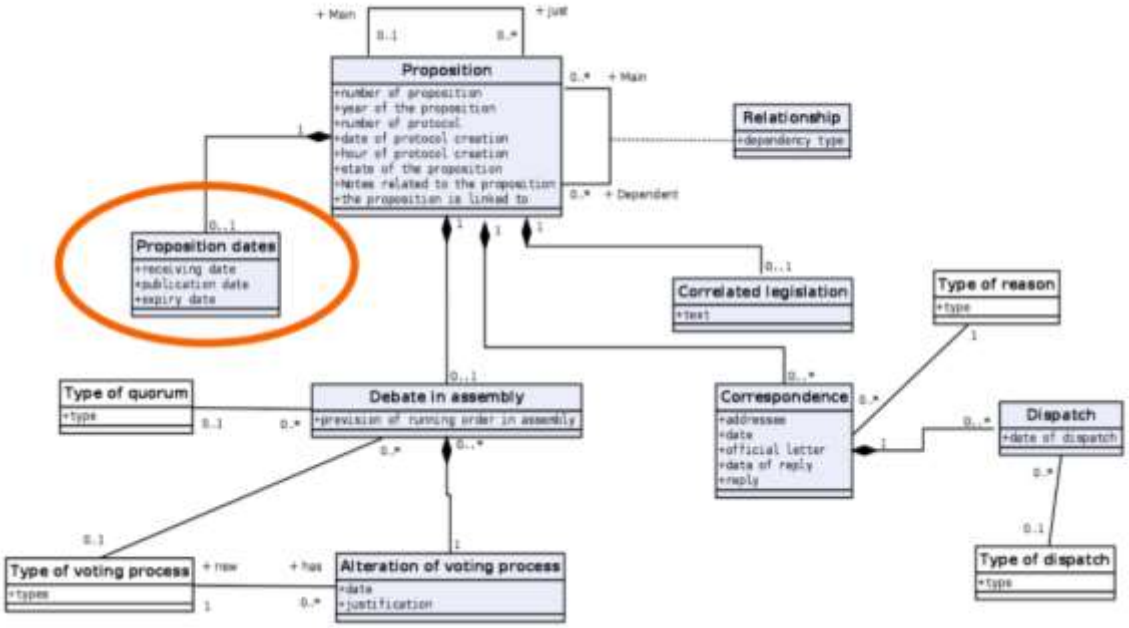


Figura 3.2. Modelo UML com falha na modelagem
Fonte: dos autores

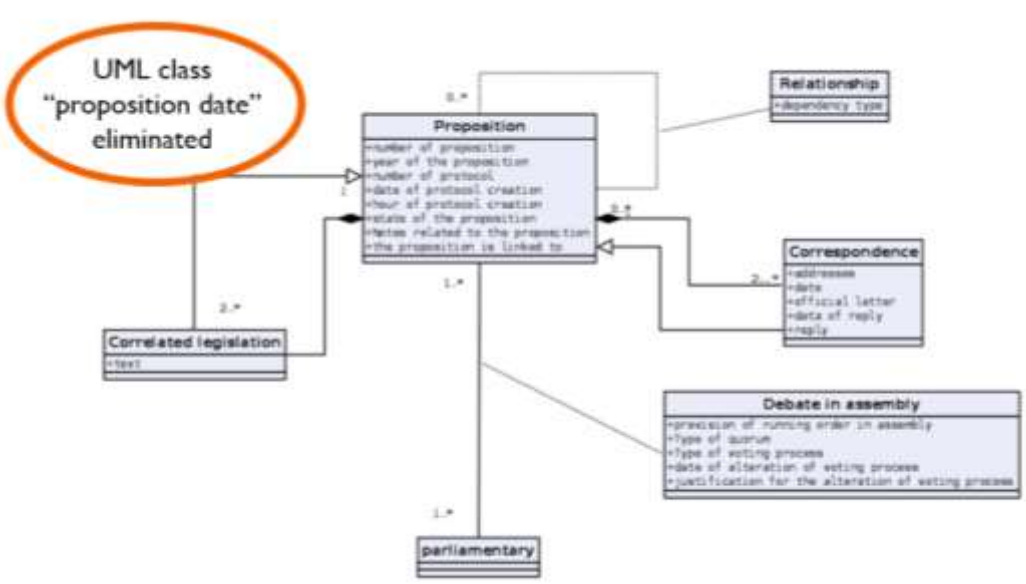


Figura 3.2. Modelo UML corrigido pelo uso da ontologia BWW
Fonte: dos autores

O segundo significado para o termo ontologia diz respeito a representação de um domínio em uma linguagem de representação computacional (ver por ex. Staab, Studer 2004), o qual denominamos aqui "ontologia como artefato". Uma ontologia, nesse caso, consiste de um conjunto de declarações expressas em uma linguagem de representação, a qual pode ser processado por mecanismos de inferência automatizados. Mecanismos de inferências são artefatos de software

que podem extrair conhecimento novo a partir de conhecimento existente em uma ontologia, devidamente axiomatizada. A inferência equivale a verificar as consequências lógicas.

Do ponto de vista filosófico, pode haver apenas uma ontologia [Smith 1998]. A fim de lidar com a questão da pluralidade de uso de termo, o qual passou a ser usado também no mundo dos SIs, distinguem-se dois tipos de ontologias: i) a ontologia real, que diz respeito a como o universo é organizado, e corresponde a uma abordagem filosófica; e ii) ontologia epistemológica, relacionada à tarefa de conceituar um domínio. Essa nomenclatura, entretanto, é confusa ao misturar os termos "ontologia" e "epistemologia", que em Filosofia estão em diferentes campos de estudo. Outras formas de distinção são encontradas na literatura, por exemplo: ontologia com "O" maiúsculo versus ontologia com "o" minúsculo [Guarino, Giaretta 1995]; ontologia "de" SI e ontologia "para" SI [Fonseca 2007]. No primeiro caso, a ontologia é usada para modelagem conceitual; no segundo, a ontologia é um componente do SI que descreve o vocabulário de um domínio com o objetivo de apoiar a criação de esquemas conceituais.

O quadro 3.1 reúne as interpretações genéricas para o termo ontologia em todos os campos em que o termo é amplamente citado. No restante do presente capitulo, adota-se assim a seguinte nomenclatura:

- *ontologia como disciplina*, para nomear a ontologia como disciplina filosófica e os princípios metafísicos subjacentes;
- *ontologia como artefato*, para se referir a qualquer artefato computacional construído com alguma linguagem declarativa de representação;
- *modelo ontológico*, para se referir a qualquer ontologia como artefato que faz uso dos princípios da ontologia como disciplina para obter boa fundamentação na modelagem.

Quadro 3.1. Quadro sinótico resumindo as visões sobre ontologia

Distinção	Campo	O que é?	Propósito	Exemplo
Ontologia como uma disciplina	Filosofia	Ontologia como um sistema de categorias	Entender a realidade, as coisas que existem e suas características	Sistemas de Aristóteles, Kant, Husserl
Ontologia como um artefato	Ciência da Computação	ontologia como uma teoria (baseada em lógica)	Entender um domínio e reduzi-lo à modelos	BFO, DOLCE (genéricas)
		ontologia como um artefato de software	Criar um vocabulário para representação em sistemas e para gerar inferências	OWL (linguagem de RC)
	Ciência da Informação	ontologia como uma teoria (informal)	Entender um domínio e classificar termos	Sistema de classificação de Ranganathan
		ontologia como um sistema conceitual informal	Criar vocabulários controlados para recuperação da informação a partir de documentos	um catálogo, um glossário, um tesauro

Fonte: adaptado de Almeida (2013)

A capacidade de integração entre modelos via a redução de ambiguidade inerente a construção de ontologias em linguagens declarativas formais possibilita aplicações relevantes em

campos críticos como a Medicina. Uma aplicação direta atualmente utilizada em todo o mundo é a melhoria da comunicação, para pessoas e máquinas, em partes dos sistemas médicos de grande importância para os cuidados a saúde, por exemplo, os prontuários de pacientes. Dessa forma, as ontologias podem auxiliar aos profissionais que lidam com sistemas de informação médicos ao prover uma fundamentação sólida para a construção de modelos alinhados com a realidade e, portanto, passiveis de melhor possibilidade de integração com outros sistemas, hoje facilmente distribuídos temporal e geograficamente.

3.1.2. Entidades em ontologias: classes, tipos, relações, propriedade, atributos, instâncias

Existe consenso de que o estudo da ontologia como disciplina diz respeito aos tipos de coisas que existem. Nesse contexto, "tipo" quer dizer "categoria", um termo que foi usado ainda por Aristóteles para discutir que declarações se pode fazer sobre uma entidade. De fato, uma teoria das categorias é o mais importante tópico do estudo da ontologia como disciplina. Tais teorias especificam sistemas de categorias estruturados em níveis hierárquicos, em geral, na forma de uma árvore invertida na qual a categoria de mais alto nível é nomeada "entidade". Qualquer coisa pode ser descrita como uma entidade de algum tipo, mas qual os próximos níveis de categorização são questão para discussão.

Em termos de Filosofia, três sistemas de categorias são os mais influentes e, mesmo que não diretamente, influenciam a forma como os modelos SIs são criados atualmente: o de Aristóteles[2], o de Kant[3] e o de Husserl[4] [Thomason 2009]. Apresenta-se a seguir uma breve introdução ao sistema de classificação de Aristóteles, uma vez que ele foi a base para todos os subsequentes e é adotado até os dias de hoje.

Aristóteles usou a linguagem como uma pista para descrever as categorias ontológicas. Kant utilizou conceitos como uma maneira de abordar categorias de objetos de uma possível cognição. O objetivo dos sistemas de categorias aristotélico e kantiano era descrever a estrutura categórica do mundo de acordo com o pensamento e a linguagem humanos. Partindo das categorias de Aristóteles, Husserl forneceu categorias descritivas das mais altas essências das coisas possíveis. Husserl não fornece um inventário das coisas que realmente existem, ainda que tenha sido o criador do termo "ontologia formal".

Aristóteles parece ter sido o primeiro filósofo a usar a palavra grega "categoria" como um termo técnico para predicação. Embora os estudos de Aristóteles tenham diferentes interpretações, os seguidores de sua tradição acreditam que um sistema de categorias deveria fornecer um inventário das coisas que existem. O primeiro sistema de categorias proposto por Aristóteles dividiu entidades em dois ramos: *dito-sobre* e *presente-em*, tornando possível as seguintes situações:

- qualquer entidade é dita de outra, ou não é dita de outra
- qualquer entidade está presente em outra; ou não está presente em outra

Entidades que são *ditas-sobre* outras são chamadas *universais*; enquanto aquelas que não são *ditas-sobre* outras são chamadas de *particulares*. Entidades que estão *presentes-em* outras são

[2] Aristóteles - filósofo grego, 384–322AC

[3] Immanuel Kant - filósofo alemão, 1724–1804

[4] Edmund Gustav Albrecht Husserl - filósofo alemão, 1859–1938

chamadas de *acidentes*, enquanto aquelas que não estão *presentes-em* outras são não acidentais. Entidades não acidentais são universais e descritos como entidades essenciais; entidades não-acidentais são particulares e descritas simplesmente como não essenciais. Reunindo as possibilidades em conjunto, obtém-se um sistema quádruplo de categorias e os nomes para cada uma [Studtmann 2008]:

- Entidades que são ditas-sobre e presentes-em são *universais acidentais*
- Entidades que são ditas-sobre e não presentes-em são *universais essenciais*
- Entidades que não são ditas-sobre e estão presentes-em são *particulares acidentais*
- Entidades nem são ditas-sobre nem estão presentes-em outra coisa são as *substâncias primárias*

Organizadas dessa forma, as entidades do mundo podem ser representadas pelo *Quadrilátero de Aristóteles*, conforme Fig. 3.4.

	Substância (não presente-em um sujeito)	**Acidente** (presente-em um sujeito)
Universal (dito-sobre um sujeito ou predicado de um sujeito)		
Particular (não dito-sobre um sujeito ou predicado de um sujeito)		

Figura 3.4a. Quadrilátero ontológico de Aristóteles
Fonte: adaptado de Smith (2003)

	Substância	**Acidente**
Universal	Segunda substância *homem* *cão* *touro*	Segundo acidente *dor de cabeça* *bronzeado* *pavor*
Particular	Primeira substância *este homem* *este cão* *este touro*	Primeiro acidente *esta dor de cabeça* *este bronzeado* *este pavor*

Figura 3.4b. Quadrilátero ontológico de Aristóteles com exemplos
Fonte: adaptado de Smith (2003)

Um segundo sistema categorias foi proposto por Aristóteles, o qual contém uma lista dos tipos de mais alto nível. Percebendo que os objetos comuns da experiência humana são alocados em classes de crescente generalidade, pode-se observar indícios de que a existência de um tipo mais elevado e abstrato é provável. No entanto, Aristóteles não acreditava em uma categoria de alto nível única, mas em dez delas, como mostrado na Figura 3.5.

Termo Aristotélico	Significado moderno	Exemplo
Ti esti, ousia	Substância	homem
Poson	Quantidade	cinco metros
Poion	Qualidade	branco
Pros ti	Relação	metade
Pou	Local	no mercado
Pote	Data	ontem
Keisthein	Postura	sentado
Echein	Estado	vestido
Poitein	Ação	queimar
Paschein	Sentimento	ser queimado

Figura 3.5. Categorias genéricas de Aristóteles
Fonte: adaptado de Sutcliffe (1993)

A categoria mais importante, *substância*, pode ser entendida listando-se suas características mais significativas [Smith 1997]: substâncias podem existir por conta própria; permanecem numericamente únicas e as mesmas, tendo propriedades diferentes em momentos diferentes; podem participar de relações causais; não têm partes próprias que são substâncias; têm uma contínua; e não têm partes temporais.

Os diferentes tipos de perguntas que teriam sido empregadas para obter a lista de categorias de Aristóteles são perguntas que podem ser feitas sobre algo. Por exemplo, a pergunta "o que é isto?" Só pode ser feita sobre uma substância, e somente respostas descrevendo substâncias são apropriadas [Ackrill 1963]. Independentemente da pergunta, as categorias de Aristóteles são entendidas como categorias de coisas, não de linguagem. Por exemplo, a definição de um tigre não nos diz o significado da palavra "tigre", mas nos diz o que é ser um tigre [Cohen 2008]. Esse fato tem reflexos diretos em atividades de modelagem atuais, sugerindo que para uma boa modelagem, deve-se observar aquelas categorias que tem referente no mundo real.

Outra questão é como e se uma categoria é mais fundamental que outra. Neste contexto, "fundamental" diz respeito a condições de existência e identidade das entidades de uma categoria [Lowe 2007]. Cada entidade tem um recurso fundamental chamado *essência real*. Quando uma entidade possui uma essência específica, significa que ela é de certo tipo; e, para ser de certo tipo, uma entidade deve compartilhar um conjunto de propriedades necessárias e suficientes com outros membros desse mesmo tipo. Em contraste com tais propriedades essenciais, há também propriedades acidentais que uma entidade pode perder sem perder sua essência real [Ackrill 1963]. Por exemplo, uma pessoa é um animal bípede, mas uma pessoa pode perder ambas as pernas e continuar a ser uma pessoa, mantendo a verdadeira essência de uma pessoa. Esse fato também tem reflexos diretos em atividades de modelagem atuais, uma vez que é a forma de se obter uma boa definição para uma entidade.

A tradição aristotélica fornece um método para ordenar categorias de coisas com base em sua essência. O método para distinguir essências usa a distinção gênero-espécie e a divisão

dicotômica [Lennox 2000]. De acordo com a distinção gênero-espécie, a verdadeira essência das espécies é uma combinação de seu gênero e sua diferenciação, que é o critério usado para distinguir uma espécie de outra do mesmo gênero. Por exemplo, a espécie humana pertence ao gênero *Homo* e sua diferenciação é racionalidade; então a verdadeira essência de um ser humano é ser um animal racional. Além disso, através da divisão dicotômica, Aristóteles também propôs a divisão de cada gênero nas entidades que possuem uma diferenciação particular e aquelas que não possuem. Por exemplo, o gênero dos seres vivos é dividido em animais e plantas pela diferenciação do movimento próprio.

Em resumo, a terminologia de origem filosófica que vai influenciar termos usados em modelagem de SIs pode ser organizada da seguinte forma (ver também Figura 3.6).

- *Universais, tipos, tipos naturais ou categorias*: derivados da segunda substância (Figura 3.4a) são abstrações de conjuntos de entidades com características similares que existem no mundo independentemente da mente humana, e que representam todas as entidades daquele tipo que existem, existiram ou vão existir; por ex., pessoas, árvores, bactérias, etc.
- *Classes*: são abstrações de conjuntos de entidades com características similares, mas não que não são naturais, ou seja, são demarcações criadas pela mente humana para um determinado fim; por exemplo, as pessoas que estão presentes a aula (para fins de fazer uma chamada), os carros no pátio da universidade (para fins de credenciamento), etc.
- *Particulares, instâncias ou indivíduos*: derivados da primeira substância (Figura 3.4a) são as entidades ou coisas do mundo em si; por exemplo, o universal "pessoa" pode ser instanciado por José, ou por Maria; o universal "planeta" instanciado por "Terra", etc.
- *Qualidades, propriedades ou atributos*: derivados do primeiro e do segundo acidentes (Figura 3.4a) são características, ou relações unárias, de uma determinada entidade no nível dos universais (segundo acidente) ou nível das instâncias (primeiro acidente); por exemplo, peso e peso do José; vermelho e vermelho deste tomate, etc.
- *Relacionamentos*: conexões mantidas entre entidades, no mínimo binárias, no nível dos universais ou nível das instâncias, além de inter-níveis com universal e instância; por exemplo, vírus causa doença, João namora-com Maria; Pedro é-um professor; etc.

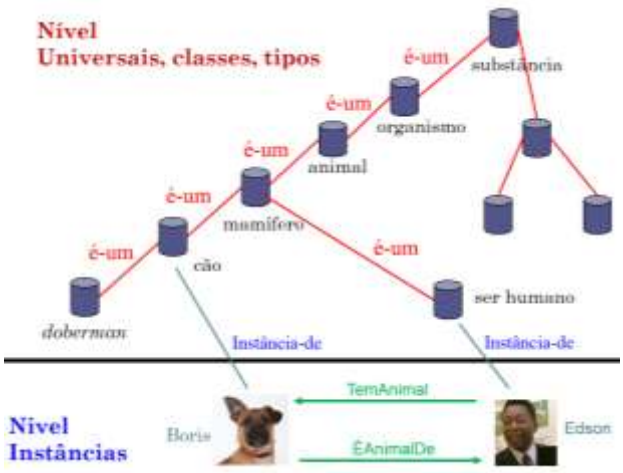


Figura 3.6. Entidades e relações
Fonte: adaptado de Smith (2003)

Nas ontologias como artefatos encontram-se todas essas denominações, e as vezes autores diversos usam uma ou outra notação, ou uma mistura delas. Outro termo também muito usado em ontologias como artefato é "axioma", que consiste de uma sentença declarativa em uma linguagem lógica, que define uma entidade ao impor-lhe restrições de significado. A Figura 3.7 apresenta um axioma da ontologia D-acts [Brochaussen, Almeida, Slaughter 2013] que define a entidade "*identity document*", em um dialeto de lógica descritiva chamado *Manchester Sintax.* Trata-se de uma expressão lógica com quantificadores existenciais, nessa sintaxe representada pelo termo "*some*".

```
document
 and ('is about' some 'documented identity')
 and ('is concretized as' some
   ('specifically dependent continuant'
    and ('inheres in' some ('bearer of' some 'credential role'))))
 and ('is concretized as' only
   ('specifically dependent continuant'
    and ('inheres in' some ('bearer of' some 'credential role'))))
```

Figura 3.7. Expressão lógica para definir "documento de identidade"
Fonte: ontologia dos Atos dos Documentos (D-acts)

Finalmente, mais alguns conceitos relevantes de origem filosófica são necessários para o correto entendimento e uso de modelos ontológicos em SIs. As teorias filosóficas subjacentes à representação são, via de regra, classificadas de acordo com duas posições principais: *realismo* e *anti-realismo*. O debate envolvendo essas correntes já dura séculos e não há consenso [MacLeod, Rubenstein 2005].

O termo realismo tem várias interpretações em Filosofia [Niiniluoto 1999], mas em geral é usado para designar a noção de que existe um mundo físico independente da mente. Entidades independentes da mente são os já mencionados "universais", os quais são instanciadas por "particulares. De acordo com os realistas, por exemplo, a frase "a lua é esférica" é verdadeira independentemente das crenças e das práticas linguísticas de qualquer pessoa.

As primeiras propostas realistas se originaram de Platão e Aristóteles. Para Platão, o conhecimento verdadeiro seria permanente e imutável e, portanto, não poderia ser oriundo de objetos ordinários sujeitos a mudanças constantes. Assim, para Platão, as coisas permanentes, os chamados universais, vêm de um reino de formas que existe à parte do mundo cotidiano. Nesta linha de pensamento, uma "cama" ou uma "mesa" são ideias (ou formas) que as pessoas têm antes mesmo de nascerem. Aristóteles não conectou o reino de formas de Platão com detalhes da vida cotidiana. Propôs que os universais não deveriam ser separados da realidade, mas deveriam ser elementos comuns presentes em particulares da mesma categoria. Por exemplo, o universal "mesa" consiste de todas as características comuns a todas as mesas.

O anti-realismo pode assumir muitas formas, dependendo de qual a dimensão do realismo é rejeitada: ou a existência ou a independência de entidades. Em geral, os anti-realistas são classificados como *nominalistas* e *conceitualistas* [MacLeod, Rubenstein 2005].

Os nominalistas acreditam que existem apenas indivíduos e os problemas de identidade e semelhança podem ser resolvidos a partir de indivíduos e nas relações entre eles. Acreditam que

as semelhanças empíricas entre entidades não são bons critérios para estabelecer a filiação de uma entidade a uma categoria ou para caracterizar um universal. A existência separada de universais, uma visão dualista exibida tanto nas visões de Platão quanto de Aristóteles, é o principal ponto de discordância com o realismo. Nominalistas defendem que os universais são desnecessários e que todo conhecimento provém de entidades particulares oriundo da experiência das pessoas.

Conceitualistas negam que os indivíduos são suficientes para resolver os problemas, mas não apelam para o uso de universais. Em vez disso, conceitualistas explicam os problemas de identidade e de semelhança referindo-se a conceitos ou ideias. Nessa visão, cada palavra da linguagem tem um conceito geral associado a ela. A visão conceitualista parece trazer de volta uma espécie de dualismo, semelhante à visão realista, que estabelece uma diferença entre as coisas e as abstrações sobre coisas, agora sobre outro nome, à saber, conceito.

Apesar de esse tipo de tema parecer um tanto desligado da realidade dos SIs, ele é a base da boa modelagem uma vez que identifica como as entidades do mundo são caracterizadas. O uso prático dos conceitos até aqui explicados será objeto de mais detalhe em seções seguintes.

3.1.3. Representação do conhecimento: categorias básicas para modelagem

O sistema de categorias apresentado na seção anterior (Figura 4b) foi expandido para um sexteto ontológico, de forma a abrigar as considerações sobre entidades *continuantes* e *ocorrentes*, entidades básicas e essenciais para a modelagem ontológica. Essa é a divisão que a ontologia como disciplina (de origem aristotélica) aplica de mundo, ou seja, todas as coisas ou são *continuantes*, entidades que mantém identidade ao longo do tempo, como por exemplo os objetos (pessoas, árvores, etc.); ou são *ocorrentes*, entidades que se alteram ao longo do tempo, como por exemplo os processos (de digestão, de matrícula, etc.). Além disso, o sexteto também diferencia aquelas entidades que dependem de outras para existir, à saber, *continuantes independentes* e *continuantes dependentes* (Figura 3.8).

	Continuante independente	**Continuante dependente**	**Ocorrente (processo)**
Universal	Segunda substância *homem* *gato* *touro*	Segunda qualidade *dor de cabeça* *bronzeado* *pavor*	Segundo processo *caminhar* *pensar* *dormir*
Particular	Primeira substância *este homem* *este gato* *este touro*	Primeira qualidade *esta dor de cabeça* *este bronzeado* *este pavor*	Primeiro processo *este caminhar* *este pensar* *este dormir*

Figura 3.8. Sexteto ontológico estendendo o quadrilátero de Aristóteles
Fonte: adaptado de Smith (2003)

Ontologias de alto nível são ontologias como artefatos desenvolvidos para atender objetivos específicos de modelagem, raciocínio automático e recuperação de informação [Hoehndorf, Schofield, Georgios 2015]. Exemplos de ontologias de alto nível são a *Descriptive*

Ontology for Linguistic and Cognitive Engineering (DOLCE) [Gangemi *et al.* 2002], a *Unified Foundational Ontology* (UFO) [Guizzardi 2005] e a *Basic Formal Ontology* (BFO) [Grenon, Smith, Goldberg 2002].

Adota-se a BFO como ontologia de alto nível, um tipo de metamodelo, uma vez que tem sido amplamente aceita em domínios como Medicina, Biologia, Bioinformática e áreas afins, Direito, Geografia, para mencionar alguns. Como uma ontologia de alto nível, o BFO segue os princípios do realismo já mencionado na seção anterior e representa apenas as categorias mais genéricas, fornecendo meios para categorizar entidades no âmbito da representação [Spear 2006]. A Figura 3.9 apresenta os dois ramos da BFO.

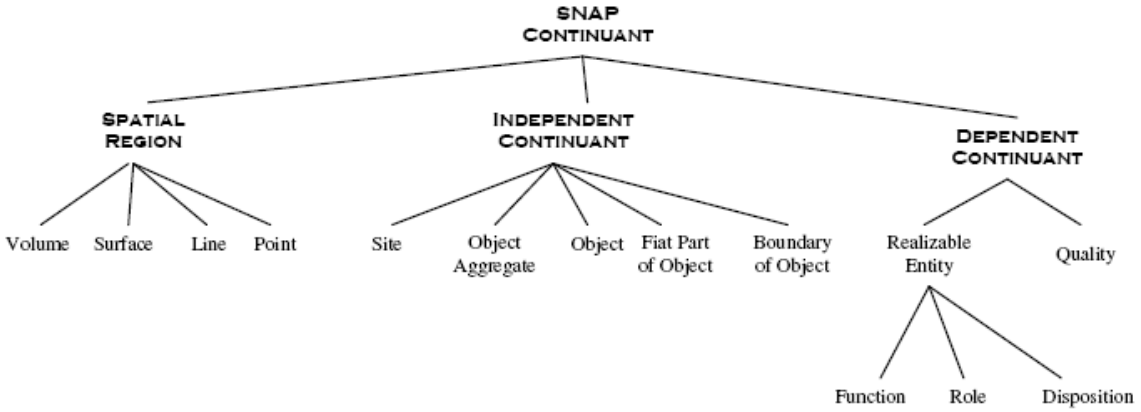


Figura 3.9a. Ontologia de alto nível BFO (versão 1), ramo dos continuantes
Fonte: Spear (2006)

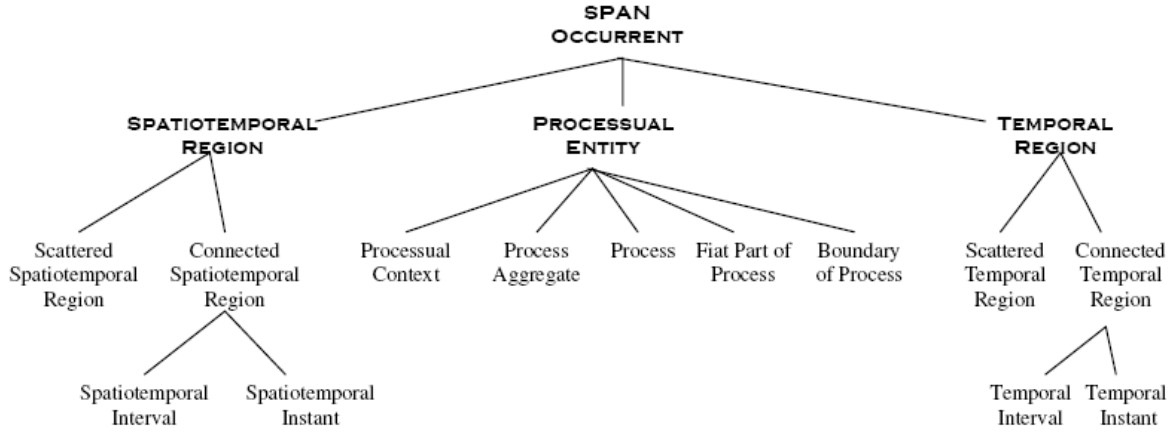


Figura 3.9b. Ontologia de alto nível BFO (versão 1), ramo dos ocorrentes
Fonte: Spear (2006)

O BFO consiste em uma taxonomia que fornece uma categorização das entidades existentes no mundo e, portanto, uma explicação sobre a realidade (Fig. 3.9). O nó raiz da taxonomia do BFO é a entidade mais genérica, identificada apenas como "entidade", e dele derivam dois ramos, à saber, *continuantes* (Fig. 3.9a) e *ocorrentes* (Fig. 3.9b). As Fig. 3.10 (de

3.10a até 3.10f, continuantes) e Fig. 3.11 (de 3.11a até 3.11f, ocorrentes) define cada nó da BFO e dá exemplos de uso [Spear 2006, Smith *et al.* 2007].

	CONTINUANTES **Entidades que mantém identidade ao longo do tempo**
Características	A entidade existe completamente em qualquer período de tempo no qual está presente. A entidade não tem partes temporais.
Exemplos	Uma pessoa, a cor de uma fruta, um conjunto musical, uma lei, o vento na UFMG, ...

Figura 3.10a. Ramo dos continuantes
Fonte: adaptado de Spear (2006))

Distinção entre continuantes	**Região espacial**	**Continuante independente**	**Continuante dependente**
Características	Corresponde a um continuante independente. Não é portador de qualidades. Não é parte natural de outras entidades.	São portadores de qualidades ou propriedades Entidades das quais outras entidades são partes naturais. Entidades que por si próprias não podem ser parte natural em nada. São existencialmente independentes de outras entidades.	São partes naturais ou originadas em outras entidades. São parte de outra entidade dependente ou independente. Exibem algum tipo de dependência existencial: para existir, outras entidades também devem existir.
Exemplos	A soma total de todos os espaços do universo, ...	Um organismo, um coração, uma orquestra, uma perna, uma pessoa...	A cor de uma laranja, o cheiro de pão, a função de ser um professor...

Figura 3.10b. Distinções no ramo dos continuantes
Fonte: adaptado de Spear (2006)

Distinção entre região espacial	**Volume**	**Superfície**	**Linha**	**Ponto**
Características	Uma região espacial com três dimensões	Uma região espacial com duas dimensões	Uma região espacial com uma dimensão	Uma região espacial sem dimensões
Exemplos	Uma parte do espaço em forma de cubo, em forma de esfera, ...	A superfície de uma parte do espaço em forma de cubo, em forma de esfera, ...	Uma aresta de uma parte do espaço em forma de cubo, ...	-

Figura 3.10c. Distinções no ramo das regiões espaciais
Fonte: adaptado de Spear (2006)

Distinção entre continuantes dependente	Entidade realizável	Qualidade
Características	São continuantes dependentes, partes naturais de continuantes que não são exibidas a cada momento em que fazem parte de uma entidade ou grupo de entidades. A exibição de uma entidade realizável corresponde a uma manifestação particular ou processo que ocorre sob certas circunstâncias. São entidades cuja vida contém períodos de atualização, com transformações em seus portadores; e períodos de latência, quando existem em seus portadores, mas não se manifestam.	É um continuante dependente que é exibido caso seja parte natural de uma entidade ou entidades. Podem ser partes naturais de outras entidades; para que a qualidade exista, essas entidades também devem existir.
Exemplos	O papel de ser um professor, a função dos órgãos respiratórios, a disposição do metal para conduzir eletricidade, a disposição do sangue para coagular, a fragilidade de um vaso, ...	A cor de uma laranja, a temperatura ambiente, a forma de uma orelha, a massa de um pedaço de ouro, o peso de um ser, ...

Figura 3.10d. Distinções no ramo das continuantes dependentes
Fonte: adaptado de Spear (2006)

Distinção continuantes independentes	Local	Agregado	Objeto	Parte fiat do objeto	Limite do objeto
Características	É um continuante independente que consiste de uma forma espacial característica em relação a um arranjo de outros continuantes. Distinto de região espacial	É um continuante independente que corresponde a uma soma de objetos separados. Possuem um grau de unidade mais fraco do que o dos objetos. Possuem limites não conectados.	É um continuante independente com extensão auto-contida. A identidade independe de outras entidades e pode ser mantida ao longo do tempo a despeito de perdas e ganhos.	É um continuante independente que é parte de um objeto, mas não é demarcado por falta de continuidade física.	É um continuante independente que é a parte dimensional inferior de outros continuante independentes.
Exemplos	Uma cidade, uma veia, um ambiente, a localização de uma guerra, o quarto onde alguém está, ...	Uma pilha de pedras, um grupo de pacientes de um hospital, uma coleção de livros, uma matilha de cães, ...	Um organismo, uma cadeira, uma célula, uma maça, uma dobradiça, uma pedra, ...	O lado oeste de São Paulo, as superfícies dorsal e ventral do corpo, uma amostra de tecido, ...	A superfície da pele, a superfície da terra, a superfície externa de uma célula, ...

Figura 3.10e. Distinções no ramo das regiões espaciais
Fonte: adaptado de Spear (2006)

Distinção entre entidades realizáveis	Função	Papel	Disposição
Características	É uma entidade realizável cuja manifestação é uma atividade de um continuante direcionada para um fim específico. A finalidade é definida em virtude do continuante ser um tipo específico de entidade no contexto.	É uma entidade realizável não essencial para um continuante, ou seja, cuja manifestação gera resultados que não são essenciais para a identidade. Podem participar no tipo de continuante em contextos sociais e institucionais	É uma entidade realizável que causa um processo específico ou um tipo de transformação no objeto do qual é parte natural. A transformação ocorre apenas sob certas circunstâncias e junto a certas leis naturais, de forma que são entidades frágeis
Exemplos	A função de um coração em bombear sangue, a função de reprodução, a função de um martelo em uma obra, a função de juiz, ...	O papel de uma pessoa como advogado, o papel de uma droga no tratamento de uma doença, o papel de um arvore no ecossistema, ...	Alimentos que estragam se não armazenados sob refrigeração, a disposição do sangue em coagular, a disposição de um metal em conduzir eletricidade, ...

Figura 3.10f. Distinções no ramo das entidades realizáveis
Fonte: adaptado de Spear (2006)

	OCORRENTES Entidades que se alteram ao longo do tempo
Características	A entidade se desdobra ao longo de um período de tempo.
Exemplos	A respiração, o funcionamento de um órgão do corpo, parte da vida de Einstein, ...

Figura 3.11a. Ramo dos ocorrentes
Fonte: adaptado de Spear (2006)

Distinção entre ocorrentes	Região espaço-temporal	Entidade processual	Região temporal
Características	É uma entidade ocorrente na qual entidades processuais podem estar localizadas.	Uma entidade ocorrente que existe no tempo, tem partes temporais e depende de um continuante. A característica é a existência de partes temporais e espaciais.	É uma entidade ocorrente que é parte do tempo.
Exemplos	A região espaço-temporal ocupada por uma vida, pelo desenvolvimento de um tumor, ...	A vida de um organismo, o processo de meiose, o curso de uma doença, o voo de um pássaro, ...	O tempo para correr uma maratona, a duração de uma cirurgia, o momento da morte.

Figura 3.11b. Distinções no ramo dos ocorrentes
Fonte: adaptado de Spear (2006)

Distinção entre regiões temporais	Região temporal distribuída	Região temporal conectada
Características	É uma região do espaço que tem dimensões espaço-temporais. Cada ponto espacial e temporal não é conectado com outro ponto espacial e temporal.	É uma região que tem dimensões temporais e espaciais tal que todos os pontos dentre da região espaço temporal são imediatamente conectadas a todos ou outros pontos dentro da mesma região espaço temporal
Exemplos	O espaço e o tempo ocupados por jogos individuais da copa do mundo, ...	A localização espacial e temporal da vida de um organismo, a localização espacial e temporal do desenvolvimento de um feto, ...

Figura 3.11c. Ramo dos ocorrentes
Fonte: adaptado de Spear (2006)

Distinção reg. temporais conectadas	Intervalo temporal	Instante temporal
Características	Uma região temporal conectada que dura por mais do que um único momento de tempo.	uma região temporal conectada contendo um simples momento de tempo.
Exemplos	Qualquer duração temporal contínua na qual um processo ocorre.	O momento de nascimento de uma criança, o momento da morte.

Figura 3.11d – Distinção entre entidades processuais
Fonte: adaptado de Spear (2006)

Distinção entidades processuais	Contexto processual	Processo agregado	Processo	Parte fiat de processo	Limite do processo
Características	É um ocorrente que consiste de uma forma característica espacial inserida em algum arranjo de outras entidades ocorrentes. Entidades nas quais ocorrentes podem estar localizados...	Um processo agregado é a soma das partes de processos. Não possui limites não conectados.	É uma entidade processual que é um todo maximamente conectado espacial e temporalmente. Possui início e fim *bona-fide* que corresponde a descontinuidades	É uma entidade processual que é parte de um processo. Não tem início e fim *bona-fide* e corresponde a descontinuida des reais.	É uma entidade processual que é o limite temporal instantâneo *bona-fide* ou fiat de um processo.
Exemplos	Uma operação cirúrgica como contexto processual para uma infecção, um check-up de rotina para encontrar uma doença, ...	O bater do coração de cada um dos sete indivíduos em um quarto, ...	A vida de um organismo, o processo de dormir, o processo de divisão celular,	Mascar chiclete durante uma refeição, o meio de uma tempestade, a pior parte de um ataque cardíaco, ...	A separação de um dedo em um acidente industrial, uma incisão no início da cirurgia, ...

Figura 3.11e. Distinção entre entidades processuais
Fonte: adaptado de Spear (2006)

Distinção reg. espaço temporal conectada	Intervalo espaço temporal	Instante espaço temporal
Características	Uma região do tempo e espaço conectados que se mantem por mais de um simples momento de tempo.	Uma região do tempo e espaço em um momento específico.
Exemplos	A região de espaço e tempo ocupada por um processo, ou por a parte fiat de um processo.	A região de espaço tempo ocupada por uma única parte temporal de um processo.

Figura 3.11f. Distinção entre regiões espaço temporais conectadas
Fonte: adaptado de Spear (2006)

As categorias filosóficas básicas são essenciais para a boa modelagem uma vez que fornecem uma fundamentação sólida para a definição, entendimento e comunicação sobre as entidades que povoam os modelos da realidade. Possibilitam construir uma base formal sobre a qual sistemas podem ser desenvolvidos e mesmo auditados.

3.1.4. Tecnologia e Web Semântica: linguagens e ferramentas

Desde os anos 1960, as aplicações computacionais têm feito uso de bancos de dados com sucesso. A questão sobre o uso de ontologias, ao invés de banco de dados, sempre surge em contextos práticos. Na verdade, ontologias como artefatos construídos sem os princípios da ontologia como disciplina, pouco tem a oferecer mais do que os bancos de dados, os quais alguns autores nomeiam como uma "promiscuidade ontológica". A razão para isso é a natureza *ad-hoc* utilizada pelos desenvolvedores para a criação de sistemas. De fato, faz sentido construir modelos ontológicos bem fundamentados, os quais são passíveis de representação lógica, o que possibilita a redução da ambiguidade e de problemas de comunicação.

Entretanto, a diferença essencial entre as duas tecnologias reside no fato de que bancos de dados são tipos de sistemas de "mundo fechado", em que se assume que o que não é sabido verdadeiro deve ser falso; e ontologias são sistemas conhecidos como de "mundo aberto", em que se pressupõe que o que não é sabido verdadeiro é simplesmente desconhecido. Esse é um apelo para o uso de ontologias em Medicina, onde muito ainda pode ser descoberto e revelado, por exemplo, via inferências automáticas. Cabe assim verificar se o problema em questão exige o uso de ontologias. Não faria sentido, por exemplo, usar um sistema de mundo aberto para descobrir quantos presidentes o Brasil já teve. Outro apelo ao uso de ontologias é de que muitas consultas feitas com a tecnologia da Web Semântica não seriam possíveis em ambientes de bancos de dados.

O uso de motores de inferência se tornou popular nos últimos anos por avanços recentes da Inteligência Artificial (IA). Trata-se de IA uma vez que conhecimento novo é deduzido a partir de conhecimento existente. A figura 3.12a apresenta um fragmento de *Web Ontology Language*[5] (OWL), o qual afirma que um cão é uma subclasse de vertebrado, que vertebrado é uma subclasse de animal equivalente a um animal que tem ossos. O mesmo fragmento é apresentado no editor de

[5] https://www.w3.org/OWL/

ontologias Protégé[6] (Fig. 3.12b). O motor de inferência da ferramenta deduz que não existem cães sem ossos ("*nothing*", na figura Fig. 3.12c).

Dog subclassOf Vertebrate
Vertebrate subclassOf Animal
Vertebra subclassOf Bone
Vertebrate equivalentTo Animal and has-part some Bone

Figura 3.12a. Fragmento OWL descrevendo o cão como um animal com ossos
Fonte: Schulz (2016)

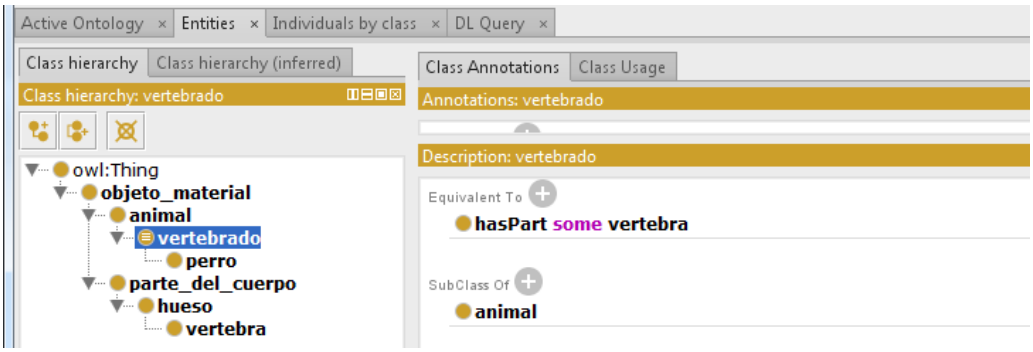


Figura 3.12b. Fragmento OWL no editor de ontologia Protegé
Fonte: Schulz (2016)

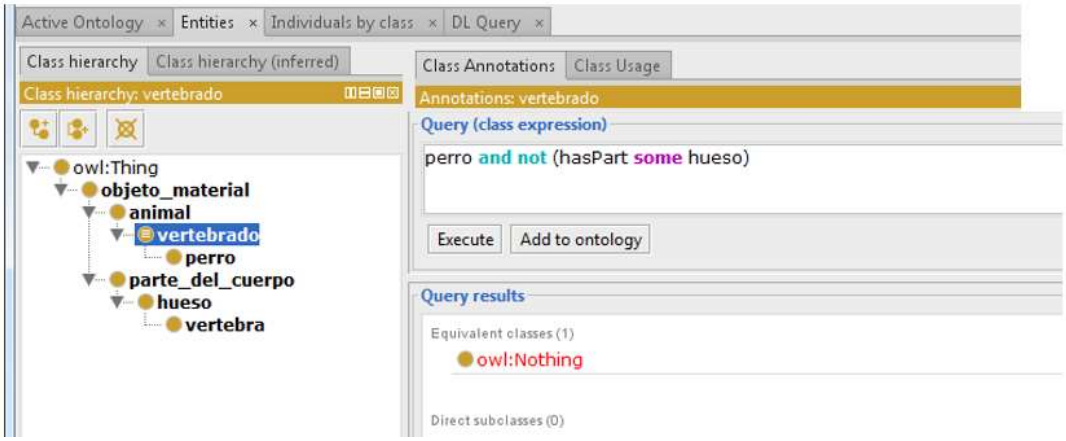


Figura 3.12c. Motor de inferência deduz que não existem cães sem ossos ("*nothing*")
Fonte: Schulz (2016)

[6] https://protege.stanford.edu/

Figura 3.12d. Fragmento OWL no editor de ontologia Protegé
Fonte: Schulz (2016)

Figura 3.12e. Motor de inferência deduz que não existem cães sem ossos (“nothing”)
Fonte: Schulz (2016)

As principais linguagens utilizadas para a construção de ontologias são as *lógicas descritivas* ou *lógicas de descrições* [Baader, Horrocks, Sattler 2002]. Esse tipo de lógica foi projetado para facilitar a tarefa de descrever objetos através de definições e propriedades (ver exemplo Figura 3.7). Trata-se de uma família de lógicas, fragmento da lógica de primeira ordem, que teve origens nas lógicas terminológicas dos anos 1990 [Baaeder *et al.* 1992] e vem se desenvolvendo na esteira das linguagens de marcação da Web Semântica.

As linguagens de marcação surgiram nos anos 1970 como um conjunto de convenções para a codificação de textos eletrônicos que especificava as marcas permitidas, as marcas exigidas, a distinção entre as marcas e o texto, e o significado da marcação. A primeira linguagem de marcação foi o *Standard Generalized Markup Language* (SGML), um padrão internacional (ISO-8879) criado para marcar arquivos eletrônicos fornecendo instruções sobre como o texto deveria ser representado.

O SGML fornecia marcas para elementos convencionais dos textos como pontuação, letras maiúsculas e minúsculas, espaços, cabeçalhos, parágrafos, sentenças, etc. No SGML e linguagens subsequentes, a parte marcada é a instância de texto a representar, circundada por elementos com a marca inicial “<” e marca final “>”, conforme exemplo na Figura 3.12.

Quem é capaz de entender a sintaxe da SGML é também capaz de lidar com duas de suas principais aplicações mais populares:

- *Hypertext Markup Language* (HTML), uma aplicação específica do SGML utilizada na Internet, a partir de 1989;
- *Extended Markup Language* (XML), uma versão abreviada da SGML que permite ao autor especificar a forma dos dados no documento;

```
<antologia>
    <poema>
      <titulo> Tenho Tanto Sentimento </titulo>
          <verso>
              <linha> Tenho tanto sentimento </linha>
              <linha> Que é freqüente persuadir-me </linha>
              <linha> De que sou sentimental, </linha>
              <linha> Mas reconheço, ao medir-me, </linha>
                ...
          </verso>
          <verso>
              <linha> Qual porém é a verdadeira</linha>
              <linha> E qual errada, ninguém </linha>
              <linha> Nos saberá explicar; </linha>
              <linha> E vivemos de maneira <linha>
                ...
          </verso>
    </poema>
              <!-- mais poemas aqui    -->
</antologia>
```

Figura 3.13. Marcação SGML em texto de poema
Fonte: dos autores

Utilizada a partir de 1998, a XML costuma ser apontado como o início da Web Semântica [Berners-Lee, Hendler, Lassila 2001], onde se buscava melhoria na recuperação da informação via o fornecimento de metadados ao conteúdo textual.

```
<produto>
      <nome língua = "inglês" > book </nome>
      <preço moeda = "dólar" > 45,00 </preço>
      <fornecedor formato = "XLB56" língua = "inglês">
            <rua> Penbridge Square </rua>
            <número> 30 </número>
            <cep> 92310 </cep>
            <país> United Kingdom </país>
      </fornecedor>
</produto>
```

Figura 3.14. Marcação XML com atributos
Fonte: os autores

A evolução do XML não foi exatamente uma linguagem de marcação, mas um padrão de metadados denominado *Resource Description Framework* (RDF). O RDF buscava resolver limitações de expressão do XML impondo limitações estruturais aos dados, proporcionando assim

métodos para expressão semântica e para a publicação de vocabulários legíveis por pessoas e por computadores, bem como a reutilização e a distribuição dos metadados para diferentes comunidades.

O RDF é de um conjunto de descritores organizados segundo um modelo de dados e empregado para descrever entidades de um domínio. O modelo de dados se vale da sintaxe XML e é baseado em triplas *recurso-propriedade-valor* (Figura 3.15) usadas para identificar, respectivamente, entidades, atributos e relações, e instâncias a representar.

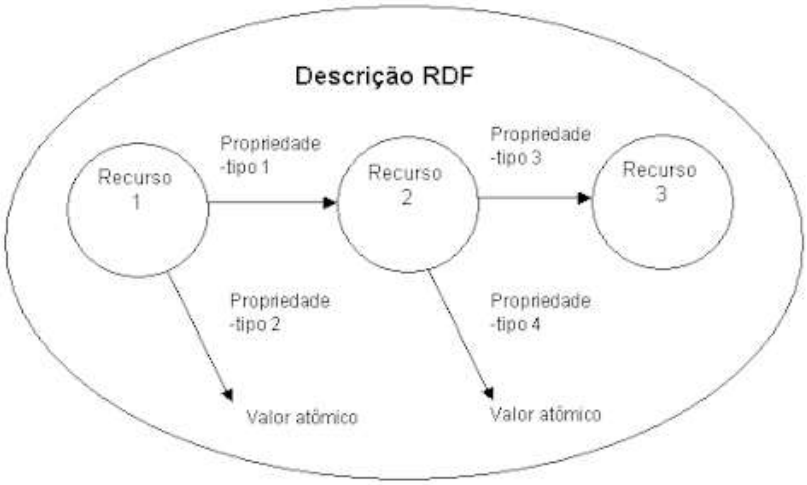


Figura 3.15. Modelo de dados RDF com a tripla objeto
Fonte: os autores

O RDF é utilizado para representar um modelo de dados e armazenar instâncias desse modelo em arquivos eletrônicos, possibilitando o intercâmbio de instâncias entre aplicações. O padrão impõe uma estrutura formal ao XML e proporciona melhor representação semântica.

A aplicação do modelo de dados RDF na questão semântica pode ser melhor entendido por um exemplo simples. Sejam as declarações (a) e (b):

a) O autor do documento X é João Silva

b) João Silva é o autor do documento X

Para uma pessoa, as declarações têm o mesmo significado, ou seja, que João Silva é o autor do documento de título X. Entretanto, para um sistema as declarações são conjuntos de caracteres diferentes e é necessário um modelo de dados para a expressão semântica, passível de utilização por sistemas em computadores. O modelo de dados para a declaração "o autor do documento 1 é João Silva" possui: i) um único recurso chamado "documento X"; ii) uma propriedade-tipo chamada "autor"; e um instância ou valor correspondente "João Silva". A Figura 3.16 apresenta um exemplo de marcação em RDF e o respectivo modelo de dados.


```
</rdf:RDF>
<rdf:Description
      xmlns:rdf="http://www.w3c.org/1999/02/22-rdf-sintax-ns#"
      xmlns:dc="http://purl.org/metadata/dublin_core#"
      rdf:about="http://jornal.com.br/artigos/a4787.html">
            <dc:creator>
                  <rdf:Description>
                        <x:nome>Jose Silva </x:nome>
                        <x:email>js@jornal.com.br</x:email>
                        <x:cargo> editor </x:cargo>
                  </rdf:Description>
            </dc:creator>
            <dc:publisher> Jornal "O Diário" </dc:publisher>
</rdf:Description>
</rdf:RDF>
```

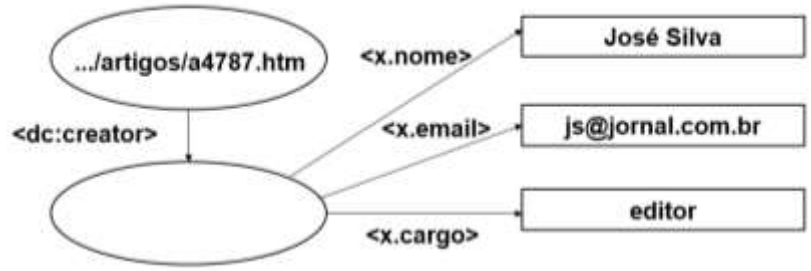


Figura 3.16. Modelo diagramático e marcação RDF
Fonte: os autores

A evolução do RDF é uma variação conhecida como *Resource Description Framework Schema* (RDFS). É a partir RDFS que se torna possível construir ontologias em função, pela primeira vez, da presença dos constructos "*class*", "*subclass*", "*type*" e "*property*". Além disso, o RDFS contém restrições adicionais:

- "*domain*", que especifica o domínio de uma propriedade P, ou seja, a classe dos recursos que podem aparecer como objetos em uma tripla da propriedade P;
- "*range*", que especifica a faixa de valores para uma propriedade P, ou seja, a classe dos recursos que pode aparecer como valores em uma tripla da propriedade P.

A figura 3.17 apresenta um exemplo de marcação RDFS, escrito na linguagem RDF:


```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-sintax-ns#"
 xmlns:rdf="http://www.w3.org/2000/01/rdf-schema#">
   <rdf:Class
       rdf:about="http://www.schemas.org/2001/01/rdf-schema#Pessoa">
   </rdf:class>
   <rdf:Class
      rdf:about="http://www.jornal.com.br/.../rdf-schema#Politico">
      <rdf:SubClassOf
         rdf:resource="http://www.schemas.org/2001/01/rdf-schema#Pessoa">
      </rdf:SubClassOf>
   </rdfs:class>
   <rdf:Class
      rdf:about="http://www.jlocal.com.br/.../rdf-schema#Candidato">
      <rdf:SubClassOf
          rdf:resource="http://www.schemas.org/.../rdf-schema#Politico"
      </rdf:SubClassOf>
</rdf:RDF>
```

Figura 3.17. Marcação RDFS com os constructos "*class*" e "*subclass*"
Fonte: os autores

RDF e RFDS não permitem definir escopo local de propriedades, disjunção de classes, combinação booleana de classes, restrições de cardinalidade e características (matemáticas) de propriedades. Por essas deficiências, desenvolveu-se a OWL, uma linguagem para representação de ontologias com as seguintes características:

- Sintaxe bem definida;
- Semântica formal bem definida;
- Suporte a inferências automáticas:
    - Classificação automática: verificar se uma entidade pertence a uma categoria;
    - Subsunção: verificar se uma categoria é subconjunto de outra pela comparação de suas de suas definições.

A Figura 3.18 apresenta um exemplo de marcação OWL, com a restrição de que só professores são permitidos como valores para a propriedade *isTaughtB*y.

```
<owl:Class rdf :about="#firstYearCourse">
<rdfs : subclassof >
    <owl:Restriction>
    <owl :onproperty rdf : resource="#isTaughtBy"/>
    <owl:allValuesFrom rdf :resource="#Professor"/>
    <owl:Restriction>
    </rdfs : subclassof >
<owl :Class>
```

Figura 3.18. Fragmento OWL com restrições de propriedade
Fonte: os autores

Para lidar com o balanço expressividade versus tratabilidade, a OWL foi definida em três versões, da mais expressiva para a menos: OWL Full, OWL DL (*description logics*) e OWL Lite. Os elementos OWL permitem ainda:

- comentários e definições (*owl:comments*);
- controle de versão (*owl:priorVersion, owl:versionInfo, owl:backwardCompatibleWith*);
- classes equivalentes (*owl:equivalent*Class);
- inclusão de outras ontologias (*owl:imports*);
- propriedades unárias e binárias (*owl:datatypeProperty* e *owl:objectProperty*).

Além disso, a OWL contém os quantificadores lógicos – universal e existencial – que atuam como restrições sobre propriedades e são definidos, respectivamente, pelos elementos *owl:allValuesFrom* e *owl:someValuesFrom.*

O elemento *owl:allValuesFrom* (Fig. 3.19) corresponde ao conjunto de indivíduos (uma classe anônima), os quais, para uma dada propriedade (prop), tem relacionamento apenas com outros indivíduos membros de uma classe específica (classe A).

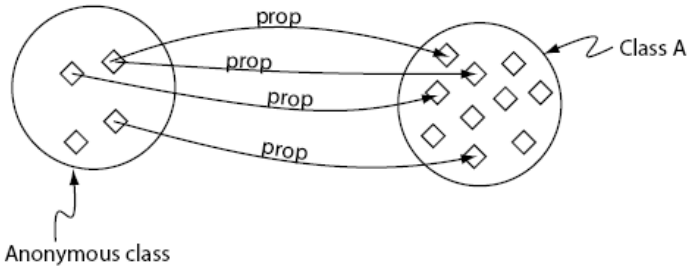


Figura 3.19. Representação do Quantificador universal a*llValuesFrom*
Fonte: Horridge (2011)

O elemento *owl:someValuesFrom* (Fig. 3.20) descreve a classe (anônima) de indivíduos, os quais tem pelo menos um tipo de relacionamento através da propriedade (prop) com indivíduos membros de uma classe (classe A).

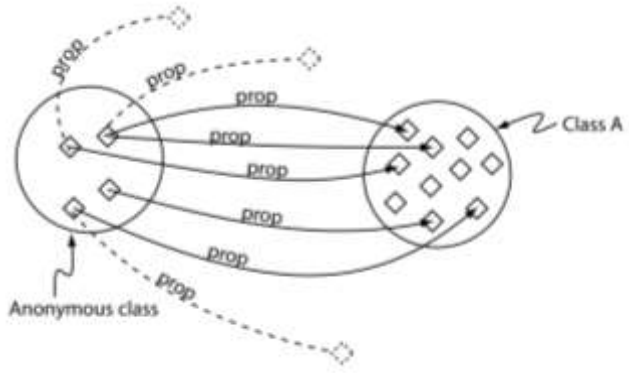
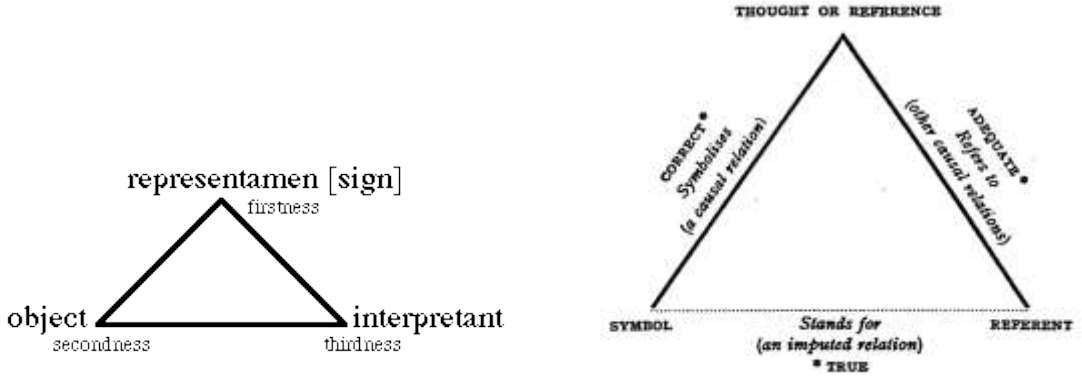


Figura 20. Representação do Quantificador Existencial *someValuesFrom*
Fonte: Horridge (2011)

3.1.3. Aplicações de ontologias em biomedicina

De forma a apresentar a infinidade de aplicações na área médica com o uso de ontologia, em primeiro lugar é preciso entender onde em uma arquitetura de sistemas médicos uma ontologia como artefato pode ser útil em questões de integração e interoperabilidade.

Para isso, descrevem-se níveis de conhecimentos a partir de visão adaptada da Teoria Semiótica de Peirce[7]. A teoria semiótica explica o processo pelo qual o significado é gerado através da percepção e da interpretação de dados sensoriais. Para explicar sua teoria, Peirce criou um triangulo que ficou conhecido genericamente como Triângulo do Significado (Figura 3.21a) contendo um objeto, um signo e o conceito de quem interpreta. A variação mais conhecida desse triangulo é o de Ogden e Richards (1923) (Figura 3.21b).



(a) (b)

Figura 3.21. Triângulo do Significado original e sua variante linguística
Fonte: Wikipédia

Independentemente do triângulo adotado, a ideia é que a observação de um símbolo traz a mente um conceito (pensamento sobre o referente) pela lembrança de a pessoa já ter visto o objeto (referente). As arestas também tem significados, conforme Figura 3.21b.

Adota-se aqui um a variação proposta por Schulz (2016), que inclui "símbolos", "universais" e "indivíduos" (Figura 3.22). Pode-se atribuir exemplos a cada vértice, e através desses exemplos explicar as formas de aquisição de conhecimento para utilizar ontologias como artefatos em SIs médicos. Essas formas de aquisição de conhecimento, vamos chamar de "níveis de conhecimento". Por exemplo, universal "cão" corresponde a todos os cães que existem ou existiram sobre a terra; os símbolos denotam indivíduos, como por exemplo, a palavra "Marley" denota um certo cachorro; e os símbolos também denotam universais, por exemplo, "cão" denota cães em geral. Uma denotação de uma palavra é o conjunto de todas as coisas individuais as quais aquela palavra pode ser usada para se referir. Os indivíduos instanciam os universais, por exemplo, o cão "Marley" é um indivíduo que instancia o universal "cão".

[7] Charles Sanders Peirce – filósofo norte americano, 1839–1914

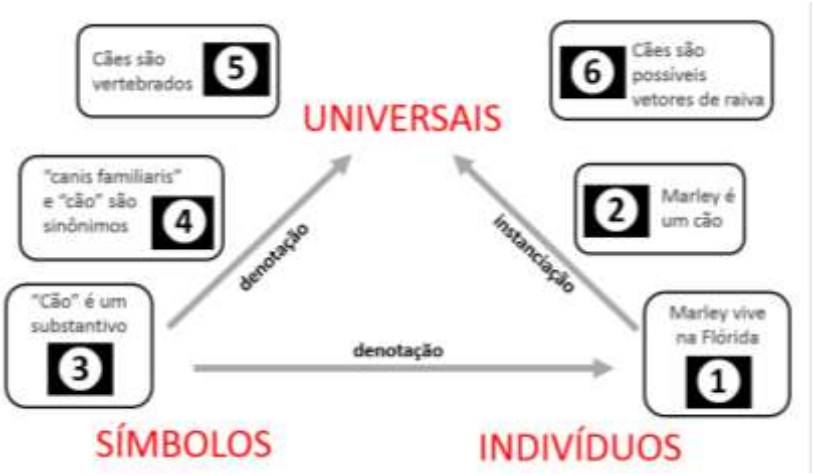


Figura 3.22. Triângulo proposto por Schulz (2016)

Atribui-se então a cada aresta do triângulo um "nível de conhecimento" que é utilizado adiante para organizar as melhores formas de tratar os recursos presentes em SIs médicos. Esses níveis correspondem a formas em que o conhecimento é apresentado (Figura 3.23):

- *Nível linguístico*, que abriga as declarações sobre propriedades da linguagem, e não especificamente das entidades; aqui podem ser tratadas questões como sinonímia, polissemia, e partes dos discursos;
- *Nível factual*, dos fatos realidade, por exemplo, que Marley é um cão que vive na Flórida;
- *Nível contingente*, que abriga as modalidades, o que é possível e típico, por exemplo, que cães são possíveis transmissores de raiva;
- *Nível ontológico*, que abriga as declarações universais, validas em qualquer contexto, por exemplo, que uma hepatite só ocorre no fígado.

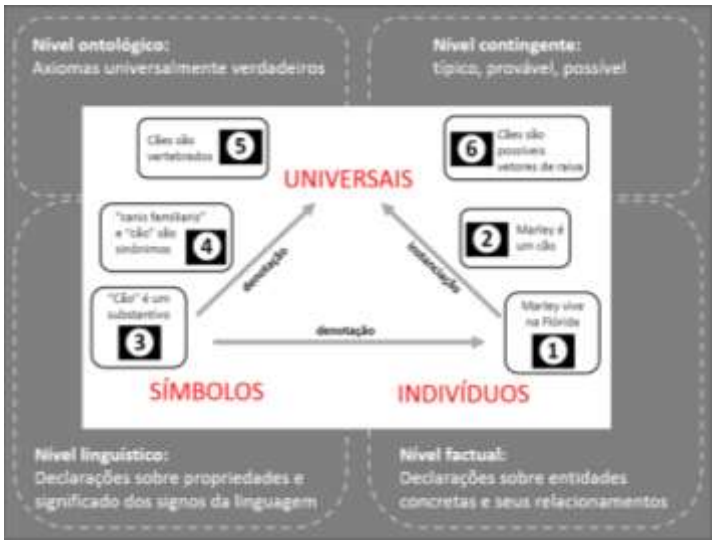


Figura 3.23. Níveis ou formas em que o conhecimento médico se manifesta
Fonte: adaptado de Schulz (2016)

Nível factual:

A linguagem natural parece ser o modo mais efetivo de comunicação entre profissionais de saúde. Mesmo não essa não seja a melhor forma para os SIs, é preciso criar infraestruturas para extração de dados que trabalham com dados não estruturados. Para isso é preciso realizar extração de dados e conexão com terminologias médicas padronizadas (Figura 3.24).

O texto que representa um prontuário de paciente com *myasthenia gravis*, possui termos que podem ser ancorados a vocabulários biomédicos, nesse caso, a SNOMED-CT[8]. Enquanto o prontuário é preenchido, identificadores são automaticamente detectados adicionando metadados ao documento. O texto é segmentado para a criação de uma estrutura semântica onde contextos específicos são identificados, como negações, hipóteses ou planos.

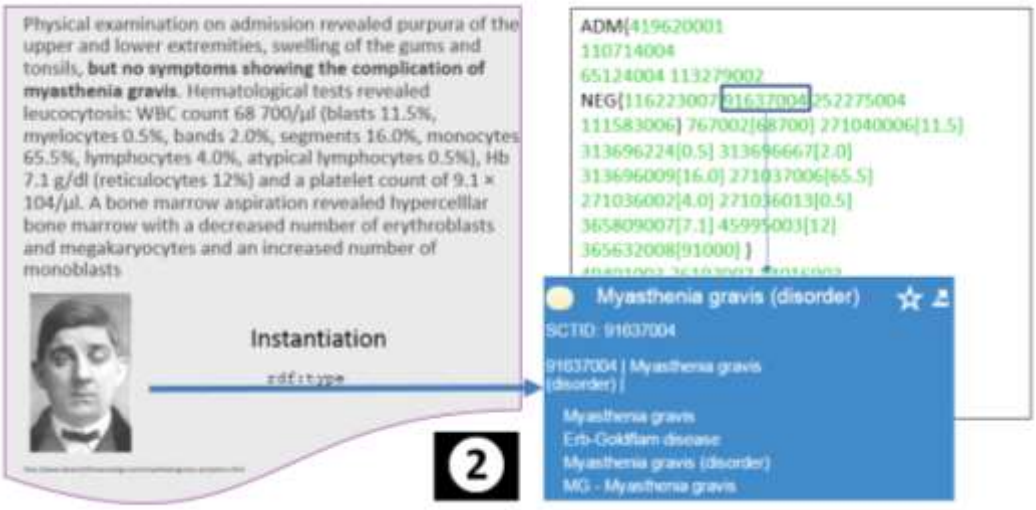


Figura 3.24. Linguagem natural ancorada à terminologia padronizada
Fonte: Schulz et al. (2009)

Nível linguístico:

Um *Part-Of-Speech Tagger* (POS *Tagger*) é um software que lê um texto e atribui partes do discurso à cada palavra, tal como substantivo, verbo, adjetivo, etc. Esse tipo de classificação, serve de suporte a várias possibilidades de tratamento de massas textuais, como por exemplo, o uso na construção de ontologias e o uso em algoritmos de *machine learning*.

A Figura 3.25 mostra um exemplo com os códigos do Quadro 3.2.

[8] http://www.snomed.org/

Quadro 3.2. Códigos do POS-Tagger

CC Coordinating conjunction	*NN Noun, singular or mass*	*SYM Symbol*
CD Cardinal number	*NNP Proper noun, singular*	*UH Interjection*
DT Determiner	*NNPS Proper noun, plural*	*VB Verb, base form*
EX Existential there	*PDT Predeterminer*	*VBD Verb, past tense*
FW Foreign word	*POS Possessive ending*	*VBG Verb, gerund or present p.*
IN Preposition or subordinating	*PRP Personal pronoun*	*VBN Verb, past participle*
JJ Adjective	*PRP$ Possessive pronoun*	*VBZ Verb, 3rd person singular p.*
JJR Adjective, comparative	*RB Adverb*	*WDT Wh-determiner*
JJS Adjective, superlative	*RBR Adverb, comparative*	*WP Wh-pronoun*
LS List item marker	*RBS Adverb, superlative*	*WP$ Possessive wh-pronoun*
MD Modal	*RP Particle*	*WRB Wh-adverb*

Fonte: Cognitive Computation Group @ Illinois[9]

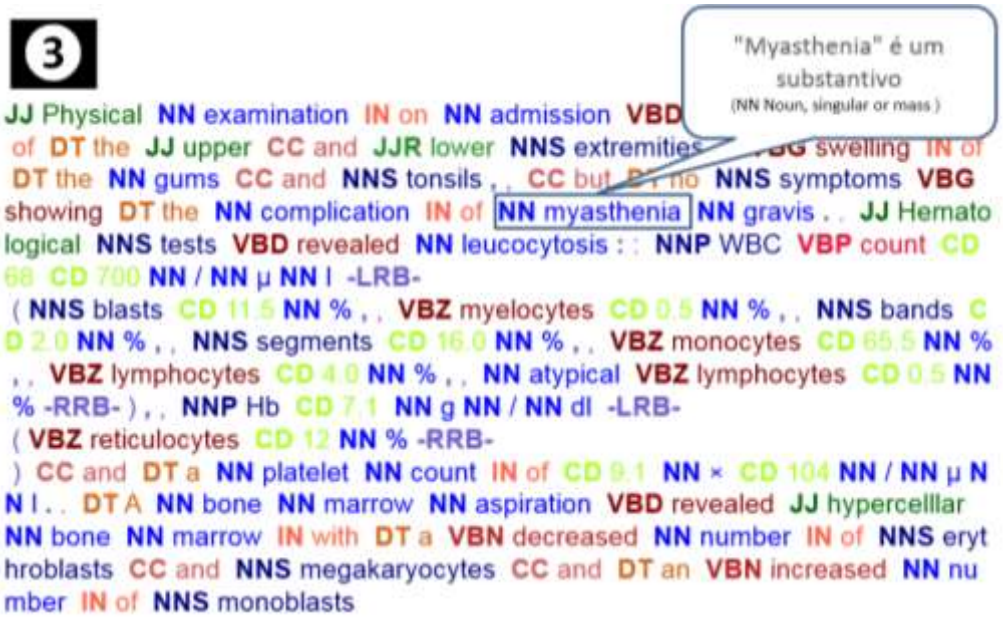


Figura 3.25. Partes do discurso marcadas automaticamente em um prontuário
Fonte: *Cognitive Computation Group @ Illinois*

Nível contingente:

O MeSH[10] é um vocabulário controlado de medicina usado para indexação de documentos e recuperação da informação. De acordo como MeSH, uma "desordem bipolar" pode ser classificada como terapia, ou prevenção ou complicação (Figura 3.26). São possibilidades, ou seja, exemplos de declarações contingentes no mundo dos SIs médicos

[9] http://cogcomp.cs.illinois.edu/

[10] https://www.ncbi.nlm.nih.gov/mesh

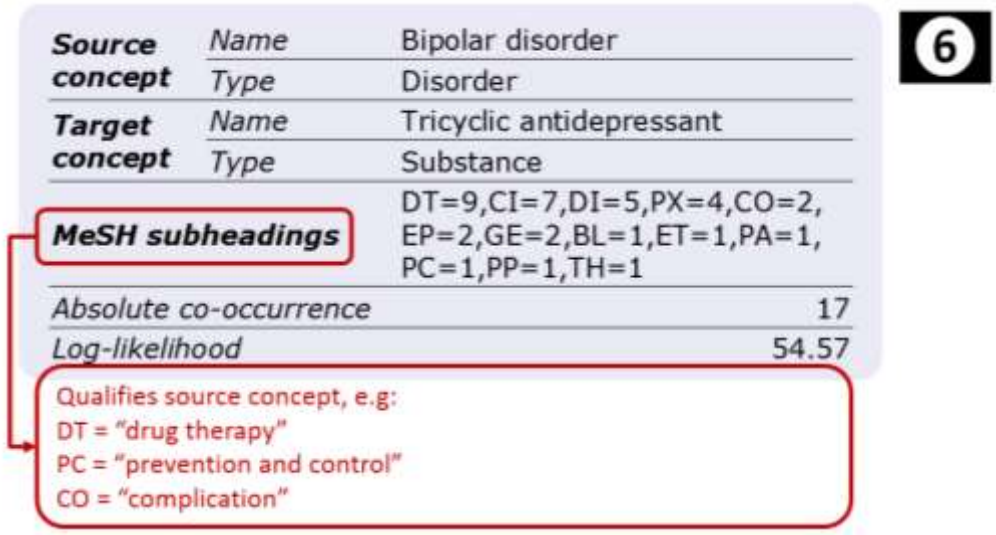


Figura 3.26. Declarações contingentes em vocabulário médico padronizado
Fonte: Miñarro-Giménez et al. (2015)

Nível ontológico:

No nível ontológico existem declarações formais que favorecem a integração de dados, uma vez que definem explicitamente a semântica dos termos. Na verdade, axiomas formais restringem o significado dos termos, por exemplo:

- *Nucleous is-part-of cell*
- *Nucleous is-contained-in cell*
- *Nucleous is-component-of cell*

Axiomas reais, em um editor de ontologias, foram apresentados anteriormente na Figura 12. Os modelos ontológicos podem resultar em benefícios reais em termos de interoperabilidade semântica visto a redução de ambiguidade e a comunicação precisa que podem prover. Entretanto, modelos ontológicos não são uma panaceia que resolve todos os problemas de integração de sistemas. Na verdade, existem problemas com a representação em ontologias:

- Fornecem apenas representações parciais da realidade
- Não possuem a expressividade necessária para criar definições completas de entidades em todos os casos
- Dificuldade para lidar com "*background knowledge*", por exemplo:
    - A declaração *"hand has-part thumb"* descarta a possibilidade de uma mão sem polegar, por exemplo, depois de um acidente;
    - A declaração *"hypertensive_Disease is-a Risk_Factor_for_Aneurysm_Rupture"* é verdadeira apenas em certas circunstâncias, não em todos os casos

Uma vez apresentados os quatro níveis de conhecimento que podem ser encontrados em SIs médicos, é possível desenhar uma arquitetura de sistemas (Figura 26) que faça uso do

conhecimento disponível de acordo com suas características e de forma a atender as questões para o qual o SI foi planejado.

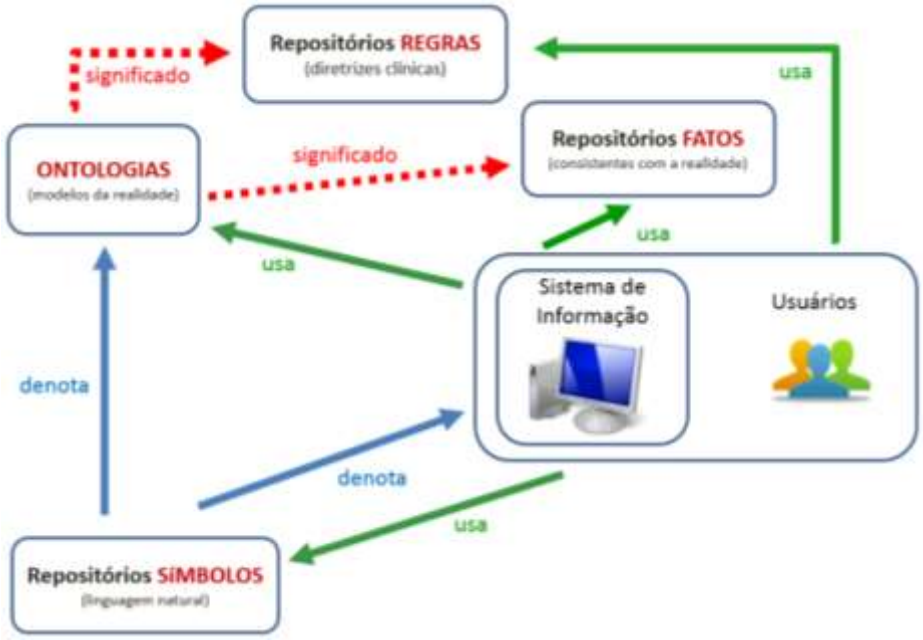


Figura 3.27. arquitetura geral de sistemas médicos: conectando tudo
Fonte: os autores

3.2. Práticas em construção de ontologia.

Na construção de artefatos ontológicos, existem diretrizes que orientam a construção de bons artefatos, a seguir, apresenta-se uma breve explicação de três princípios sugeridos por [Schulz *et al.* 2012]:

- Os artefatos ontológicos devem ser formais, ou seja, suas entidades (termos) são representadas sem ambiguidade de significado e a especificação das relações entre os termos incluem axiomas lógicos, permitindo a automatização tanto da recuperação quanto do raciocínio.
- As ontologias usam especificações explícitas permitindo que as pessoas raciocinem e entendam o domínio representado pela ontologia.
- As ontologias devem ser adequadas ao domínio alvo da representação, significa que a ontologia construída deve representar a realidade e o conhecimento presentes no domínio desejado.

Existem várias metodologias para o desenvolvimento de ontologias, sem um consenso sobre qual metodologia é mais apropriada. A lista abaixo apresenta algumas das metodologias mais conhecidas de meados dos anos 90 até os dias atuais:

- *Toronto Virtual Enterprise* (TOVE) [Grüninger, Fox 1995].
- *Methontology* [Fernández-López *et al.* 1997].

- Metodologia *NeOn (Network Ontology)* [Suárez-Figueroa 2010].
- *Systematic Approach for Building Ontologies* (SABiO) [Falbo 2014].
- *Up for ONtology (*UPON) [De Nicola *et al.* 2005].
- Metodologia do realismo ontológico [Arp *et al.* 2015, Smith, Ceusters 2010].

Algumas metodologias focam em aspectos semânticos para a construção de artefatos ontológicos, como é o caso da metodologia do realismo ontológico, e outras focam em aspectos do processo de engenharia de ontologias, como a metodologia NeOn que tem como base o processo de desenvolvimento de software. Para maximizar a aderência as diretrizes apresentadas por Schulz *et al.* (2012), o ideal combinar duas ou mais metodologias como apresentado por Farinelli e Elkin (2017) e Farinelli (2017) e ilustrado na Figura 3.28.

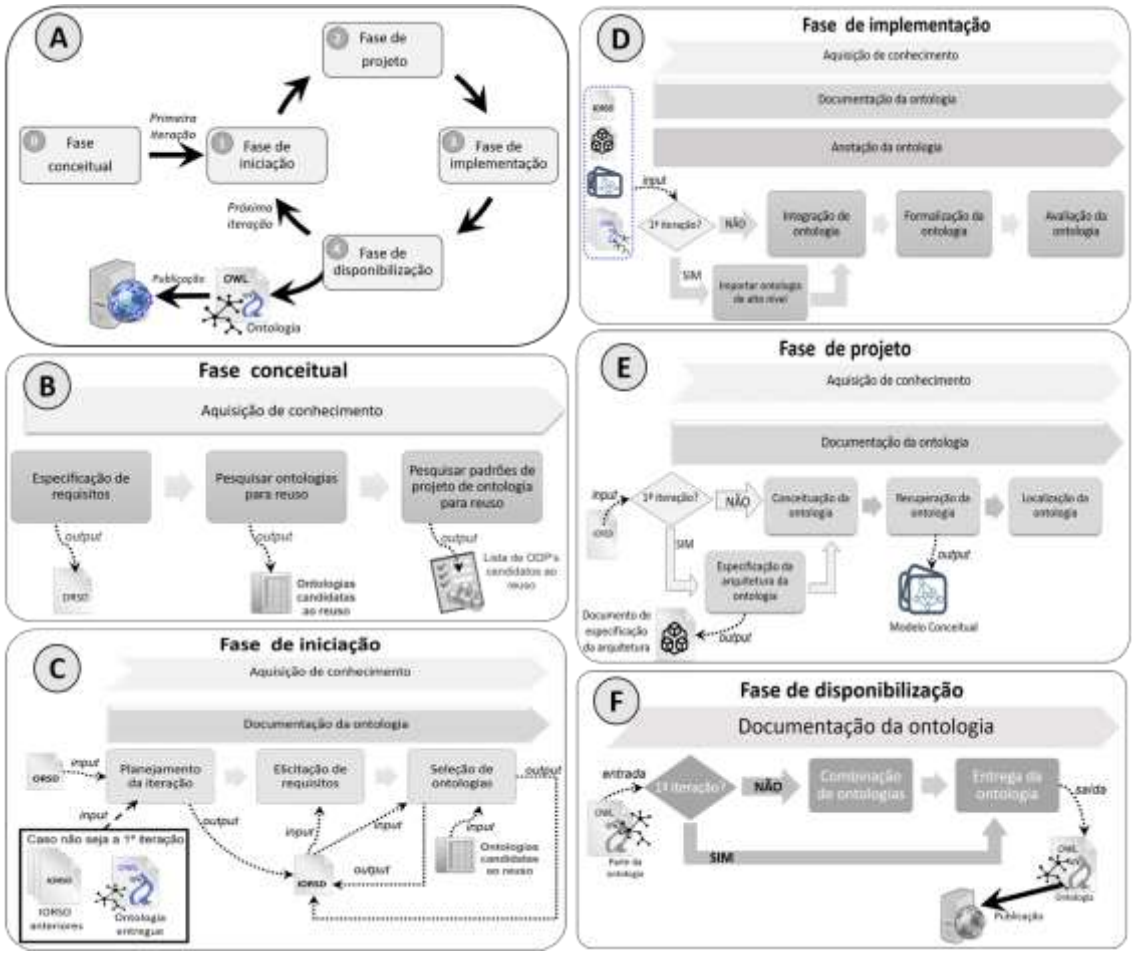


Figura 3.28. Fases do processo de construção de ontologias (A) e suas respectivas atividades (B, C, D, E e F).

Esta metodologia combina as cinco etapas da metodologia do realismo ontológico [Arp *et al.* 2015, Smith, Ceusters 2010] aos cenários e atividades da metodologia NeOn [Suárez-Figueroa 2010]. Sugere-se a adoção de um ciclo de vida de desenvolvimento, iterativo-incremental, como sugerido

pelas duas metodologias de referência. Nas próximas subseções são detalhadas as principais atividades práticas presentes no processo de construção de ontologias biomédicas, esta metodologia foi utilizada para a construção da OntONeo[11] (Ontologia do domínio Obstétrico e Neonatal) que atualmente se encontra publicada no portal OBO Foundry.

3.2.1. OBO Foundry

A OBO Foundry é uma iniciativa que surgiu com o objetivo de construir um conjunto de artefatos ontológicos[12] que representem os termos da área biomédica, minimizando as ambiguidades de termos e definições entre os artefatos de forma a torná-los não redundantes, além de promover a integração entre os artefatos criados [Ghazvinian *et al.* 2011, Smith *et al.* 2007].

O acrônimo OBO originalmente se referia a "Ontologias Biomédicas Abertas" (em inglês *Open Biomedical Ontologies*) e mais tarde tornou-se "Ontologias Biológicas e Biomédicas Abertas" (em inglês *Open Biological and Biomedical Ontologies*). A ideia de fundação refere-se à criação de um conjunto de ontologias biomédicas interoperáveis, e bem formadas e que claramente representem o conhecimento cientifico biomédico, tornando-se uma referência na representação deste conhecimento.

A OBO Foundry fornece um conjunto de princípios[13] considerados como boas práticas na construção de ontologias. Assim, os artefatos ontológicos que são publicados no portal da OBO Foundry devem ser aderentes a tais princípios. Algumas premissas ou princípios são o uso da BFO como ontologia de alto nível e a adoção de um conjunto comum de relações atualmente existentes na ontologia RO (*Relation Ontology*). Para mais detalhes sobre as ontologias BFO e RO disponível em Smith *et al.* (2005) Smith *et al.* (2015) e Arp *et al.* (2015).

Existem 13 princípios definidos na OBO Foundry, como por exemplo: permitir o uso aberto da ontologia, evitar sobreposições de termos e escopo entre ontologias, prover desenvolvimento colaborativo, garantir formato de publicação comum, entre outros que podem ser consultados no portal OBO Foundry[14].

Atualmente, a OBO Foundry possui uma lista de cento e cinquenta e oito ontologias biomédicas, sendo oito ontologias fundacionais (*status Foundry*), cento e nove ontologias possuem a status *Library* que significa que a ontologia foi concebida conforme os princípios da OBO, e 41 ontologias ainda sem especificação de status.

3.2.2. Reuso de ontologias

A prática de reuso é uma estratégia de desenvolvimento que foca no uso de conceitos ou definições, entidades ou classes, relacionamentos, previamente elaborados ou definidos para criação de um novo artefato ontológico. Assim como no desenvolvimento de software, busca reduzir o custo de tempo de construção do artefato ontológico, além de reduzir ambiguidades e duplicação conceituais e semânticas de termos e entidades.

---

[11] Ontologia disponível bilíngue (Português e Inglês) disponível em: https://ontoneo.com/.

[12] Artefatos ontológicos são o mesmo que "*ontologia como um artefato*" conforme quadro 3.1.

[13] Detalhes sobre os princípios em: http://www.obofoundry.org/principles/fp-000-summary.html

[14] Disponível em: http://www.obofoundry.org/

O reuso de artefatos ontológicos minimiza retrabalho no desenvolvimento de um novo projeto, sempre levando em consideração trabalhos anteriores, fazendo com que soluções previamente desenvolvidas sejam aproveitadas e implementadas em novos contextos. No caso das ontologias biomédicas fundamentadas na BFO, o portal OBO Foundry agrega diversas áreas de domínio já cobertas e representadas, servindo assim de uma ótima fonte de artefatos para reuso. Ressalta-se, porém, que as ontologias da OBO Foundry atualmente estão definidas apenas em inglês, exceto a OntONeo que já tem definições em português.

As atividades relacionadas ao reuso de ontologias ou artefatos ontológicos, seja o reuso total ou parcial do artefato, está presente em diversas fases do processo de construção de ontologias (Figura 3.28 - A), divergindo quanto a grau de detalhamento da atividade.

Na fase conceitual do processo (Figura 3.28 - B), deve-se identificar o conjunto de artefatos ontológicos candidatos ao reuso. Para isso, deve-se pesquisar em repositórios de ontologias termos que dizem respeito ao domínio em modelagem.

Existem vários repositórios e mecanismos de busca para artefatos ontológicos biomédicos para apoiar a identificação de artefatos ontológicos candidatos ao reuso, por exemplo:

- Ontobee[15]: Um repositório de artefatos ontológicos que seguem as recomendações da OBO Foundry, que possibilita a pesquisa por artefatos [Xiang *et al.* 2011].
- Bioportal[16]: Um repositório web de ontologias e terminologias biomédicas, desenvolvido e mantido pelo Centro Nacional de Ontologia Biomédica (NCBO), que possibilita a pesquisa por artefatos [Noy *et al.* 2008].
- Ontology Lookup Service (OLS): Um repositório web para ontologias biomédicas, que é desenvolvido e mantido pelo Instituto Europeu de Bioinformática (EMBL-EBI) [Côté *et al.* 2006].

No caso de construção de artefatos ontológicos biomédicos que sejam fundamentados na BFO, sugere-se usar preferencialmente o portal *Ontobee* (Figura 3.29), e apenas no caso de não encontrar nenhum artefato candidato ao reuso utilizar os outros dois para guiar uma nova definição.

Na fase de iniciação (Figura 3.28 - C), deve-se efetivamente realizar a coleta dos artefatos que serão reutilizados. Neste ponto, é importante incorporar ao projeto as diretrizes introduzidas por Courtot *et al.* (2011) conhecidas como MIREOT (do inglês *Minimal Information to Reference External Ontology Terms*). Estas diretrizes recomendam que na reutilização de artefatos ontológicos, deve-se reutilizar estritamente os termos relevantes de outras ontologias ao invés de toda a ontologia. Portanto, de acordo com MIREOT, um ontologista deve importar de outra ontologia apenas elementos necessários (entidades, termos, propriedades e relações) para serem reutilizados, evitando a sobrecarga de importar toda a ontologia.

[15] Disponível em: http://www.ontobee.org/

[16] Disponível em: https://bioportal.bioontology.org/

Figura 3.29. Exemplo de pesquisa por termos no portal Ontobee.

As ontologias como um artefato são formalizadas usando uma linguagem formal, em geral, utilizando a Ontology Web Language (OWL). Para detalhes sobre a linguagem OWL consulte Antoniou e Harmelen (2009), W3c (2004a) e W3c (2004b). No caso das ontologias da OBO Foundry, elas devem ser expressas por OWL.

Para reutilizar elementos de artefatos ontológicos já existentes é necessário recortar os elementos para reuso no arquivo OWL da ontologia preexistente. Para garantir a integridade dos elementos que serão reutilizados, sugere-se o uso da ferramenta Ontofox [Xiang *et al.* 2010].

A OntoFox é uma ferramenta baseada na Web que implementa as diretrizes MIREOT extraindo das ontologias apenas as informações mínimas necessárias de acordo com as entradas dos usuários e gerando um arquivo de saída no formato OWL (Figura 3.30).

O arquivo OWL gerado pela Ontofox, contendo o subconjunto de elementos do artefato ontológico em reuso, deverá ser importado no artefato ontológico em desenvolvimento.

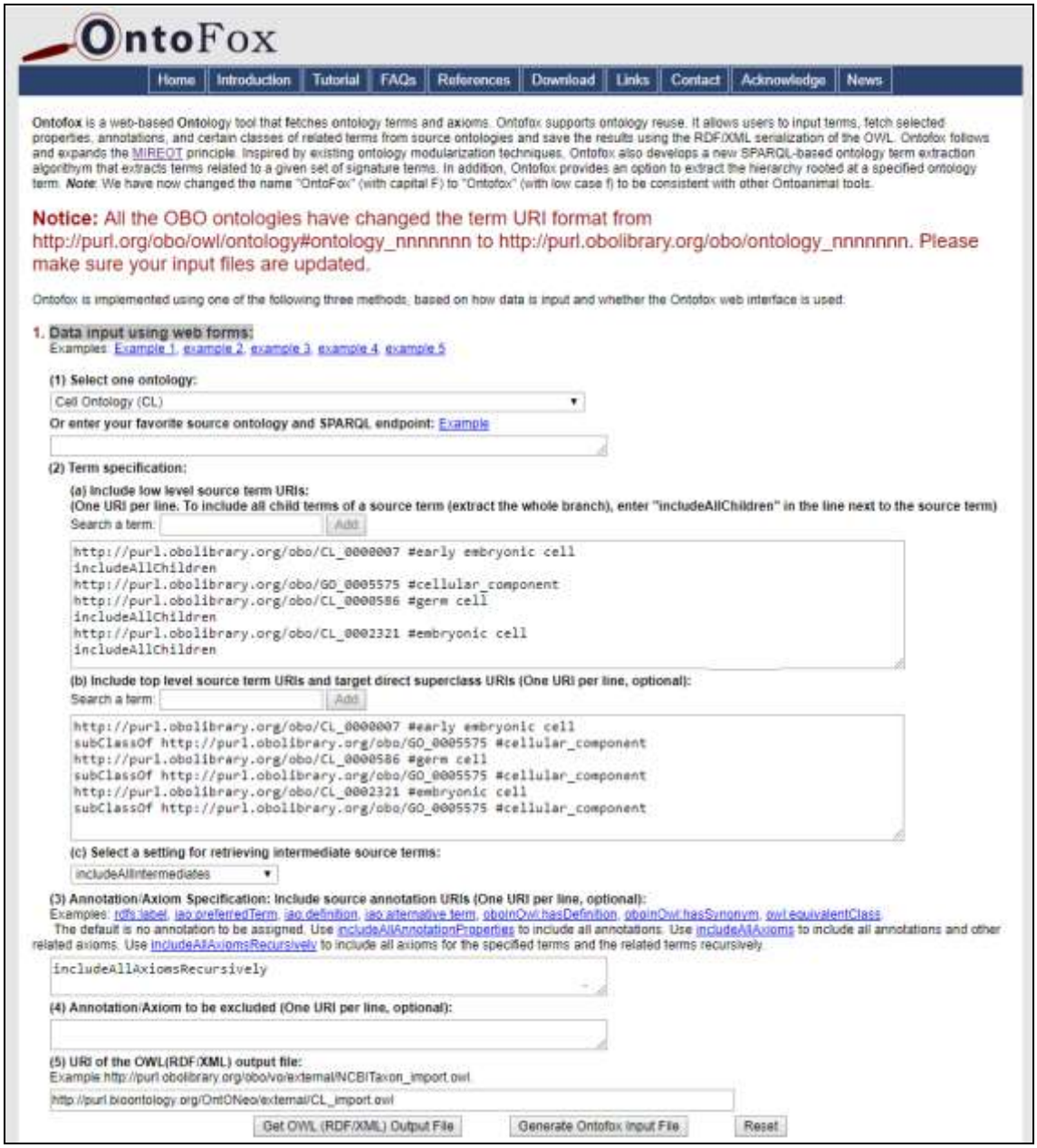


Figura 3.30. Exemplo de extração de elementos para reuso na ontologia *Cell Ontology*.

3.2.3. Definição de *namespace* e URIs

Na fase de projeto do processo de construção de artefatos ontológicos (Figura 3.28 - E), uma atividade chave é determinar elementos arquiteturais da ontologia que sejam aderentes aos princípios da OBO Foundry. Nesta fase são definidos:

- Prefixo do artefato ontológico.
- Identificador local dos elementos do artefato ontológico.
- *Namespace* do artefato ontológico e URI padrão do artefato e de seus elementos.

O prefixo é um acrônimo que será associado ao artefato ontológico quando for necessário fazer menção tanto ao artefato quanto aos elementos do artefato. Por exemplo, o prefixo é comumente utilizado na composição do *namespace* e do identificador dos elementos do artefato.

Para determinar o prefixo da Ontologia do domínio Obstétrico e Neonatal, foi definido um prefixo candidato, ONTONEO, e realizada uma pesquisa na lista de ontologias da OBO Foundry para garantir que o prefixo candidato não era usado por nenhum artefato já existente. Em seguida, utilizou-se o serviço *prefix.cc*, para garantir que nenhum outro recurso usasse o prefixo em questão. *Prefix.cc* (Figura 3.31) é um serviço criado para desenvolvedores de RDF no qual os ajuda na busca por prefixos de URI e *namespaces*.

Figura 3.31. Tela de pesquisa do serviço prefix.cc.

A OBO Foundry recomenda que o identificador local dos elementos do artefato ontológico seja uma concatenação do prefixo (também identificado como ID-Space), um caractere "_" (sublinhado ou *underscore*) e uma sequência de números (Figura 3.32).

OBO Foundry local identifier: <IDSPACE>_<NUMBER>
Example: ONTONEO_00000001

Figura 3.32. Exemplo do padrão de nomes OBO Foundry.

Um *namespace* é um mecanismo para fornecer contexto genérico para encapsular itens. O *namespace* ou espaço de nomes é um sistema de nomeação de objetos que delimita o espaço abstrato para os itens que ele armazena, permite a desambiguação para itens ou objetos que possuem o mesmo nome mas que residem em espaços de nomes diferentes [Antoniou, Harmelen 2009]. Assim, cada ontologia tem seu próprio *namespace,* que é uma sequência de caracteres que precede os identificadores de seus elementos. Um *namespace* padrão é um URI (*Uniform Resource Identifiers*) ou IRI (*Internationalized Resource Identifiers*) válido seguido por um caractere "/" ou "#". Para isso, definiu um URL persistente (purl) usando o Serviço PURL (*PURL Service*[17]). No caso da OntONeo, seguindo as recomendações da OBO Foundry, para

[17] Disponível em: https://archive.org/services/purl/.

utilizar o domínio OBO PURL (*http://purl.obolibrary.org/obo/*), a URL definida foi *http://purl.obolibrary.org/obo/ontoneo*. O serviço *prefix.cc* também pode ser usado para verificar a existência ou não da URL ou *namespace* em questão.

3.2.4. Utilização da ferramenta Protégé

O Protégé é um ambiente de desenvolvimento integrado (IDE, do inglês *Integrated Development Environment*) gratuito e de código aberto para a construção de artefatos ontológicos e funcionalidades afins [Almeida 2006, Noy *et al.* 2003].

<u>Configurações iniciais do Protégé:</u>

Na fase conceitual do processo (Figura 3.28 - D), para iniciar a implementação da ontologia é necessário configurar o Protégé para evitar retrabalho durante o processo de formalização.

As configurações básicas que devem ser realizadas são: (i) URI/IRI base para os elementos do artefato (Figura 3.33 - A); (ii) Anotações básicas do artefato: Nome do criador e data de criação (Figura 3.33 – B e C); (iii) Geração automática do identificador local para os elementos do artefato (Figura 3.34); (iv) URI/IRI do padrão do artefato e de versionamento do artefato (Figura 3.35).

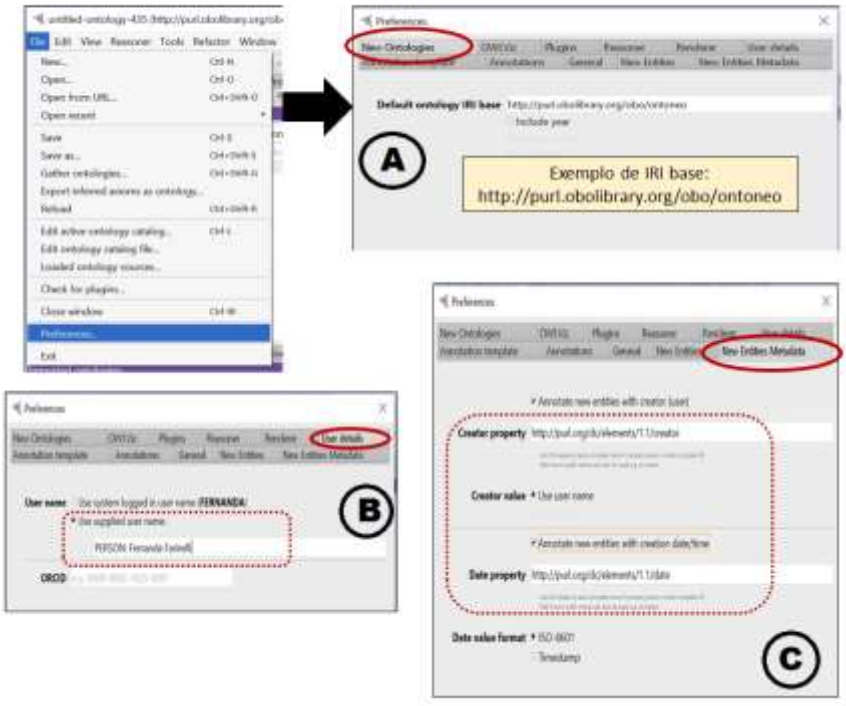


Figura 3.33. Tela de preferencias do Protégé para configurações básicas.

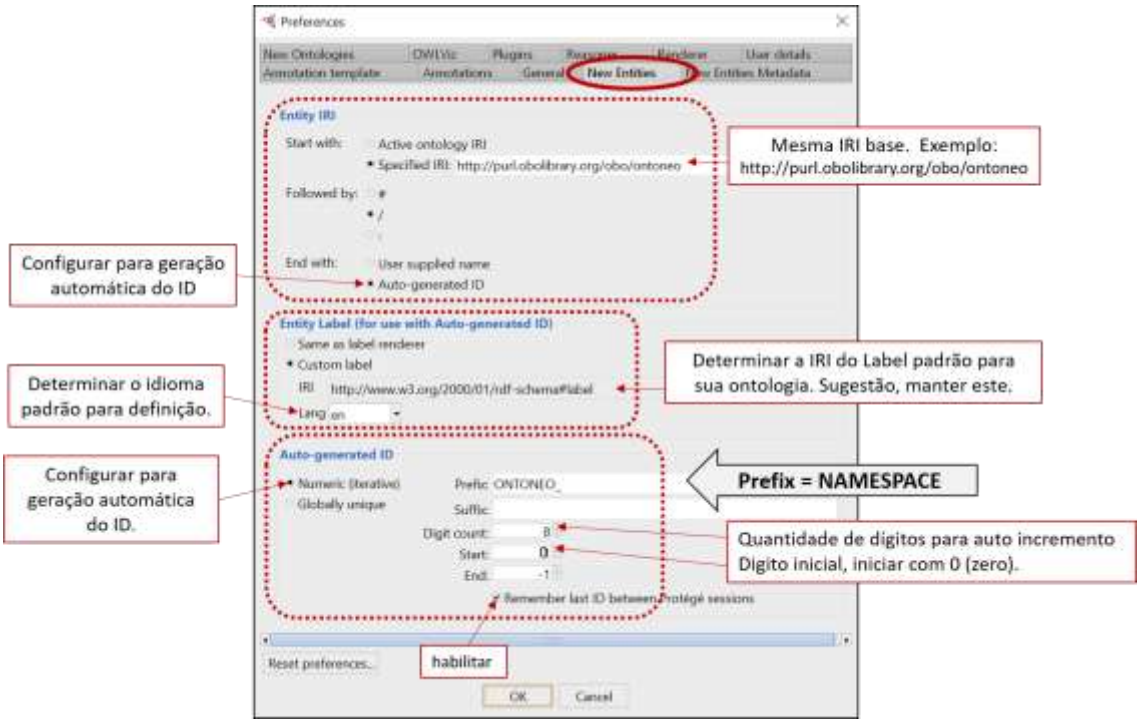


Figura 3.34. Tela de preferencias do Protégé para configurações básicas.

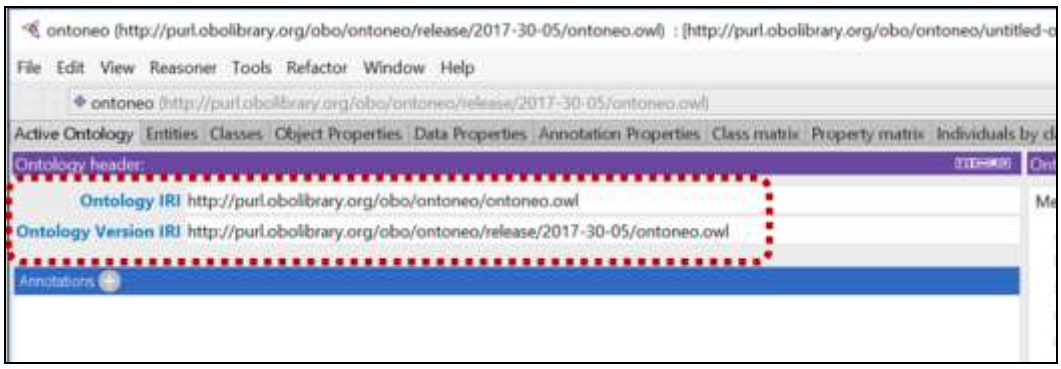


Figura 3.35. Tela inicial do Protégé para configurações da IRI e IRI de versionamento.

Importação de artefatos ontológicos em reuso

Ainda na fase de implementação do processo de construção do artefato ontológico (Figura 3.28 - D), está prevista a importação de outros artefatos ontológicos, seja tal artefato a ontologia de alto nível que vai fundamentar o artefato em construções ou seja os artefatos ontológicos que serão reutilizados.

No caso de o projeto seguir os princípios da OBO Foundry, primeiramente deve-se importar a ontologia de alto nível BFO[18], seguido pela importação da ontologia de relacionamentos

[18] Arquivo OWL disponível para download em: http://www.obofoundry.org/ontology/bfo.html

a RO[19]. Após a importação dos artefatos ontológicos base, deve-se realizar a importação dos artefatos ontológicos que serão reutilizados. Estes artefatos são os arquivos OWL gerados pelo Ontofox devem ser importados para seu projeto.

A Figura 3.36 abaixo ilustra o passo a passo para importação de artefatos ontológicos que se encontram disponíveis em arquivo OWL. Lembre-se de organizar as entidades importadas das ontologias em reuso conforme a hierarquia de entidades da ontologia de alto nível em uso.

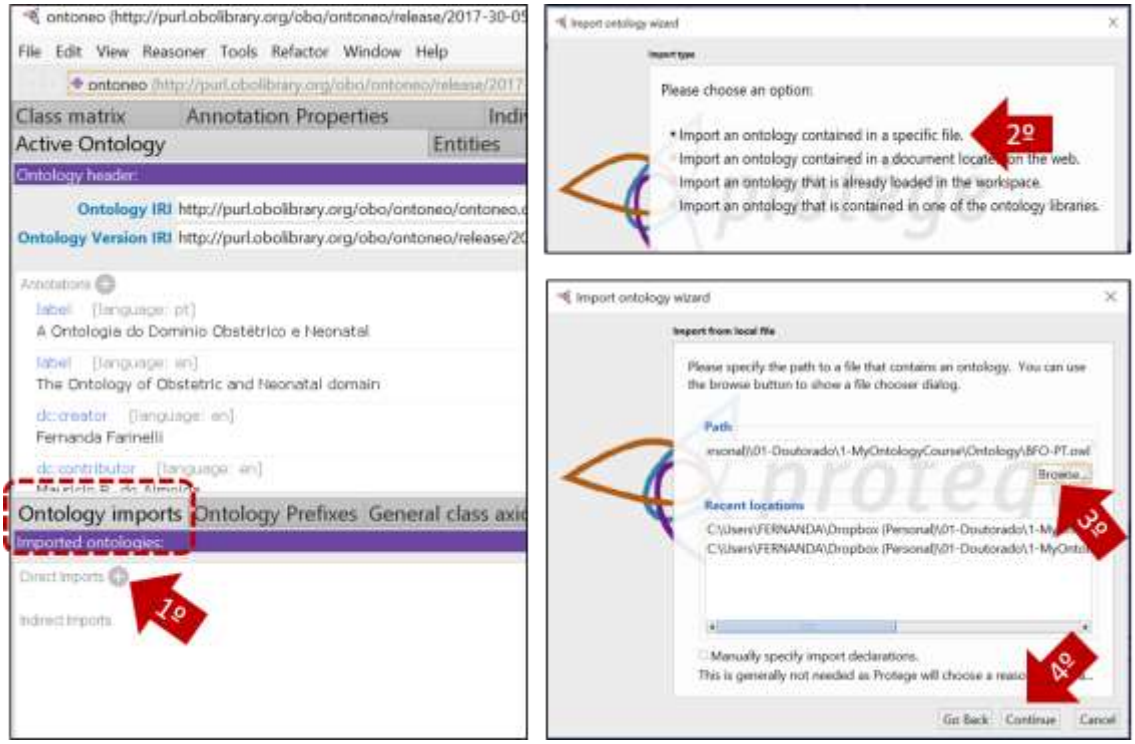


Figura 3.36. Tela inicial do Protégé para importação de artefatos ontológicos para reuso no projeto.

Documentação do artefato ontológico

A atividade de documentação de ontologia é uma atividade de suporte que gera qualquer documentação valiosa para entender o próprio artefato ontológico e as decisões tomadas durante seu desenvolvimento (Figura 3.28 - D).

A documentação do artefato pode ser realizada no próprio artefato por meio das anotações disponíveis. As anotações (*annotation*) podem ser vistas como metadados do artefato. Não existe um padrão de documentação, no geral, cada desenvolvedor de ontologia determina quais são as anotações que são relevantes para seu próprio projeto. Considerando a experiência no desenvolvimento da OntONeo, recomenda-se a adoção das anotações listadas na tabela 3.1 para enriquecer as informações no artefato em desenvolvimento. Esta lista foi obtida pela compilação das diversas anotações utilizadas nos projetos disponíveis na OBO Foundry.

[19] Arquivo OWL disponível para download em: http://www.obofoundry.org/ontology/ro.html

Tabela 3.1. Lista de anotações recomendadas para documentar o artefato ontológico.

Origem da anotação		Nome da anotação	Escopo de uso
PREFIX	***namespace***		
dc	http://purl.org/dc/elements/1.1/	creator	Ontologia
		contributor	Ontologia
		license	Ontologia
		date	Qualquer elemento
		source	Importados
foaf	http://xmlns.com/foaf/0.1/	homepage	Ontologia
		mbox	Ontologia
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#	Description	Qualquer elemento
rdfs	http://www.w3.org/2000/01/rdf-schema#	label	Ontologia
		range	Propriedades
		domain	Propriedades
		comment	Qualquer elemento
		isDefinedBy	Qualquer elemento
owl	http://www.w3.org/2002/07/owl#	versionIRI	Ontologia
		versionInfo	Ontologia
		priorVersion	Ontologia
obo	http://purl.obolibrary.org/obo/	IAO_0000111 editor preferred term	Entidade
		IAO_0000114 has curation status	Entidade e Propriedades
		IAO_0000115 definition	Entidade
		IAO_0000117 term editor	Entidade
		IAO_0000118 alternative term	Entidade
		IAO_0000119 definition source	Entidade
		IAO_0000412 imported from	Importados
oboInOwl	http://www.geneOntologia.org/formats/oboInOwl#	id	Entidade
		created_by	Qualquer elemento
		creation_date	Qualquer elemento
		hasDbXref	Entidade
owl	http://www.w3.org/2002/07/owl#	versionInfo	Ontologia
		priorVersion	Ontologia
protege	http://protege.stanford.edu/plugins/owl/protege#	defaultLanguage	Ontologia

3.2.5. Definição do artefato ontológico

O processo de construção de artefatos ontológicos, como ilustrado na figura 3.28 – A envolve uma fase exclusiva, a fase conceitual (Figura 3.28 – B), para determinar o propósito da ontologia, o

escopo ou domínio de cobertura do artefato a ser construído, e o conjunto de requisitos que o artefato deve satisfazer.

Como uma das primeiras atividades no processo de construção de ontologias, é importante realizar a identificação dos requisitos que a ontologia de cobrir. Para isso, sugere-se seguir as orientações apresentadas em [Suárez-Figueroa *et al.* 2009].

O processo de construção de ontologias envolve muitas outras atividades que descritas em Farinelli e Elkin (2017) e Farinelli (2017).

References

Almeida, M. B. D. (2006) "Noções Básicas para uso do Protégé", online, Tutorial, 09/07/2006. <*http://mba.eci.ufmg.br/onto_frames/*>, 16/04/2019.

Almeida, M.B. (2013). Revisiting Ontologies: a necessary clarification. Journal of the American Society for Information Science and Technology. vol. 64, n. 8, p. 1682–1693.

Abrial, J. R. (1974). "Data Semantics". In J. W. Klimbie & K. L. Koffeman, (Eds.), Proceedings of the IFIP Working Conference Data Base Management (pp. 1-60). Amsterdam: North-Holland.

Ackrill, J. L. (1963). Aristotle: Categories and De Interpretatione. Oxford: Clarendon Press.

Alturki, A.; Gable, G.; Bandara, W. (2013). "BWW ontology as a lens on IS design theory: extending the design science research roadmap". Proceedings of the 8th International Conference on Design Science at the Intersection of Physical and Virtual Design. p. 258-277.

Antoniou, G.Harmelen, F. V. (2009). "Web Ontology Language: OWL". In: Staab, S.Studer, R. (Ed.). Handbook on Ontologies. Berlin, Heidelberg: Springer Berlin Heidelberg. p.91-110.

Arp, R., Smith, B., Spear, A. D. (2015). Building Ontologies with Basic Formal Ontology. Cambridge, Massachusetts: The Mit Press, 2015.

Baader, F., Bürckert, H., Hollunder,H, Laux,A., Nutt, W (1992). "Terminologische Logiken". KI vol. 6, n. 3. p. 23-3.

Baader, F., Horrocks, I., Sattler, U. (2002). "Description Logics for the Semantic Web". KI vol. 16, n. 4. p. 57-59.

Berners-Lee, T.B., Hendler, J., Lassila, O. (2001). "The Semantic Web". Retrieved July 20, 2008, from https://www-sop.inria.fr/acacia/cours/essi2006/Scientific%20American_%20Feature%20Article_%20The%20Semantic%20Web_%20May%202001.pdf.

Bernus, P., Nemes, L., & Williams, T. J. (1996). "Architectures for enterprise integration". London: Chapman & Hall.

Booch, G. (1993). Object-oriented Analysis and Design with Applications, 2nd ed., Redwood: Benjamin Cummings.

Bosak, R., Richard, F. Clippinger, R. F., Dobbs, C., Goldfinger, R., Jasper, R. B., Keating, W., Kendrick, G., Sammet, J. E. (1962). "An information algebra: phase 1 report - language structure group of the CODASYL development committee". Communications of the ACM, vol. 5, n. 4, p.190-204.

Brochaussen, M., Almeida, M.B., Slaughter, L. (2013). Towards a formal representation of document acts and the resulting legal entities. In: Ingthorsson, R.D., Svennerlind, C., and Almäng J. (Ed.). Johanssonian Investigations. Ontos: Frankfurt, p. 120-139.

Chen, P. (1976). "The entity-relationship model: towards a unified view of data". ACM Transactions on Database Systems, vol. 5 , n. 1. p. 9-36.

Cohen, S. M. (2008). Aristotle's Metaphysics. Retrieved February 10, 2012, from http://plato.stanford.edu/entries/aristotle-metaphysics/

Cood, E. F. (1979). "Extending the database relational model to capture more meaning". ACM Transactions on Database Systems, vol. 4, n. 4, p. 397-434.

Côté, R. G. *et al.* (2006) "The Ontology Lookup Service, a lightweight cross-platform tool for controlled vocabulary queries". BMC bioinformatics**,** v. 7, n. 1, p. 97.

Courtot, M. *et al.* (2011) "MIREOT: The minimum information to reference an external ontology term". Applied ontology**,** v. 6, n. 1, p. 23-33.

De Nicola, A.Missikoff, M.Navigli, R. (2005). "A proposal for a unified process for ontology building: UPON". In: K.V., A.J., D.R., W., Database and Expert Systems Applications. DEXA 2005, Springer, Berlin, Heidelberg. p.655-664.

Falbo, R. D. A. (2014). "SABiO: Systematic Approach for Building Ontologies". In: Guizzardi, G. *et al*, ONTO-COM-ODISE 2014- Ontologies in Conceptual Modeling and Information Systems Engineering, Rio de Janeiro, Brazil. CEUR Workshop Proceedings.

Farinelli, F. (2017). "Improving semantic interoperability in the obstetric and neonatal domain through an approach based on ontological realism". 256 p. Doctoral (Doctor in Information Science). School of Information Science Federal University of Minas Gerais at Brazil, Belo Horizonte. *Publicado em* English.

Farinelli, F., Elkin, P. L. (2017). "Construção de ontologia na prática: um estudo de caso aplicado ao domínio obstétrico". Ciência da Informação**,** v. 46, n. 1.

Fernández-López, M.Gómez-Pérez, A.Juristo, N. (1997). "Methontology: from ontological art towards ontological engineering". Proceedings of the Ontological Engineering AAAI-97 Spring Symposium Series, Stanford University, EEUU. American Asociation for Artificial Intelligence.

Fillion, E., Menzel, C., Blinn, T., Mayer, R. (1995). "An ontology-based environment for enterprise model integration". Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing at IJCAI95. Montreal: AAAI Press (pp. 33-45).

Frigg, R. (2006). Models in Science. Retrieved July 18, 2008, from http://plato.stanford.edu/entries/models-science/

Fonseca, F. (2007). "The Double Role of Ontologies in Information Science Research". Journal of the American Society for Information Science and Technology. vol. 58, n. 6, p. 786–793.

Fox, M. S. (1992). "The TOVE Project: towards a common-sense model of the enterprise". In F. Belli, & F. J. Radermacher (Eds.), Proceedings of 5th International Conference Industrial and Engineering Applications of Artificial Intelligence and Expert Systems. pp. 25-34. London: Springer-Verlag.

Gandon, F. (2002). Distributed artificial intelligence and knowledge mangement: ontologies and multi-agent systems for a corporate semantic web. PhD Thesis, INRIA and University of Nice, Nice, FR, School of Sciences and Technologies of Information and Communication.

Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Oltramari, R., & Schneider, L. (2002). Sweetening Ontologies with DOLCE. Retrieved January 10, 2010, from http://www.loa.istc.cnr.it/Papers/DOLCE-EKAW.pdf

Genesereth, M. R., Nilsson, L. (1987). Logical foundation of AI. San Francisco: Morgan Kaufman.

Ghazvinian, A., Noy, N. F., Musen, M. A. (2011) "How orthogonal are the OBO Foundry ontologies?". Journal of Biomedical Semantics**,** vol. 2, n. 2, p. S2.

Grenon, P., Smith, B., Goldberg, J. (2007). Biodynamic Ontology: Applying BFO in the Biomedical Domain. Retrieved September 20, 2011, from http://ontology.buffalo.edu/medo/biodynamic.pdf

Guarino, N. (1995). Formal ontology, conceptual analysis and knowledge representation. International Journal of Human-Computer Studies, vol. 43, n.5-6, p. 625-640.

Guarino, N. (1998). Formal ontology and information systems. In N. Guarino (Ed.), Formal ontology in information systems. pp. 3–15. Amsterdam: IOS Press.

Guarino, N., Giaretta, P. (1995). “Ontologies and KBs: towards a terminological clarification”. In N. Mars (Ed.), Towards a Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing. pp. 25-32. Amsterdam: IOS Press.

Guizzardi, G. (2005). Ontological Foundations for Structural Conceptual Models. PhD Thesis, University of Twente, Twente, NL, Centre for Telematics and Information Technology.

Gruber, T. (1993). What is an ontology? Retrieved September 14, 2002, from http://www-ksl.stanford.edu/kst/what- is-an-ontology.html

Grüninger, M., Fox, M. S. (1995). “Methodology for the Design and Evaluation of Ontologies”. In: Mellish, C. S., 14th International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, CA. Morgan Kaufmann.

Hoehndorf, R., Schofield, P.N., Georgios, G. V. (2015). “The role of ontologies in biological and biomedical research: a functional perspective”. Briefings in bioinformatics.vol. 16, n. 6, p. 1069-1080.

Horridge, M. (2011). A” Practical Guide To Building OWL Ontologies Using Protegé”. Retrieved September 20, 2014, from http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_3.pdf

Jacobson, I., Christerson, M., Jonsson, P., Overgaard, G. (1992). Object-Oriented Software Engineering: A Use Case Driven Approach. Massachusetts: Addison-Wesley.

Jardine, D. A. (1976). “The ANSI/SPARC DBMS model”. Proceedings of the 2nd SHARE Working Conference on Database Management Systems. Amsterdam: North Holland.

Lennox, J. (2000). Aristotle's Philosophy of Biology: Studies in the Origins of Life Science. Cambridge: Cambridge University Press.

Lowe, E. J. (2007). The Four-Category Ontology: A Metaphysical Foundation for Natural Science. New York: Oxford University Press.

MacLeod, M.C.and Rubenstein, E.M.(2005). Universals. Retrieved March 30, 2010, from http://www.iep.utm.edu/universa/

Milton, S. (2000). An Ontological Comparison and Evaluation of Data Modelling Frameworks. PhD Thesis, The University of Tasmania, Hobart, AU, School of Information Systems.

Miñarro-Giménez, J.A., Kreuzthaler, M., Schulz, S. (2015). "Knowledge Extraction from MEDLINE by Combining Clustering with Natural Language Processing". Proceedings of the AMIA Annual Symp.

Mylopoulos, J. (1992). "Conceptual Modelling and Telos". In P. Loucopoulos, & R. Zicari (Eds.), Conceptual Modelling, Databases and CASE: An Integrated View of Information Systems Development. New York: Wiley.

Mylopoulos, J. (1998). "Information Modeling in the time of revolution". Informations Systems, vol. 23, n. 3, p. 127-155.

Niiniluoto, I. (1999). Critical Scientific Realism. New York: Oxford University Press.

Noy, N. F. *et al.* (2003) "Protege-2000: an open-source ontology-development and knowledge-acquisition environment". AMIA Symposium. p.953.

Noy, N. F. *et al.* (2008) "BioPortal: A Web Repository for Biomedical Ontologies and Data Resources". In: Joshi, C. B. A., 7th International Semantic Web Conference (ISWC2008), Karlsruhe, Germany. CEUR-WS.org.

Ogden, C. K., Richards, I.A (1923). The Meaning of Meaning.London: Roultledge.

Peckham, J., Maryanski, F. (1988). "Semantic Data Models". ACM Computing Surveys, vol. 20, n. 3, p. 153-189.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W. (1991). Object-Oriented Modeling and Design, New York: Prentice Hall.

Schulz, S., Daumke, P., Stenzhorn, H., Poprat, M. (2009). "Incremental Semantic Enrichment of Narrative Content in Electronic Health Records". Proceedings of the World Congress on Medical Physics and Biomedical Engineering.

Schulz, S. *et al.* (2012). "Guideline on Developing Good Ontologies in the Biomedical Domain with Description Logics". December 11th, 2012, p.85 <*https://www.iph.uni-rostock.de/en/forschung/homepage-goodod/guideline/*>.

Schulz, S. (2016). "Knowledge acquisition and management for clinical decision-making". Grand Rounds September 9th, 2016. Dr. Stefan Schulz, stefan.schulz@medunigraz.at, Medical University of Graz.

Smith, B., Welty, C. (2001). "Ontology: Towards a New Synthesis". In: B. Smith & C. Welty (Eds.), Proceedings of the International Conference on Formal Ontology in Information Systems (pp. 3-9). Ogunquit: ACM Press.

Smith, B. (1998). "The Basic Tools of Formal Ontology". In: N. Guarino (Ed.), Proceedings of Formal Ontology in Information Systems (pp. 3-15). Amsterdam: IOS Press.

Smith, B. (2003). Ontology and Information Systems. Retrieved March 20, 2005, from http://www.ontology.buffalo.edu/ontology(PIC).pdf

Smith et al. (2007). The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. Retrieved May 22, 2009, from http://www.nature.com/nbt/journal/v25/n11/full/nbt1346.html.

Smith, B. (1997). "On Substances, Accidents and Universals: In Defence of a Constituent Ontology". Philosophical Papers, vol. 26, p. 105–127.

Smith, B. (2003). Ontology and Information Systems. Retrieved March 20, 2005, from http://www.ontology.buffalo.edu/ontology(PIC).pdf

Smith, B. *et al.* (2015) "Basic Formal Ontology 2.0: Specification and User's Guide"" June 26th, 2015, p.97 *<https://github.com/BFO-ontology/BFO/raw/master/docs/bfo2-reference/BFO2-Reference.pdf>*.

Smith, B. *et al.* (2007). "The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration". Nature Biotechnology**,** v. 25, n. 11, p. 1251-1255.

Smith, B., Ceusters, W. (2010) "Ontological realism: A methodology for coordinated evolution of scientific ontologies". Applied ontology**,** v. 5, n. 3-4, p. 139-188, November 15th, 2010.

Smith, B. *et al.* (2005). "Relations in biomedical ontologies". Genome Biology**,** vol. 6, n. 5.

Spear, A. D. (2006). Ontology for the Twenty First Century: An Introduction with Recommendations. Retrieved March 20, 2010, from https://www.researchgate.net/publication/238687379_Ontology_for_the_Twenty_First_Century_An_Introduction_with_Recommendations

Staab, S., Studer, R. (2004). Handbook on Ontologies. Berlin: Springer.

Studtmann, P. (2008). The foundations of Aristotle's categorial scheme. Milwaukee: Marquette University Press.

Suárez-Figueroa, M.Gómez-Pérez, A.Villazón-Terrazas, B. How to Write and Use the Ontology Requirements Specification Document. In: Meersman, R.Dillon, T.Herrero, P. (Ed.). On the Move to Meaningful Internet Systems: OTM 2009: Springer Berlin Heidelberg, v.5871, 2009. cap. 16, p.966-982. (Lecture Notes in Computer Science).

Suárez-Figueroa, M. C. (2010) "NeOn Methodology for building ontology networks: specification, scheduling and reuse". 268 p. (Doctoral thesis). Inteligência Artificial, Universidad Politécnica de Madrid, Madri. *Publicado em* English.

Sutcliffe, J. P. (1993). "Concept, class, and category in the tradition of Aristotle". In: I. van Mechelen, J. Hampton J, R.S. Michalski,&P. Theuns P (Eds). Categories and Concepts. pp. 35-65. London: Academic Press.

Thomason, A. (2009).Categories. Retrieved March 3, 2011, from http://plato.stanford.edu/entries/categories/

Uschold, M., King, M., Moralee, S., Zorgios, Y. (1998). "Enterprise ontology". The Knowledge Engineering Review, vol. 13, p. 1-69.

Vickery, B. C. (1997). "Ontologies". Journal of Information Science. 23 (4), 227-286.

Wand,Y., Weber, R. (1990). "Mario Bunge's ontology as a formal foundation for information systems concepts". In: P. Weingartner & J.W.G. Dorn (Eds.), Studies on Mario Bunge's treatise. Amsterdam: Rodopi.

Wand, Y., Storey, V. C., Weber, R. (1999). "An ontological analysis of the relationship construct in conceptual modeling". ACM Transactions on Database Systems, vol. 24, n. 4, p. 494-528.

W3c. (2004a). "OWL Web Ontology Language Guide". Smith, M. K.Welty, C.Mcguinness, D. L.: World Wide Web Consortium OWL Working Group. 2004 <*https://www.w3.org/TR/2004/REC-owl-guide-20040210/*>,

_____. (2004b) "OWL Web Ontology Language Reference". Bechhofer, S. *et al*: World Wide Web Consortium OWL Working Group. 2017 <*http://www.w3.org/TR/2004/REC-owl-ref-20040210/*>,

Xiang, Z. *et al*. (2010). "OntoFox: web-based support for ontology reuse". BMC research notes**,** vol. 3, n. 1, p. 175.

Xiang, Z. *et al*. (2011). "Ontobee: A Linked Data Server and Browser for Ontology Terms, Ontology". 2nd International Conference on Biomedical Ontology (ICBO2011), Buffalo, NY, USA. CEUR-WS.org, 2011. p.279-281.

Young, J. W., Kent, H. K. (1958). "Abstract formulation of data processing problems". The Journal of Industrial Engineering, vol. 9, n. 6, p. 471-479.

Capítulo 4

O estado da arte em pesquisa observacional de dados de saúde: A iniciativa OHDSI

Maria Tereza Fernandes Abrahão (USP), Moacyr Roberto Cuce Nobre (USP), Pablo Jorge Madril

Abstract

The Observational Medical Outcomes Partnership (OMOP) initiative, together with its follow-up, Observational Health Data Sciences and Informatics (OHDSI), redefined the area of observational research in health data, bringing the possibility of systematic analysis in large masses of data (big data), from a variety of sources, through the definition of a common data model, mechanisms to treat different vocabularies, and the availability of a set of free software tools to analyze them. These tools are expected to leverage evidence-gathering in the medical field and evaluation of therapies and procedures in the real world to support clinical research in general. The chapter gives an overview of the OHDSI platform and tools.

Resumo

A iniciativa da Observational Medical Outcomes Partnership (OMOP) em conjunto com a sua sequência, a Observational Health Data Sciences and Informatics (OHDSI), redefiniram a área de pesquisa observacional em dados de saúde trazendo a possibilidade de realizar análises sistemáticas em grandes massas de dados (big data), provindas de diversas fontes, através da definição de um modelo comum de dados, de mecanismos para tratamento de diferentes vocabulários, e da disponibilidade de um conjunto de ferramentas de software livre para análise dos mesmos. Com estas ferramentas espera-se alavancar o levantamento de evidências na área médica e na avaliação de terapias e procedimentos no mundo real, para apoiar a pesquisa clínica em geral. O capítulo apresenta uma visão geral da plataforma e ferramentas OHDSI.

4.1. Introdução

Com a disponibilidade das bases de dados de saúde em formato eletrônico, surgiu a possibilidade de gerar evidências e discernimentos sistemáticos em larga escala sobre a aplicação dos cuidados de saúde aos pacientes.

Os dados de assistência médica podem variar muito de uma organização para outra e são coletados para diferentes propósitos, como reembolso de provedor, pesquisa clínica e atendimento direto ao paciente. Esses dados podem ser armazenados em diferentes sistemas de banco de dados, formatos e modelos de informação e, apesar do uso crescente de terminologias padrão na área da saúde, o mesmo conceito pode ser representado de várias maneiras, de um ambiente para outro. A padronização de dados é o processo de trazer dados para um formato comum que permita a pesquisa colaborativa, análises em grande escala e compartilhamento de ferramentas e metodologias.

No entanto, a pesquisa na área de saúde requer informações de qualidade que estejam disponíveis no formato adequado e com a devida estruturação, para que efetivamente auxiliem os pesquisadores com informações relevantes acerca do universo de sua pesquisa. É necessário uma definição correta e clara sobre o acesso aos dados, as etapas de busca, limpeza, análise e uso das informações, bem como, a recuperação sistemática dos dados, informações e conhecimentos relevantes que se encontram distribuídos de modo disperso. O uso secundário do dado assistencial se constitui como uma fonte de informação importante para pesquisa de desfechos, porém apresenta uma série de desafios metodológicos peculiares a este tipo de fonte de dados.

Os estudos observacionais e análises secundárias de conjuntos de dados existentes na área da saúde, podem ser de grande valor na geração de evidências e eficácia em contextos do mundo real. Embora os estudos observacionais não possam fornecer evidências definitivas de segurança ou eficácia, eles podem: fornecer informações sobre o uso e a prática do "mundo real"; detectar sinais sobre os benefícios e riscos do uso de terapias na população em geral; ajudar a formular hipóteses a serem testadas em experimentos subsequentes; fornecer parte dos dados a nível populacional necessários para os ensaios clínicos; e informar a prática clínica.

A iniciativa da *Observational Medical Outcomes Partnership* (OMOP) em conjunto com a sua sequência, a *Observational Health Data Sciences and Informatics* (OHDSI)[1], redefiniram a área de pesquisa observacional em dados de saúde trazendo a possibilidade de realizar análises sistemáticas em grandes massas de dados (*big data*). Através da definição de um modelo comum de dados (*Common Data Model* - CDM-OMOP)[2], de mecanismos para tratamento de diferentes vocabulários, e da disponibilidade de um conjunto de ferramentas de software livre para análise dos dados,

[1] OMOP/OHDSI https://ohdsi.org/
[2] CDM-OMOP https://www.ohdsi.org/data-standardization/the-common-data-model/

espera-se alavancar o levantamento de evidências na área médica e na avaliação de terapias e procedimentos no mundo real, para apoiar a pesquisa clínica em geral.

O objetivo deste capítulo é apresentar os fundamentos da pesquisa observacional e os tipos de estudos disponíveis em bases de dados de saúde; o modelo comum de dados (CDM); e as ferramentas de geração e análise do OHDSI de forma prática com o uso de exemplos de estudos reais. Espera-se que, ao final do capítulo, o leitor tenha uma visão geral da plataforma e ferramentas OHDSI e seja capaz de: i) compreender o modelo comum de dados (CDM-OMOP), ii) ter uma visão dos vocabulários e da definição e utilização dos conceitos, e iii) entender o processo da definição de uma coorte e a visualização dos dados assistenciais, fomentando assim uma pesquisa observacional de alto nível.

O capítulo está estruturado como se segue. Primeiro, a seção 4.1 oferece uma breve fundamentação sobre o uso dos sistemas de registros eletrônicos de saúde como fontes de dados, os desafios que envolvem a reprodutibilidade da pesquisa científica e a iniciativa OMOP/OHDSI. Em seguida, a seção 4.2 apresenta a arquitetura do sistema OHDSI, a seção 4.3 o modelo comum de dados (CDM - OMOP) e a seção 4.4 os vocabulários. A seção 4.5 apresenta a definição de uma coorte e a seção 4.6 os tipos de análises. As considerações finais e conclusões são apresentadas na seção 4.7 e a seção 4.8 apresenta um glossário de termos e na sequência, as referências consultadas.

4.1.1. Sistemas de registros eletrônicos de saúde (RES) como fontes de dados

A disponibilidade de bases de dados de saúde em formato eletrônico abriu a possibilidade de gerar evidências e discernimentos sistemáticos em larga escala sobre a aplicação dos cuidados de saúde aos pacientes. Essa pesquisa é denominada Estudos Observacionais de Desfechos, e utiliza dados clínicos longitudinais no nível do paciente para descrever e compreender a patogênese da doença e o efeito de eventos clínicos, bem como intervenções de tratamento, sobre a progressão da doença. Constitui uso secundário dos dados o aproveitamento de informações que estão sendo coletadas normalmente para outros fins que não a pesquisa: dados administrativos, solicitações de saúde complementar, faturamento, contabilidade geral, fontes de saúde pública, biobancos, fontes farmacêuticas e o Registro Eletrônico de Saúde (RES).

O uso desses dados têm como vantagens o baixo custo de coleta e a possibilidade de utilizar amostras maiores, proporcionando maior força para detectar pequenas diferenças ou eventos raros, dispensando o contato direto com o paciente. Possibilita também, maior diversidade metodológica, permitindo que modificações no estudo possam ser implementadas diferentemente dos projetos de pesquisa de maior rigor de protocolos, como os ensaios randomizados. Além disso, um relatório do *New England Journal of Medicine*[3] aponta que "foram encontradas poucas evidências de que

[3] Benson K. A Comparison of Observational Studies and Randomized, Controlled Trials. The New England Journal of Medicine. 2000;9.

as estimativas dos efeitos do tratamento em estudos observacionais relatados após 1984 sejam consistentemente maiores ou qualitativamente diferentes daquelas obtidas em ensaios controlados e randomizados".

As desvantagens estão relacionadas à falta da padronização na coleta dos dados, que afeta a qualidade dos dados registrados; a perda potencial de seguimento após um determinado ponto no tempo; a ausência de informações clínicas relativas a um paciente que podem ser importantes para as análises de interesse.

Os conjuntos de dados de origem diferem uns dos outros em formato e representação de conteúdo e por vezes introduzem viés nos dados. Tudo isso torna a pesquisa robusta, reproduzível e automatizada um desafio significativo. Uma solução é a padronização dos dados e da sua representação. Isso permite que métodos e ferramentas operem em dados de origem díspar, e que métodos analíticos desenvolvidos para um determinado conjunto de dados possam ser aplicados a qualquer outro conjunto de dados no mesmo formato.

4.1.2. Estudos de coorte observacionais retrospectivos

O uso secundário de registros eletrônicos de saúde são indicados para estudos observacionais, retrospectivos e comparativos, a partir de uma coorte extraída da base assistencial. Definimos "coorte" para significar um conjunto de pacientes que satisfazem um ou mais critérios de inclusão por um período de tempo; "observacional" para significar que não há intervenção ou atribuição de tratamento imposta pelo estudo; "retrospectivo" para significar que o estudo será conduzido usando dados já coletados antes do início do estudo; "design de coorte comparativo" para significar a comparação formal entre duas coortes, uma coorte alvo e uma coorte de referência, para o risco de um desfecho durante um período de tempo definido após a entrada da coorte.

4.1.3. Desafios da pesquisa científica na atualidade: reprodutibilidade

A reprodutibilidade é a possibilidade de um experimento ou estudo poder ser repetido, tanto pelo mesmo pesquisador quanto por um pesquisador atuando independentemente. Reproduzir um experimento é chamado de replicar o mesmo. A reprodutibilidade é um dos princípios fundamentais do método científico.

A pesquisa reprodutível exige que os conjuntos de dados, os códigos, software ou outros tipos de instruções de computador que foram usados para computar os resultados publicados devem ser fornecidos para a verificação ou a realização de análises alternativas.

4.1.4. A iniciativa OMOP e OHDSI

A necessidade de se gerar evidências clínicas acessíveis e confiáveis, acessando a experiência do atendimento de saúde de milhões de pacientes em todo o mundo, é uma realidade. A *Observational Health Data Sciences and Informatics* (OHDSI, pronunciado "Odyssey") baseou-se em aprendizados da iniciativa da *Observational Medical Outcomes Partnership* (OMOP), para transformar métodos de pesquisa em um conjunto de aplicativos e ferramentas de exploração que aproximam a pesquisa de campo do objetivo final de gerar evidências sobre todos os aspectos de saúde para atender às necessidades de pacientes, clínicos e todos os outros tomadores de decisão.

4.1.4.1. Os cegos e o elefante[4]

Como na parábola hindu onde um grupo de cegos tentam descrever um elefante apenas apalpando partes dele, cada prestador de serviço de saúde possui na sua base uma visão parcial da população.

Bases clínicas de hospitais possuem informação dos pacientes e de suas doenças, mas, não tem dados de pessoas sadias, e, mesmo dentro de um grupo de hospitais, a especialização influencia o perfil das bases. Por exemplo, em uma maternidade vou achar fundamentalmente informações de mulheres grávidas, num centro de especialidades cardiológicas terei predominância deste grupo de doenças.

Bases de administradoras de seguros de saúde tem dados de sinistros e também de pessoas sadias mas não possuem dados clínicos. Bases laboratoriais armazenam resultados de exames e o seu histórico para cada paciente, porém sem relação com um diagnóstico (ou a ausência de um). No fim, ninguém possui uma visão completa da população.

Estudos observacionais restritos a bases independentes estão sujeitos a vieses e deformações por não termos condições de capturar amostras randômicas da população completa.

A iniciativa OMOP/OHDSI se propõe a montar uma imagem que incorpore as diversas fontes de informações médicas através do uso de um modelo comum de dados para conseguir ter uma visão de toda a população, tanto atual quanto histórica.

4.1.4.2. Resumo histórico

Em 2007, reconhecendo o aumento da utilização de registros eletrônicos de saúde, o Congresso Americano requisitou ao *Food and Drug Administration* (FDA)[5] criar um novo programa de vigilância farmacológica para identificar de forma mais agressiva, potenciais problemas de segurança. O FDA lançou várias iniciativas para alcançar esse objetivo, incluindo o programa “Sentinela”, para criar uma rede de dados a nível nacional para o monitoramento de drogas, utilizando dados eletrônicos de detentores de informações de saúde.

[4] https://en.wikipedia.org/wiki/Blind_men_and_an_elephant
[5] FDA https://www.fda.gov/

Em particular, a *Pharmaceutical Research and Manufacturers of America* (PhRMA)[6], o FDA e a *Foundation for the National Institutes of Health* (FNIH)[7] criaram a *Observational Medical Outcomes Partnership* (OMOP), uma parceria público-privada estabelecida nos EUA. Este grupo de pesquisa interdisciplinar abordou uma tarefa que é fundamental para os objetivos mais amplos da comunidade de investigação: identificar os métodos mais confiáveis para a análise de grandes volumes de dados extraídos de fontes heterogêneas.

Empregando uma variedade de abordagens das áreas de epidemiologia, estatística e ciências da computação, OMOP procura responder a um desafio crítico: o que podem pesquisadores médicos aprender com a avaliação dessas novas bases de dados de saúde? Poderia uma abordagem única ser aplicada a várias doenças e poderiam as suas conclusões serem provadas? A comunidade de pesquisa médica poderia fazer mais estudos em menos tempo, utilizando menos recursos e obtendo resultados mais consistentes. No final, isso significaria um melhor sistema de monitoramento de drogas, dispositivos e procedimentos para a comunidade de saúde poder identificar com segurança os riscos e oportunidades para melhorar o atendimento ao paciente.

A peça central do projeto OMOP foi o desenvolvimento do modelo comum de dados (CDM), que representa dados de saúde de diversas fontes heterogêneas de uma forma consistente e padronizada. Este CDM é um modelo de informação no qual a codificação e as relações entre os conceitos são explícita e formalmente especificadas. Em conjunto com o CDM, foram construídos os vocabulários padronizados para a condução dos experimentos OMOP.

A OHDSI surgiu na continuação do projeto de cinco anos desenvolvido pela OMOP (2009-2013). A OHDSI realizou sua primeira reunião anual no Columbia University Medical Center em Nova York em outubro de 2014. Cinquenta e oito participantes revisaram a visão e os objetivos que deram origem à criação do OHDSI e formaram grupos de trabalho para abordar o modelo de dados comum, vocabulário, bases de conhecimento, métodos de estimativa, geração de fenótipo, caracterização clínica e definição de coorte. Os investigadores da pesquisa OMOP iniciaram o esforço OHDSI, e o laboratório de pesquisa mudou-se para a Fundação *Reagan-Udall* no âmbito do *Innovation in Medical Evidence Development and Surveillance* (IMEDS)[8].

A equipe do OHDSI adotou e continuou a manutenção deste modelo e de seus serviços de vocabulário associados. A abordagem geral do OHDSI é criar uma rede aberta de detentores de dados observacionais a partir da tradução dos dados para o CDM - OMOP. Cada elemento no banco de dados do participante deve ser mapeado para o vocabulário de CDM aprovado e colocado no esquema de dados. Em troca, essa abordagem possibilita implementar várias ferramentas existentes de exploração e geração de evidências e participar de estudos em todo o mundo, visto que, qualquer consulta pode ser executada em qualquer site sem necessidade de modificações.

[6] PhRMA https://www.phrma.org/
[7] FNIH https://fnih.org/
[8] IMEDS http://reaganudall.org/innovation-medical-evidence-development-and-surveillance

Análises globais e multicêntricas podem ser executadas de forma rápida e eficiente usando aplicativos ou programas desenvolvidos em um único site.

A OHDSI conta com mais de 140 colaboradores em 16 países e é composta por pesquisadores clínicos, cientistas da computação, bioestatísticos e profissionais do setor de saúde.

4.1.4.3. Onde achar: principais referências

As informações a respeito dos componentes da OHDSI podem ser classificadas em:

- Informações gerais: http://www.ohdsi.org - É o site principal;
- Código e instalações: https://github.com/OHDSI/ - Aqui está disponibilizado o código fonte de todas as ferramentas. Em particular destacamos: *Common Data Model* (https://github.com/OHDSI/CommonDataModel), com a definição completa do modelo e as implementações para os diversos bancos suportados; Broadsea (https://github.com/OHDSI/Broadsea), que disponibiliza uma versão em containers Docker do conjunto de ferramentas, ver também: repositório Docker com os componentes dockerizados (https://hub.docker.com/u/ohdsi/); OHDSI-In-a-Box (https://github.com/OHDSI/OHDSI-in-a-Box) que disponibiliza uma máquina virtual pronta para testes e demonstrações;
- Tutoriais e vídeos: Procure no Google por YouTube OHDSI (https://www.google.com/search?q=youtube+ohdsi), existe muita documentação e tutoriais em vídeo dos eventos anuais do grupo;
- Fórum: Para resolver dúvidas mais frequentes, consulte o fórum (http://forums.ohdsi.org/).

4.2. Arquitetura do sistema OHDSI

A arquitetura do sistema disponibilizado pela iniciativa OHDSI consiste em um modelo comum, o CDM-OMOP v5, e um conjunto de ferramentas construídas ao redor deste modelo para oferecer acesso e suporte a consultas e atualizações das informações nele contidas. O CDM-OMOP consiste em:

- O esquema CDM v5;
- Dados dos pacientes migrados de outras fontes de informação;
- Um conteúdo de Vocabulários;

Esse conjunto é armazenado em uma base relacional. Atualmente são suportadas as seguintes tecnologias SQL: BigQuery, Impala, Netezza, Oracle, Parallel Data Warehouse, Postgres, Redshift, e SQL Server.

Para acesso ao modelo são construídas as seguintes ferramentas:

- OHDSI Web API: fornece acesso padronizado ao modelo via serviços REST;
- ATLAS: Aplicação Web, que utiliza a API para permitir o acesso aos usuários comuns as informações contidas no modelo;

- WhiteRabbit e Rabbit in a Hat, ferramentas de ETL (Extração, Transformação e carga de dados) que auxiliam no processo de migração de outras fontes para o modelo;
- ATHENA: Aplicação web que permite o acesso e manutenção dos vocabulários utilizados no modelo;
- ARACHNE: Permite gerar e executar o código de um estudo (R e SQL), coletar estatísticas e inferência de evidências;
- ACHILLES: Fornece estatísticas descritivas da base OMOP.

Atualmente, estas ferramentas estão sendo incorporadas no ATLAS. A Figura 4.1. apresenta os diversos componentes da arquitetura:

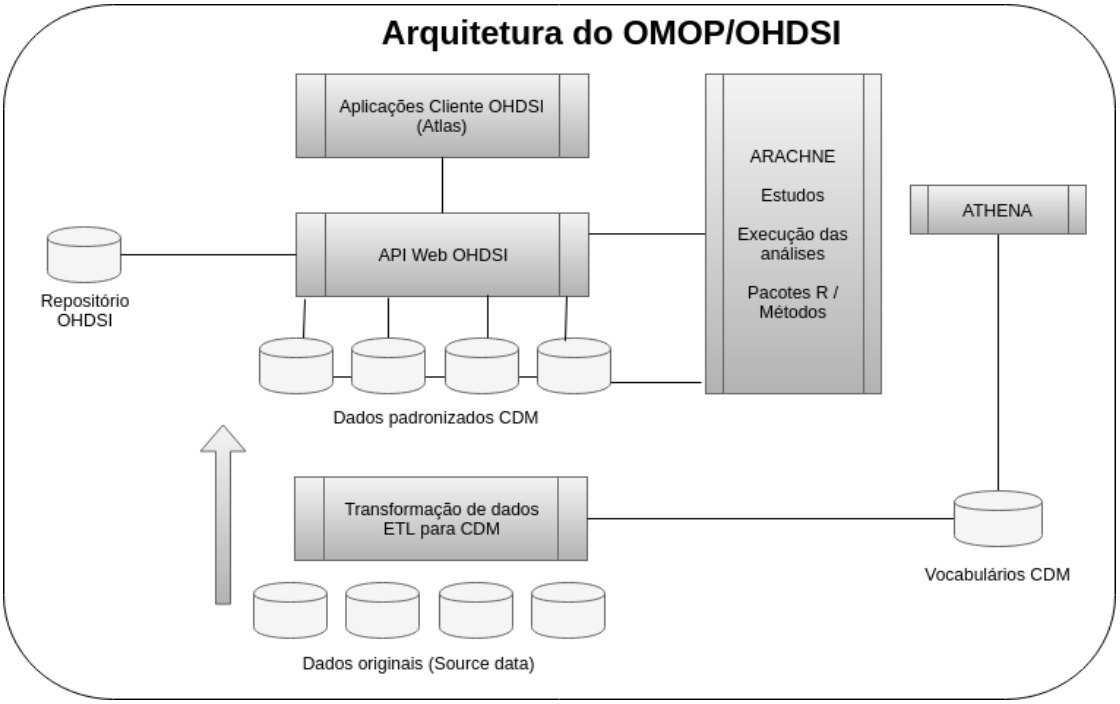


Figura 4.1. Componentes da arquitetura OHDSI

4.3. Modelo comum de dados (*Common Data Model* CDM-OMOP)

Nenhuma fonte única de dados observacionais fornece uma visão abrangente dos dados clínicos que um paciente acumula ao receber cuidados de saúde e, portanto, nenhuma delas é suficiente para atender a todas as necessidades da pesquisa observacional. Uma peça central do projeto OMOP foi o desenvolvimento do modelo comum de dados (CDM). A finalidade do modelo comum de dados é padronizar o formato e o conteúdo dos dados observacionais, oriundos de sistemas heterogêneos, para que aplicativos, ferramentas e métodos padronizados possam ser aplicados. O CDM representa dados de saúde de diversas fontes de uma forma consistente e padronizada.

O CDM é projetado para apoiar a realização de pesquisas, identificar e avaliar associações entre intervenções (exposição a medicamentos, procedimentos, mudanças na política de saúde, etc.) e os resultados causados por essas intervenções (ocorrências de condições, procedimentos, exposição a drogas etc.). Os resultados podem ser eficazes (benefício) ou adversos (risco de segurança). Muitas vezes, coortes específicas de pacientes (por exemplo, aqueles que tomam determinado medicamento ou sofrem de uma determinada doença) podem ser definidos para tratamentos ou resultados, usando eventos clínicos (diagnósticos, observações, procedimentos) que ocorrem em relacionamentos temporais pré-definidos. Com o seu conteúdo padronizado (através dos vocabulários padronizados), assegurará que os métodos de pesquisa possam ser sistematicamente aplicados para produzir de forma significativa resultados comparáveis e reprodutíveis.

O CDM é flexível o suficiente para armazenar dados do RES, dados de sinistros, bem como o vocabulário padronizado. Cada tabela contém um conjunto mínimo de campos que devem ser preenchidos. A rede de pacientes disponível para pesquisa no OHDSI inclui aproximadamente 84 bancos de dados, tanto clínicos quanto informativos, totalizando mais de 650 milhões de pacientes.

4.3.1. Modelo comum de dados para pesquisa

O CDM foi elaborado para incluir todos os elementos observacionais de dados de saúde que sejam relevantes para análise de casos de uso para apoiar a geração de evidências científicas confiáveis sobre a história natural da doença, cuidados médicos, identificação de informação demográfica, intervenções de saúde e resultados.

Portanto, o CDM é projetado para armazenar dados observacionais para permitir a pesquisa, sob os seguintes princípios:

- Adequação à finalidade: O CDM visa fornecer dados organizados de maneira ideal para a análise;
- Proteção de dados: Todos os dados que possam comprometer a identidade e a proteção dos pacientes, como nomes, datas, etc., são limitados. Exceções são possíveis quando a pesquisa exige expressamente informações mais detalhadas, como datas de nascimento precisas para o estudo de bebês;
- Domínios: os domínios são modelados em um modelo de dados relacionais centrados na pessoa e são identificados e definidos separadamente em um modelo de relacionamento de entidade;
- Vocabulários Padronizados: Para padronizar o conteúdo desses registros, o CDM se baseia nos vocabulários padronizados contendo todos os conceitos de saúde padrão correspondentes necessários e apropriados, explicitamente representando todos os fatos e eventos clínicos. Com poucas exceções, não há informações textuais nas tabelas do CDM;
- Reutilização de vocabulários existentes: Se possível, esses conceitos são aproveitados de organizações ou iniciativas de padronização nacional ou de

indústria ou definição de vocabulário, como a *National Library* of *Medicine*[9], o *Department of Veterans' Affairs*[10], o Centro de Controle e Prevenção de Doenças, etc;

- Manutenção de códigos-fonte: Embora todos os códigos sejam mapeados para os Vocabulários Padronizados, o modelo também armazena o código-fonte original para garantir que nenhuma informação seja perdida;
- Neutralidade da tecnologia: O CDM não requer uma tecnologia específica. Ele pode ser realizado em qualquer banco de dados relacional, como Oracle, SQL Server etc., ou como conjuntos de dados analíticos do SAS;
- Escalabilidade: O CDM é otimizado para processamento de dados e análise computacional para acomodar fontes de dados que variam em tamanho, incluindo bancos de dados com milhões de pessoas e bilhões de observações clínicas;
- Compatibilidade retroativa: Todas as alterações dos CDMs anteriores são claramente delineadas no repositório do github[11]. Versões mais antigas do CDM podem ser facilmente criadas a partir do CDM versão 5 sem perda de nenhuma informação.

O CDM foi elaborado para incluir todos os elementos observacionais de dados de saúde que são relevantes para análise de casos de uso para apoiar a geração de evidências científicas confiáveis sobre a história natural da doença, assistência médica, efeitos de intervenções médicas, identificação de informações demográficas, intervenções de saúde e resultados. A versão 5 do CDM foi apresentada em 14 de outubro de 2014, disponível em[12]. A Figura 4.2. apresenta uma cópia traduzida do CDM-OMOP na versão 5.

Além dos dados da pessoa, da condição, droga, procedimento e informações de visitas, o modelo provê informações de custo e do provedor do atendimento. Esta proposta tende a apoiar a economia da saúde e estudos de casos de uso de tratamento médico, incluindo a segurança de dispositivos médicos, eficácia comparativa e qualidade de saúde.

Em 11 de outubro de 2018 foi publicada uma especificação do modelo comum de dados CDM-OMOP, versão 6.0. O manual técnico e as diferenças entre as versões estão disponível em[13].

4.3.2. Outros modelos

Existem outros modelos de dados utilizados para organizar as informações médicas. Dentre eles podemos destacar os seguintes:

[9] NLM https://www.nlm.nih.gov/
[10] VA https://www.va.gov/
[11] CDM repositório github https://github.com/OHDSI/CommonDataModel
[12] CDM v5 https://github.com/OHDSI/CommonDataModel/blob/v5-historical/OMOP%20CDM%20v5.pdf
[13] CDM v6 https://github.com/OHDSI/CommonDataModel/blob/master/OMOP_CDM_v6_0.pdf

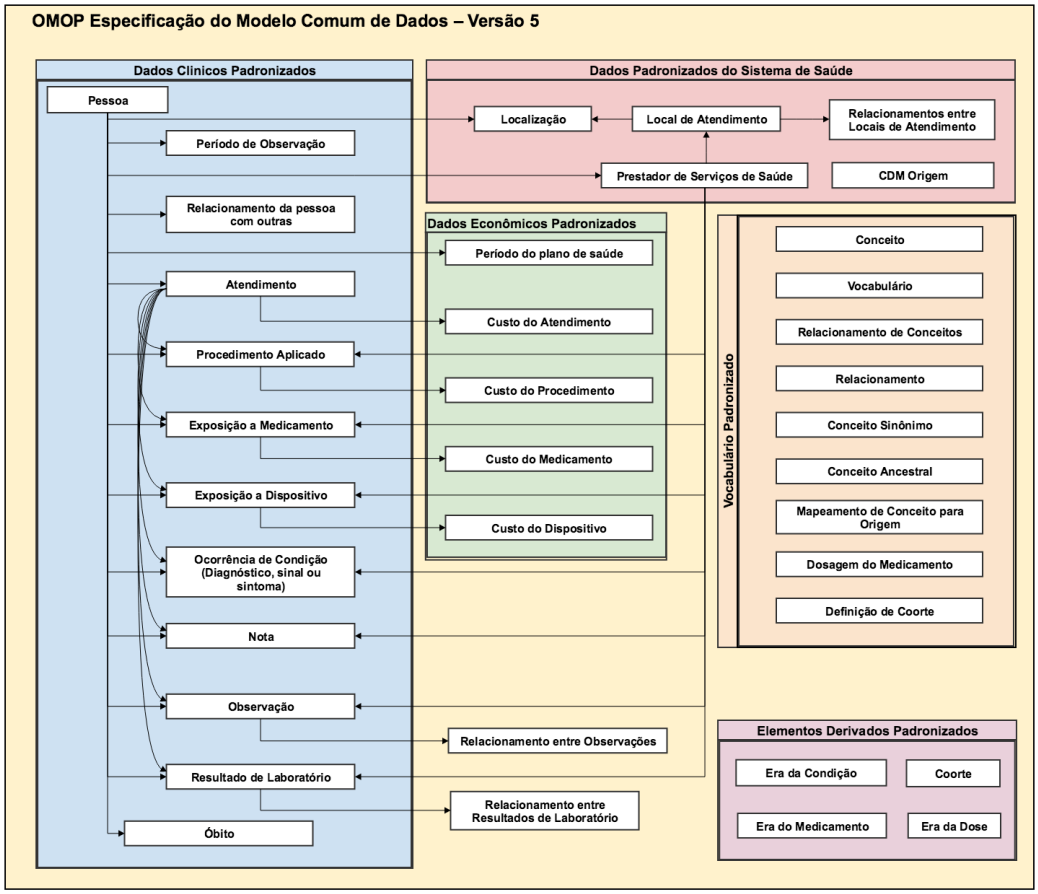


Figura 4.2. Modelo CDM OMOP versão 5

- Sentinel Common Data Model: A Sentinel Initiative da Food and Drug Administration (FDA) dos EUA é um esforço de longo prazo para melhorar a capacidade do FDA de identificar e avaliar questões de segurança de produtos médicos que utiliza dados de saúde eletrônicos pré-existentes de várias fontes para monitorar a segurança de produtos médicos regulamentados; [https://www.sentinelinitiative.org/sentinel/data/distributed-database-common-data-model]
- PCORnet Common Data Model: A rede do National Patient-Centered Clinical Research Network financiada pelo Patient-Centered Outcomes Research Institute (PCORI), integra dados de 11 redes heterogêneas para permitir pesquisa de efetividade comparativa em larga escala. [https://pcornet.org/pcornet-common-data-model/]
- i2b2: O projeto Informatics for Integrating Biology and the Bedside (i2b2) [https://www.i2b2.org/index.html] suporta a interoperabilidade entre as fontes de informação por meio de uma abordagem orientada pela ontologia para o armazenamento de dados. [https://academic.oup.com/jamia/article/23/5/909/2379861]

- Base Pauá: Trabalho de doutorado que resultou na elaboração um modelo comum de dados para o sistema de informações hospitalares do InCor-HCFMUSP, hospital de referência em doenças cardiovasculares, na cidade de São Paulo, Brasil. Este modelo, centrado no paciente, foi pensado para poder ser utilizado em outros sistemas e para isolar as particularidades do sistema subjacente sendo possível elaborar consultas genéricas que são aplicadas ao modelo. A falta de vocabulários padronizados fez com que as consultas genéricas se limitassem a diagnósticos codificados em CID-10, na atualidade, o único vocabulário padronizado de uso obrigatório no Brasil. [http://www.teses.usp.br/teses/disponiveis/5/5131/tde-04082016-160129/pt-br.php]

4.3.3. Estrutura das tabelas CDM

O CDM define as estruturas de tabela centrada na pessoa. No lugar de uma tabela central com fatos, temos uma coleção dos mesmos diferenciada por domínio: procedimentos, condição, droga, medição, observação, etc.
Isso permite uma visão longitudinal de todos os eventos de saúde de um indivíduo. Estes eventos estão ligados aos prestadores de cuidados de saúde, como médicos, consultórios médicos, postos de saúde, ambulatórios, hospitais, departamentos hospitalares, etc.).

Para representar os domínios relevantes, o CDM contém as 39 tabelas descritas a seguir:

Vocabulários Padronizados:

CONCEPT - Essa tabela contém registros que identificam exclusivamente cada unidade usada para expressar informações clínicas. Os conceitos são derivados de vocabulários de origem, que representam informações clínicas em diferentes domínios (por exemplo, condições, medicamentos, procedimentos) por meio do uso de códigos e descrições associadas. Alguns conceitos são designados como conceitos padrão, o que significa que esses conceitos podem ser usados no CDM e em análises padronizadas. Cada conceito padrão possui um domínio principal, que define a localização em que o conceito deveria ser observado dentro do CDM.

VOCABULARY - Essa tabela inclui uma lista dos vocabulários coletados de várias fontes ou criados pela comunidade OMOP. Essa tabela de referência é preenchida com um único registro para cada fonte de vocabulário e inclui um nome descritivo e outros atributos associados ao vocabulário.

DOMAIN - Essa tabela inclui uma lista dos domínios dos elementos de dados que estão contidos no CDM. Um domínio define o conjunto de conceitos permitidos para cada

campo padronizado. Essa tabela de referência é preenchida com um único registro para cada domínio e inclui um nome descritivo para o domínio.

CONCEPT_CLASS - Essa tabela inclui uma lista das classificações usadas para diferenciar conceitos dentro de um determinado vocabulário. Essa tabela de referência é preenchida com um único registro para cada classe conceitual e inclui um nome descritivo para a classe conceitual.

CONCEPT_RELATIONSHIP - Essa tabela contém registros que definem relacionamentos entre dois conceitos e a natureza do relacionamento. O tipo de relacionamento é definido na tabela RELATIONSHIP e é geralmente classificado como hierárquico (pai-filho) ou não hierárquico (lateral). Todos os relacionamentos são direcionais e cada relação de conceito é representada duas vezes simetricamente na tabela de relacionamento de conceito. Por exemplo, os dois conceitos SNOMED de 'infarto agudo do miocárdio da parede anterior' e 'infarto agudo do miocárdio' têm duas relações conceituais: 1- 'infarto agudo do miocárdio da parede anterior' 'é um' 'infarto agudo do miocárdio', e 2- 'Infarto agudo do miocárdio' 'agrupa o' 'Infarto agudo do miocárdio da parede anterior'.

RELATIONSHIP - Essa tabela fornece uma lista de referência de todos os tipos de relacionamentos permitidos que podem ser usados para associar quaisquer dois conceitos na tabela CONCEPT_RELATIONSHIP. Os relacionamentos são classificados como hierárquicos (pai-filho) ou não-hierárquicos e são usados para determinar quais registros de relacionamento de conceito devem ser incluídos na tabela CONCEPT_ANCESTOR.

CONCEPT_SYNONYM - Essa tabela é usada para armazenar nomes alternativos para um conceito. Cada sinônimo é atribuído a seu próprio identificador exclusivo e contém o texto de uma descrição e o identificador do conceito que ele representa.

CONCEPT_ANCESTOR - Essa tabela contém registros que definem os relacionamentos hierárquicos entre todos os conceitos padrão. A tabela CONCEPT_ANCESTOR permite a identificação de relacionamentos hierárquicos em várias etapas, como medicamentos de marca que se enquadram em uma classe terapêutica ou diagnóstico específico que são classificados dentro de um sistema específico de classes.

SOURCE_TO_CONCEPT_MAP - Essa tabela é uma estrutura de dados legada no CDM, recomendada para uso em processos de extração, transformação e carregamento (ETL), para manter códigos fonte locais que não estão disponíveis como conceitos nos vocabulários padronizados e para estabelecer mapeamentos para cada código-fonte em um conceito padrão que pode ser usado para preencher as tabelas do CDM.

DRUG_STRENGTH - Essa tabela contém conteúdo estruturado sobre a quantidade ou concentração e unidades associadas de um ingrediente específico dentro de um determinado medicamento. A tabela de composição do medicamento é um arquivo suplementar para apoiar a análise padronizada da utilização de medicamentos usando conceitos do vocabulário RxNorm[14]. Um conceito clínico de medicamento que contenha múltiplos ingredientes ativos resultará em um registro do medicamento para cada ingrediente ativo.

COHORT_DEFINITION - Essa tabela contém registros para definir cada coorte derivada por meio de uma descrição e sintaxe associadas. Coortes são elementos derivados de um conjunto de assuntos que satisfazem um determinado conjunto de critérios de inclusão por um período de tempo. A tabela COHORT_DEFINITION fornece uma estrutura padronizada para manter as regras de inclusão de um assunto em uma coorte, e pode armazenar código para instanciar a coorte dentro do CDM.

ATTRIBUTE_DEFINITION - Essa tabela contém registros para definir cada atributo por meio de uma descrição e sintaxe associadas. Atributos são elementos derivados que podem ser selecionados ou calculados para um assunto dentro de uma coorte. A tabela ATTRIBUTE_DEFINITION fornece uma estrutura padronizada para manter as regras do cálculo de covariáveis para um sujeito em uma coorte, e pode armazenar código para instanciar os atributos para uma dada coorte dentro do CDM.

Meta-dados normalizado:

CDM_SOURCE - Essa tabela contém detalhes sobre a fonte de dados e o processo utilizado para transformar os dados para o CDM. Se um banco de dados de origem for derivado de vários fontes de dados, espera-se que a integração dessas diferentes fontes seja documentada nas especificações de ETL.

Dados clínicos padronizados:

Essas tabelas contêm as informações básicas sobre os eventos clínicos que ocorreram longitudinalmente durante os períodos de observação válidos para cada pessoa, bem como as informações demográficas.

PERSON - Essa tabela contém registros que identificam exclusivamente cada paciente nos dados de origem que tem tempo em risco para ter eventos clínicos registrados nos sistemas de origem. Uma pessoa deve ter pelo menos um período de observação para definir o tempo em risco, mas pode ou não ter quaisquer eventos clínicos registrados nos outros domínios de dados. Cada registro pessoal tem atributos demográficos

[14] RxNorm https://www.nlm.nih.gov/research/umls/rxnorm/

associados que são considerados constantes para o paciente ao longo de seus períodos de observação. Todos os outros domínios de dados no nível do paciente têm uma referência de chave estrangeira ao domínio da pessoa.

OBSERVATION_PERIOD - Essa tabela contém registros que definem com exclusividade os períodos de tempo em que uma pessoa está em risco de ter eventos clínicos registrados nos sistemas de origem. Uma pessoa pode ter um ou mais períodos de observação disjunta, durante os quais as análises podem assumir que os eventos clínicos seriam capturados se observados, e fora do qual nenhum evento clínico poderia ser registrado.

SPECIMEN - Essa tabela contém os registros que identificam cada amostra biológica de uma pessoa.

DEATH - Essa tabela contém o evento clínico de como e quando uma pessoa morre. Uma pessoa pode ter até um registro se os sistemas de origem contiverem evidências de que ele é falecido. Todas as pessoas que estavam vivas durante todos os períodos de observação não devem conter nenhuma informação na tabela DEATH.

VISIT_OCCURRENCE - Essa tabela contém os períodos de tempo em que uma pessoa recebe continuamente serviços médicos de um ou mais prestadores de serviços em uma instalação em um determinado ambiente dentro do sistema de assistência médica. As visitas são classificadas em quatro configurações: atendimento ambulatorial, internação, sala de emergência e cuidados de longa duração. As pessoas podem fazer a transição entre essas configurações ao longo de um episódio de atendimento. As visitas de internação são definidas pelo período de tempo entre a admissão e a alta de uma instalação hospitalar específica. As consultas ambulatoriais são definidas como período de tempo dentro do consultório de um provedor específico, que é esperado para menos de 1 dia. Visitas de cuidados de longo prazo são definidas como o período de tempo em que uma pessoa é tratada dentro de uma instalação específica de cuidados de longo prazo.

PROCEDURE_OCCURRENCE - Essa tabela contém registros de atividades ou processos solicitados e / ou executados por um profissional de saúde para que o paciente tenha uma finalidade diagnóstica e / ou terapêutica.

DRUG_EXPOSURE - Essa tabela captura registros sobre a utilização de uma substância bioquímica com um efeito terapêutico fisiológico quando ingerida ou de outra forma introduzida no corpo. As drogas incluem medicamentos prescritos e de venda livre, vacinas e terapias biológicas. A exposição a medicamentos é inferida a partir de eventos clínicos associados a pedidos, prescrições escritas, dispensas de

farmácia, administrações de procedimentos e outras informações relatadas pelo paciente.

DEVICE_EXPOSURE - Essa tabela captura registros sobre a exposição de uma pessoa a um objeto físico ou instrumento que é utilizado para fins de diagnóstico ou terapêuticos. Os dispositivos incluem objetos implantáveis (marcapassos, stents, articulações artificiais), equipamentos e suprimentos médicos duráveis (bandagens, muletas, seringas) e outros instrumentos usados em procedimentos médicos (suturas, desfibriladores, etc.).

CONDITION_OCCURRENCE - Essa tabela captura os registros de uma doença ou de uma condição médica com base na avaliação de um provedor ou relatada por um paciente.

MEASUREMENT - Uma medida é a captura de um valor estruturado (numérico ou categórico) obtido através do exame sistemático de uma pessoa ou amostra. A tabela MEASUREMENT captura ordens de medição e resultados de medição. O domínio de medição pode conter resultados laboratoriais, sinais vitais ou descobertas quantitativas de relatórios de patologia.

NOTE - Essa tabela captura informações não estruturadas que foram gravadas por um provedor ou paciente em notas de texto livre em uma determinada data.

OBSERVATION - Essa tabela capta qualquer fato clínico sobre um paciente obtido no contexto de um exame, questionamento ou procedimento. O domínio de observação suporta a captura de dados não representados por outros domínios, incluindo medidas não estruturadas, histórico médico e histórico familiar.

FACT_RELATIONSHIP - Essa tabela contém registros para detalhar as relações entre fatos em um domínio ou entre dois domínios e a natureza do relacionamento. Exemplos de tipos de relacionamentos de fatos incluem: relacionamentos pessoais (ligação mãe-filho), relacionamentos no local de cuidados (representando a estrutura organizacional hierárquica de instalações dentro dos sistemas de saúde), exposições de medicamentos fornecidas devido a condição indicada associada, dispositivos usados durante o curso de um procedimento e as medidas derivadas de uma amostra. Todos os relacionamentos são direcionais e cada relacionamento é representado duas vezes simetricamente na tabela de relacionamento de fatos. Por exemplo, duas pessoas (PERSON_ID = 1 é a mãe de PERSON_ID = 2) têm dois relacionamentos de fatos: 1- 'PERSON_ID 1' 'mãe de' 'PERSON_ID 2' e 2 'PERSON_ID 2' 'filho de' 'PERSON_ID 1'.

Dados padronizados do sistema de saúde:

LOCATION - Essa tabela representa uma maneira genérica de capturar localização física ou informações de endereço. Os locais são usados para definir os endereços de pessoas e locais de atendimento.

CARE_SITE - Essa tabela contém uma lista de unidades organizacionais onde a prestação de cuidados de saúde é praticada (consultórios, alas, hospitais, clínicas, etc.).

PROVIDER - Essa tabela contém uma lista provedores de assistência à saúde identificados exclusivamente. Estes são tipicamente médicos e enfermeiros.

Dados padronizados de economia em saúde: Essas tabelas contêm informações de custo sobre assistência médica. Dependem do sistema de prestação de cuidados de saúde em que a população de doentes está envolvida, que pode variar significativamente em diferentes países. No entanto, o modelo atual do CDM está focado no sistema de saúde dos EUA.

PAYER_PLAN_PERIOD - Essa tabela captura registros que detalham o período de tempo em que uma pessoa está continuamente inscrita em uma estrutura de benefício de plano de saúde específico de um determinado pagador. Cada pessoa que recebe cuidados de saúde e está coberta por benefícios de saúde está sujeita a um plano definido pelo pagador para a pessoa ou sua família. Para uma dada política de benefícios, pode haver um ou mais planos ativos por determinados períodos de tempo, definindo o custo dos serviços de saúde fornecidos.

VISIT_COST - Essa tabela captura os custos da visita de saúde de um paciente que não estão relacionados a procedimentos, medicamentos ou dispositivos específicos usados no encontro.

PROCEDURE_COST - Essa tabela captura o custo de um procedimento executado em uma pessoa. As informações sobre o custo são derivadas apenas dos valores pagos pelo procedimento.

DRUG_COST - Essa tabela captura registros que indicam o custo de uma droga de exposição. A informação sobre o custo é definida pela quantidade de dinheiro pago pela pessoa e pagador pelo medicamento, bem como pelo custo cobrado do medicamento.

DEVICE_COST - Essa tabela captura o custo de um dispositivo médico ou fornecimento usado em uma pessoa. As informações sobre o custo são derivadas apenas dos valores pagos pelo dispositivo.

Elementos derivados padronizados: São tabelas montadas a partir dos dados já descritos através de algoritmos ou de seleções feitas utilizando os dados. Por exemplo, a partir das informações de DRUG_EXPOSURE, são geradas as eras, intervalos contínuos de exposição ao medicamento ou intervenção.

COHORT - Essa tabela contém registros derivados como um conjunto de assuntos que satisfazem determinados critérios de inclusão por um período de tempo. A definição da coorte está contida na tabela COHORT_DEFINITION. Exemplos de coortes podem incluir pacientes diagnosticados com uma condição específica, pacientes expostos a um determinado medicamento ou provedores que realizaram um procedimento específico.

COHORT_ATTRIBUTE - Essa tabela contém atributos associados a cada assunto dentro de uma coorte, conforme definido por um determinado conjunto de critérios de inclusão por um período de tempo. A definição do atributo de coorte está contida na tabela ATTRIBUTE_DEFINITION. Exemplos de atributos de coorte podem ser idade, índice de massa corpórea ou pontuação de comorbidades.

Eras: Uma era é definida como o intervalo de tempo durante o qual se presume que a pessoa tem uma determinada condição ou ficou exposta a um determinado princípio ativo. As eras são calculadas utilizando algoritmos padronizados a partir das informações de datas. Cada era corresponde a uma ou mais exposições que formam um intervalo contínuo. As eras são calculadas no momento da transformação da base. Temos DRUG_ERAS para medicamentos, DOSE_ERAS para doses constantes de medicamentos e CONDITION_ERAS para condições.

Por exemplo, no caso de medicamentos, as DRUG_ERAS são calculadas a partir das informações das datas de dispensação, ou DRUG_EXPOSURE. Uma pessoa tem 4 prescrições para a droga A (A1, A2, A3, A4), válida para 60 dias de dispensa. A pessoa também tem duas prescrições para a droga B (B1, B2). O diagrama da Figura 4.3. mostra a situação.

Para definir a era da droga para o medicamento A, o momento, a duração, a sobreposição e a persistência das prescrições do medicamento A devem ser consideradas. A3 foi preenchida antes do final esperado de A2. A4 foi preenchida após a conclusão do A3, mas dentro da janela de persistência para o medicamento A. Portanto, as quatro prescrições do medicamento A serão consolidadas em uma única era de medicamentos (Drug Era1), com o início da receita A1 registrado como o início a data do registro consolidado e a data final da prescrição A4 registrada como a data final.

Como a janela de persistência foi excedida entre o preenchimento das duas prescrições para a droga B, elas são definidas como duas eras de drogas distintas. As datas de início e término de Drug Era2 e Drug Era3 são as datas de início e término das prescrições B1 e B2, respectivamente. Observe que nenhuma janela de persistência adicional está sendo adicionada no final da última exposição à droga.

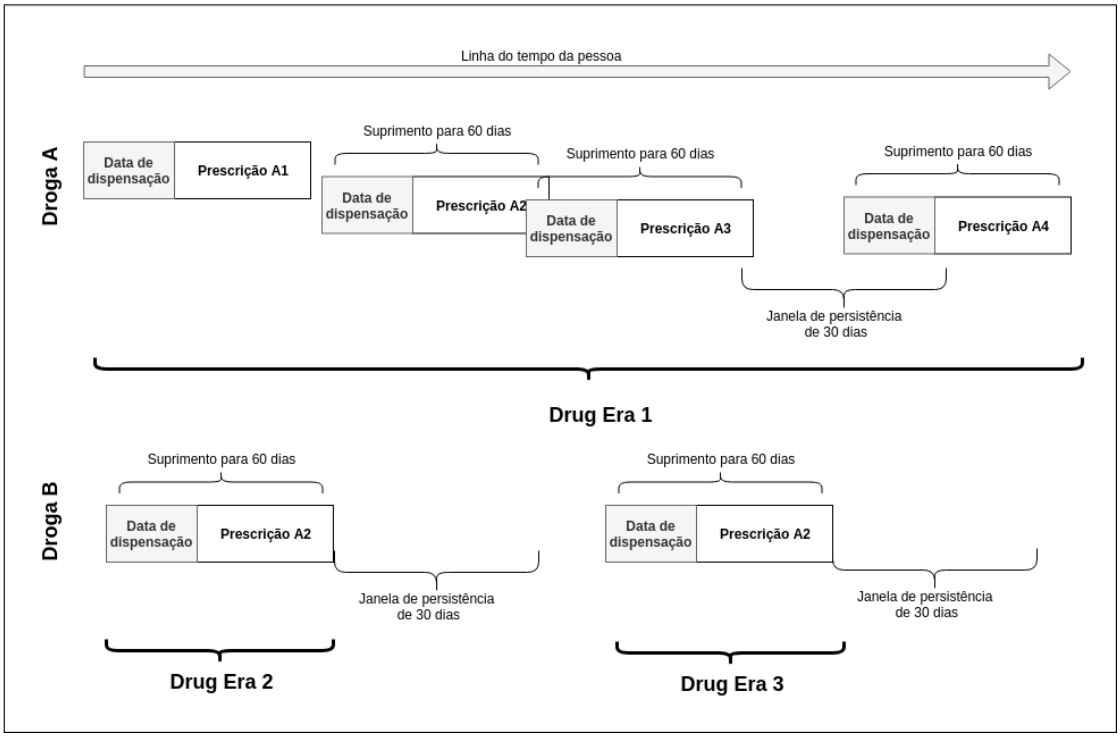


Figura 4.3. Definição de Eras

DRUG_ERA - É definida como um intervalo de tempo em que a pessoa é considerada exposta a um ingrediente ativo em particular.

DOSE_ERA - É definida como um intervalo de tempo em que a pessoa é considerada exposta a uma dose constante de um ingrediente ativo específico. É derivada das informações das tabelas de DRUG_EXPOSURE e DRUG_STRENGTH usando um algoritmo padronizado.

CONDITION_ERA - É definida como um período de tempo quando a pessoa é considerada como tendo uma determinada condição. Similar ao DRUG_ERA, CONDITION_ERA são períodos cronológicos da ocorrência da condição. Têm dois propósitos:

- Permite a agregação de condições crônicas que exigem frequentes cuidados contínuos, em vez de tratar cada ocorrência como um evento independente;
- Permite a agregação de múltiplas visitas médicas programadas para atender a mesma condição.

Por exemplo, considere uma pessoa que visita seu médico de cuidados primários e é encaminhado para um especialista. A pessoa visita o especialista, que define o diagnóstico e fornece o tratamento adequado para resolver a condição. Estas duas consultas médicas independentes devem ser agregadas em uma CONDITION_ERA.

4.3.4. Convenções do Modelo de Dados

Há um número de convenções que foram adotadas no CDM, sendo:

4.3.4.1. Convenções gerais das tabelas de dados

O CDM é independente de plataforma. Os tipos de dados são definidos genericamente usando ANSI SQL (VARCHAR, INTEGER, FLOAT, DATE, DATETIME, CLOB). O CDM não impõe o formato de data e data/hora.

Na maioria dos casos, o primeiro campo em cada tabela termina em '_ID', contendo um identificador de registro que pode ser usado como uma chave estrangeira em outra tabela.

4.3.4.2. Convenções gerais dos campos

Os nomes de variáveis em todas as tabelas seguem uma convenção:

- <entity>_SOURCE_VALUE:
  - informações dos dados de origem, geralmente usadas em ETL para mapear para CONCEPT_ID e não para serem usadas por nenhuma análise padrão
  - Ex: condition_source_value = '787.02' foi o código ICD-9 capturado como um diagnóstico da declaração administrativa.
- <entity> _ID:
  - Identificadores exclusivos para entidades chave, que podem servir como chaves estrangeiras para estabelecer relações entre entidades
  - Ex: person_id identifica exclusivamente cada indivíduo. visit_occurrence_id identifica exclusivamente um encontro PERSON em um ponto de atendimento.
- <entity> _CONCEPT_ID:
  - Chave estrangeira nos Vocabulários Padronizados (ou seja, o atributo standard_concept para o termo correspondente é true), que serve como base principal para todas as análises padronizadas
  - Ex: condition_concept_id =31967 contém valor de referência para o conceito SNOMED de ' Náusea '
- <entidade> _SOURCE_CONCEPT_ID:
  - Chave estrangeira nos vocabulários padronizados representando o conceito e terminologia utilizados nos dados de origem, quando aplicável
  - Ex: condition_source_concept_id = 35708202 denota o conceito de 'Náusea' na terminologia da terminologia MedDRA; o análogo condition_concept_id pode ser 31967, uma vez que SNOMED-CT é o vocabulário padronizado para a maioria dos diagnósticos e descobertas clínicas
- <entidade> _TYPE_CONCEPT_ID:
  - delineia a origem da informação da fonte, dentro dos vocabulários padronizados

- Ex: drug_type_concept_id pode permitir discriminar entre 'distribuição pharmacy' e 'receita escrita'

4.3.4.3. Representação do conteúdo através de conceitos

Os vocabulários padronizados contêm registros, ou conceitos, que identificam com exclusividade cada unidade fundamental de significado usada para expressar informações clínicas. Conceitos são derivados de vocabulários, que representam informações clínicas em diferentes domínios (por exemplo, condições, drogas, procedimentos) através do uso de códigos e descrições associadas. Alguns conceitos são designados como conceitos padrão, o que significa que esses conceitos podem ser usados como expressões normativas de uma entidade clínica dentro do CDM e dentro de análises padronizadas. Cada conceito padrão possui um domínio principal, que define a localização em que o conceito deveria ocorrer dentro do CDM.

Os conceitos podem representar categorias amplas (como "doença cardiovascular"), elementos clínicos detalhados ("infarto do miocárdio da parede anterolateral") ou características modificadoras e atributos que definem conceitos em vários níveis de detalhe (gravidade de uma doença, morfologia associada, etc.).

Os registros nas tabelas de vocabulários padronizados são derivados de vocabulários nacionais ou internacionais, como SNOMED-CT[15], RxNorm e LOINC[16], ou conceitos personalizados definidos para cobrir vários aspectos da análise de dados observacionais.

Nas tabelas de dados CDM o significado do conteúdo de cada registro é representado usando conceitos. Os conceitos são armazenados com seu concept_id como chaves estrangeiras para a tabela CONCEPT nos vocabulários padronizados, que contém conceitos necessários para descrever a experiência de assistência médica de um paciente. Se um conceito padrão não existir ou não puder ser identificado, é usado um conceito com o concept_id = 0, representando um conceito que não existente ou não é mapeável.

Os registros na tabela CONCEPT contêm todas as informações detalhadas (nome, relacionamentos, tipos etc.). A Tabela 4.1. apresenta a tabela CONCEPT, campos, se requerido ou não, tipo de dado e descrição.

[15] SNOMED-CT http://www.snomed.org/
[16] LOINC https://loinc.org/

Tabela 4.1. Tabela CONCEPT

Campo	Requerido	Tipo	Descrição
concept_id	sim	inteiro	Um identificador exclusivo para cada conceito em todos os domínios
concept_name	sim	varchar (255)	Um nome inequívoco, significativo e descritivo para o conceito
domain_id	sim	varchar (20)	Uma chave estrangeira para a tabela DOMAIN a qual o conceito pertence
vocabulary_id	sim	varchar (20)	Uma chave estrangeira para a tabela VOCABULARY indicando de qual fonte o conceito foi adaptado
concept_class_id	sim	varchar (20)	O atributo ou classe de conceito do Conceito. Exemplos são “Droga Clínica”, “Ingrediente”, “Localização Clínica” etc
standard_concept	não	varchar (1)	Esse sinalizador determina se o Conceito é um Conceito Padrão, um Conceito de Classificação ou um Conceito de Origem não padrão. Os valores permitidos são 'S' (Conceito Padrão) e 'C' (Conceito de Classificação), caso contrário, o conteúdo é NULL
concept_code	sim	varchar (50)	O código conceitual representa o identificador do Conceito no vocabulário de origem, como os IDs de conceitos SNOMED-CT, RxNorm, etc. Observe que os códigos conceituais não são exclusivos entre os vocabulários
valid_start_date	sim	date	A data em que o conceito foi gravado pela primeira vez. O valor padrão é 1-jan-1970, significando que o Conceito não possui data (conhecida) de início
valid_end_date	sim	date	A data em que o Conceito se tornou inválido porque foi excluído ou substituído (atualizado) por um novo conceito. O valor padrão é 31-Dec-2099, ou seja, o Conceito é válido até que se torne obsoleto
invalid_reason	não	varchar (1)	Motivo porque o conceito foi invalidado. Os valores possíveis são D (excluídos), U (substituídos por uma atualização) ou NULL quando valid_end_date tem o valor padrão

4.3.4.4. Conceito padrão, de classificação e de origem

Dentro de um Domínio, os códigos vêm de vários vocabulários, e frequentemente, têm significados idênticos ou sobrepostos. Para organização, para cada um deles é atribuído uma das três designações:

- Conceito padrão (standard_concept = 'S'): O conceito padrão é o conceito “oficial” que deve ser usado para representar uma entidade clínica única nas tabelas de dados clínicos padronizados. Seu código é gravado nos respectivos

campos concept_id. Normalmente, o conceito padrão é originado de vocabulários estabelecidos que têm uma cobertura abrangente e são bem definidos. Por exemplo, conceito obtido através do vocabulário SNOMED.

- Conceito de classificação (standard_concept = 'C'): conceito que têm um relacionamento hierárquico com o conceito padrão e, portanto, podem ser usados para consulta usando os registros da tabela CONCEPT_ANCESTOR. No entanto, eles próprios não podem aparecer nas tabelas de dados. Por exemplo, o conceito MedDRA para "COPD (chronic obstructive pulmonary disease)" têm relações hierárquicas com os conceitos padrão SNOMED-CT, que são todas as formas desta doença. Da mesma forma, o conceito 4283987 "ANTICOAGULANTES" do vocabulário VA Class[17] não pode aparecer nas tabelas DRUG_EXPOSURE ou DRUG_ERA, mas seus conceitos descendentes que possuem a classe de conceito "Ingrediente", "Droga Clínica" ou "Droga de Marca" podem.

  Os conceitos de classificação podem ser originados de diferentes vocabulários e não são exclusivos. Por exemplo, para a classe de medicamentos 'Anticoagulantes' há conceitos provenientes dos vocabulários NDF-RT[18], VA Class e ATC[19]. Observe também que a associação depende do vocabulário. Na maioria dos casos, a classificação é semelhante ou idêntica, mas não fornece uma definição padrão.
- Conceitos de origem (standard_concept = NULL): São todos os conceitos restantes que não são conceitos padrão ou de classificação. Observe que os conceitos podem alterar sua designação ao longo do tempo: se eles forem invalidados (valid_end_date for menor que 31-12-2099 e invalid_reason = 'D' (excluídos), ou 'U' (substituídos por uma atualização)), os antigos conceitos padrão ou de classificação se transformarão em conceitos de origem. Conceitos de origem só podem aparecer nos campos source_concept_id das tabelas de dados. Eles representam o código nos dados de origem. Cada conceito de origem é mapeado para um ou mais conceitos padrão durante o processo ETL. Se nenhum mapeamento estiver disponível, o conceito padrão com o concept_id = 0 será gravado no campo concept_id.

4.4. Vocabulários

Um dos principais problemas no agrupamento de fontes de dados diversas é a procura por uma definição comum do significado das informações nelas armazenadas.

Os vocabulários padronizados contêm registros, ou conceitos, que identificam de forma exclusiva cada unidade fundamental de significado usada para expressar

[17] VA Class (Veterans Affairs Drug Class) https://www.pbm.va.gov/clinicalguidance/drugclassreviews.asp
[18] NDF-RT (National Drug File - Reference Terminology) https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/NDFRT/
[19] ATC (Anatomical Therapeutic Chemical) https://www.whocc.no/atc_ddd_index/

informações clínicas em todas as tabelas de domínio CDM. São construídos com alguns princípios que representam o processo contínuo de melhoria e desenvolvimento:

1. Padronização: vários vocabulários usados em dados observacionais são consolidados em um formato comum. Isso alivia os pesquisadores de ter que entender e lidar com vários formatos e convenções de ciclo de vida diferentes dos vocabulários de origem;
2. Conceito padrão único: para cada entidade clínica existe apenas um conceito representando-o, denominado 'conceito padrão'. Outros conceitos equivalentes ou similares são designados como não-padrão e mapeados para os padrão;
3. Domínios: cada conceito recebe um domínio. Conceitos não-padrão também podem pertencer a mais de um domínio. Isso também define em qual tabela CDM uma entidade clínica deve ser colocada no momento da consulta;
4. Cobertura abrangente: todos os eventos relevantes de assistência médica do paciente (por exemplo, condições, procedimentos, exposições a medicamentos, etc.) e alguns dos artefatos administrativos do sistema de saúde (por exemplo, visitas, locais de atendimento, etc.) são cobertos pelo conceito de um domínio;
5. Hierarquia: dentro de um domínio, todos os conceitos são organizados em uma estrutura hierárquica. Isso permite consultar todos os conceitos (por exemplo, medicamentos) que são subordinados hierarquicamente sob um conceito de nível superior (por exemplo, uma classe de medicamentos). Isso implica abordar dois problemas distintos:
    - Cada conceito deve ter uma ou mais classificações (de baixo para cima);
    - Cada classificação deve conter todos os conceitos relevantes (de cima para baixo).
6. Relacionamentos entre conceitos dentro e entre vocabulários e mapeamentos de conceitos não padronizados para conceitos padrão;
7. Ciclo de vida mantendo a representação de dados atualizada, mas suportando o processamento de conceitos descontinuados e atualizados.

É importante notar que esses critérios têm o objetivo de servir à pesquisa observacional. Nesse sentido, os vocabulários padronizados diferem de grandes coleções com mapeamentos de equivalência de conceitos como o UMLS[20] que suporta indexação e pesquisa de toda a literatura biomédica. Os recursos da UMLS têm sido usados como base para a construção de muitos dos componentes do Vocabulário Padronizado, mas esforços adicionais significativos foram feitos para ajustar a estrutura:

- São estabelecidos vocabulários adicionais, principalmente para fins de metadados;
- Estão sendo adicionados mapeamentos e relacionamentos para obter uma cobertura abrangente. Se não for possível uma equivalência, serão criados relacionamentos de conceitos-padrão mais granulares, não padronizados, para níveis mais elevados;

[20] UMLS https://www.nlm.nih.gov/research/umls/

- É estabelecida uma estrutura de domínio abrangente e cada conceitos recebe um domínio ou uma combinação de domínios;
- É construída uma árvore hierárquica dentro dos domínios representando as classificações usadas na ciência médica e na prática clínica.

Descrevemos aqui a solução adotada pela iniciativa OHDSI, a forma como os diversos vocabulários são incorporados dentro da plataforma e como estes afetam as pesquisas.

4.4.1. Estrutura dos Vocabulários Padronizados

Os vocabulários padronizados contêm todos os conjuntos de códigos, terminologias, vocabulários, nomenclaturas, léxicos, tesauros, ontologias, taxonomias, classificações, abstrações e outros dados que são necessários para:

- Geração dos dados transformados (padronizados) do conjunto de dados brutos para o CDM;
- Pesquisar, consultar e extrair dados transformados e navegar pelas hierarquias de classes e abstrações inerentes aos dados transformados;
- Interpretar os significados dos dados.

Os dados em nível de paciente disponíveis no CDM exigem explicitamente a representação de todos os fatos e eventos clínicos usando conceitos dos vocabulários padronizados. Com poucas exceções, não há informações textuais nas tabelas do CDM. Portanto, os vocabulários padronizados são parte integrante do CDM. Geralmente, todos os componentes são *Open Source* - a menos que especificado de outra forma para alguns vocabulários comerciais.

A Tabela 4.2. apresenta os termos e descrições usados na estrutura dos vocabulários.

Os vocabulários padronizados fornecem uma representação padronizada de dados nos seguintes domínios clínicos:

- Dados demográficos: gênero, etnia, raça
- Condição
- Droga
- Procedimento
- Medição
- Observação
- Nota
- Dispositivo
- Espécime
- Unidade
- Visita
- Óbito
- Fornecedor
- Custo

Tabela 4.2. Tabela de termos

Termos	Descrição
Vocabulários padronizados	Contém um sistema de vocabulários, classificações, domínios e conceitos, todos consolidados em um formato comum e armazenados em um conjunto de tabelas CDM
Vocabulário	Um conjunto de códigos ou conceitos, incluindo, se disponíveis, relações entre eles, incluindo, se disponível, uma hierarquia, ontologia ou taxonomia dos conceitos. Muitos vocabulários são adotados de organizações nacionais ou internacionais, como o ICD-9-CM[1], o SNOMED-CT, o RxNorm, o Read[2]
Terminologia	Semelhante ao vocabulário e frequentemente usado como sinônimo
Esquema de codificação	Semelhante ao vocabulário e frequentemente usado como sinônimo
Classificação	Um sistema hierárquico de conceitos e relações conceituais que define classes semanticamente úteis, como estruturas químicas para drogas
Domínio	Uma categoria semântica clínica, como droga, condição, procedimento definido para todos os conceitos nos vocabulários padronizados
Conceito	Unidade básica de informação definida em cada vocabulário
Classe Concept	Um atributo de um conceito que caracteriza sua classificação dentro de um vocabulário. A diferença para a classificação é que uma classe concept é um único atributo sem qualquer estrutura hierárquica

4.4.2. Vocabulários padronizados

Os Vocabulários Padronizados estão organizados em domínios e vocabulários. Os domínios referem-se à natureza ou tipo de uma entidade clínica. Ele também define a tabela de dados do CDM onde um registro de dados deve ser armazenado. Vocabulários são conjuntos de conceitos importados de um padrão externo nacional ou internacional existente, ou criados pela equipe dos vocabulários padronizados, se nenhum padrão adequado estiver disponível. Não existe uma relação de um para um entre domínios e vocabulários. Alguns vocabulários são muito amplos, como SNOMED ou Read, e contêm conceitos de todos os domínios médicos. Da junção do SNOMED RT e do Read Codes surgiu o SNOMED CT. Outros vocabulários são específicos para um determinado domínio, como RxNorm for Drugs ou ICD9CM. Em muitos casos, os vocabulários são geralmente assumidos na comunidade como sendo de um único domínio, quando na verdade eles não são. Por exemplo, CPT4[21] e HCPCS[22] são esperados para conter apenas códigos de procedimento, mas na realidade contêm conceitos de observação, condição, dispositivo e droga. A Figura 4.4. apresenta um

[21] CPT (Current Procedural Terminology) https://www.ama-assn.org/
[22] HCPCS (Healthcare Common Procedure Coding System) https://www.ama-assn.org/

esquema dos domínios e a amplitude das relações de alguns vocabulários com cada domínio[23].

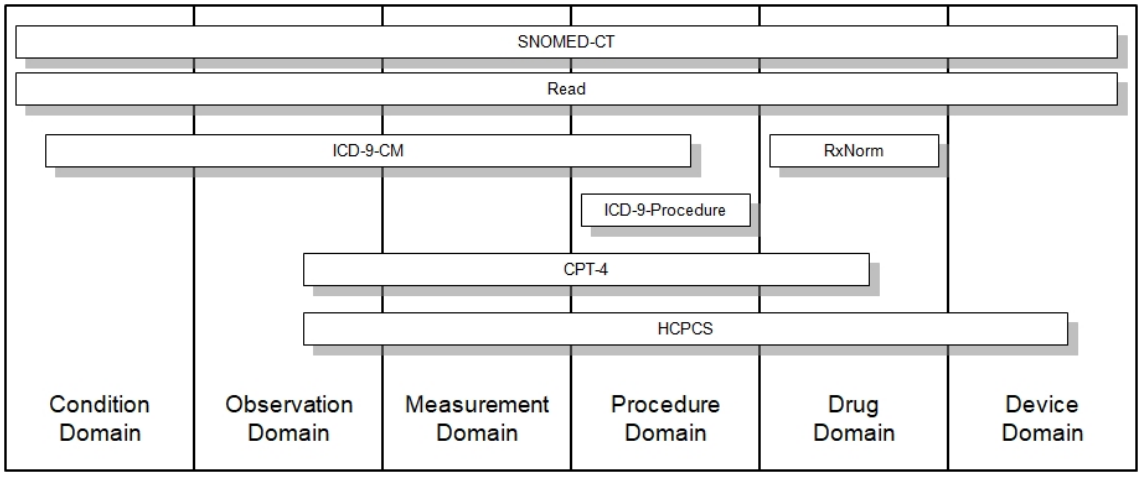


Figura 4.4. Domínios e vocabulários

As tabelas contêm informações detalhadas sobre os conceitos usados em todas as tabelas de fatos do CDM. O conteúdo das tabelas de vocabulários padronizados são mantidas centralmente como um serviço para a comunidade. Várias proposições foram feitas para o projeto das tabelas de vocabulários padronizados, sendo:

- Existe um esquema que acomoda todas as diferentes terminologias e classificações de origem;
- Todas as terminologias são carregadas na tabela CONCEPT;
- Cada termo carregado recebe um código novo como chave (concept_id). O código original da terminologia não é utilizado como identificador porque ele não é exclusivo entre terminologias;
- Alguns conceitos são declarados conceitos padrão, isto é, são usados para representar uma determinada entidade clínica nos dados. Todos os conceitos podem ser conceitos de fonte; eles representam como a entidade foi codificada na fonte. Os conceitos padrão são identificados por meio do campo standard_concept na tabela CONCEPT;
- Registros na tabela CONCEPT_RELATIONSHIP definem relações semânticas entre conceitos. Essas relações podem ser hierárquicas ou laterais;
- Os registros na tabela CONCEPT_RELATIONSHIP são usados para mapear códigos fonte para conceitos padrão, substituindo o mecanismo da tabela SOURCE_TO_CONCEPT_MAP usada em versões anteriores de vocabulários padronizados. A tabela SOURCE_TO_CONCEPT_MAP é mantida como um

[23] Documentação Vocabulários
http://www.ohdsi.org/web/wiki/doku.php?id=documentation:vocabulary:domains_and_vocabularies

auxílio opcional aos códigos de contabilidade não encontrados nos vocabulários padronizados;

- As cadeias de relacionamentos hierárquicos são registradas na tabela CONCEPT_ANCESTOR. Os relacionamentos de ancestralidade são registrados apenas entre os conceitos padrão que são válidos (não preteridos) e são conectados por meio de relacionamentos válidos e hierárquicos na tabela RELATIONSHIP (sinalizador define_ancestry). A vantagem dessa abordagem reside na preservação de códigos e relacionamentos entre eles, sem a adesão às várias estruturas de dados de origem diferentes, um design simples para acesso padronizado e a otimização do desempenho para a análise OHDSI. Navegação entre conceitos padrão não requer conhecimento do vocabulário de origem. Finalmente, a abordagem é escalável e vocabulários futuros podem ser facilmente integrados. Por outro lado, é necessária uma transformação extensiva de dados de origem para o vocabulário e nem toda estrutura de dados de origem e hierarquia de origem podem ser retidas.

Atualmente, 81 vocabulários fazem parte dos vocabulários padronizados. Muitos deles são adotados de fontes de terceiros, que os desenvolvem e os mantêm para fins específicos, como, ICD10[24] ou SNOMED-CT. A consolidação dos vocabulários em uma forma padronizada requer uma série de decisões e convenções. Um grupo trata da organização dos vocabulários dentro dos domínios clínicos, a implementação específica de cada vocabulário nos vocabulários padronizados, e o mapeamento de conceitos dentro e entre os vocabulários. Também fornece orientação sobre como aplicá-los para transformação de dados de origem no CDM e sobre a consulta de dados, uma vez estabelecidos no formato CDM. Os vocabulários padronizados estão na versão 5.x. e todos os conceitos nas versões anteriores ainda estão disponíveis e identificados usando os mesmos IDs de conceito.

4.4.2.1. Mapeamento de conceitos

O mapeamento é o processo para transformar um conceito em outro. As tabelas de dados clínicos do CDM permitem apenas conceitos padrão. Todos os outros códigos usados nos bancos de dados de origem precisam ser traduzidos para os conceitos padrão. O mapeamento é feito por meio de registros na tabela CONCEPT_RELATIONSHIP. Eles conectam cada conceito a um conceito padrão através de um número especial de relationship_id (maps to e maps to value).

Relacionamentos **'Maps to'**: Os conceitos que participam do mapeamento 'mapear para' são conceitos de origem e conceitos padrão. O mapeamento tenta mapear para o conceito de destino equivalente. Equivalente significa que ele carrega o mesmo significado e, mais importante, os filhos na hierarquia (se houver algum) também são equivalentes ou cobrem o mesmo espaço semântico. Se um conceito equivalente não

[24] ICD10 https://www.who.int/classifications/icd/icdonlineversions/en/

estiver disponível, o mapeamento tentará corresponder a um conceito mais amplo. Isso garante que uma consulta no vocabulário de destino recupere os mesmos registros, como se tivessem sido consultados no vocabulário de origem original.

Geralmente, conceitos de origem e conceitos padrão são mapeados:

- Conceitos de origem são mapeados para um ou vários conceitos padrão. Se eles forem mapeados para mais de um conceito padrão, então, na tabela CDM resultante, mais de um registro será gravado para cada registro na origem;
- Os conceitos padrão também são mapeados para os conceitos padrão, geralmente este é um mapa para si mesmo.

Os conceitos de classificação (standard_concept = 'S') não possuem um mapeamento para um conceito padrão.

Relacionamentos '**Maps to value'**: essas relações são projetadas para distinguir entre observação e medidas e seus resultados. Por exemplo, ICD9CM V12.71 concept_id 44820383 "História pessoal de doença ulcerosa péptica" tem uma relação "Maps to" para SNOMED 4214956 "História de descoberta clínica em questão" (Domínio de Observação) e outro relacionamento "Maps to value" para SNOMED 4027663 "Úlcera péptica" (domínio de condição).

'Perdas' devido ao mapeamento: há uma preocupação significativa sobre perdas do mapeamento de um vocabulário para outro, sobre a qualidade dos dados e a capacidade de identificar com segurança pacientes para uma coorte, dado os critérios de inclusão e exclusão. Essa perda pode ocorrer por vários motivos:

- Está faltando o mapeamento. Informe os mapeamentos perdidos para o fórum OHDSI[25], para que possam ser adicionados;
- O código fonte está mal definido, por exemplo, CID9CM 799 "Outras causas de morbidade e mortalidade mal definidas e desconhecidas";
- Afirmação negativa, por exemplo ICD9CM V64.0 "Vacinação não realizada";
- Código irrelevante para o paciente, por exemplo, ICD9CM V65 "Outras pessoas que procuram consulta";
- Hierarquias de origem e destino incongruentes.

O último efeito não é incomum para conceitos altamente pré-coordenados (conceitos complexos e refinados que são combinações de diferentes dimensões), a coordenação depende da estrutura da topologia dos vocabulários e para os quais o conceito é mapeado. Se não forem equivalentes, nenhum mapeamento direto poderá ser estabelecido. Mas, para manter a capacidade de recuperar esses conceitos ao pesquisar usando conceitos hierárquicos, dois ou mais mapeamentos são fornecidos em seu lugar. Por exemplo, o ICD10 conceito 45755355 "Diabetes mellitus não dependente de insulina com coma" (código E10.0) não pode mapear diretamente para SNOMED. A Figura 4.5. apresenta o fluxo do mapeamento desse conceito para o seu conceito padrão.

[25] Fórum OHDSI http://forums.ohdsi.org/c/cdm-builders

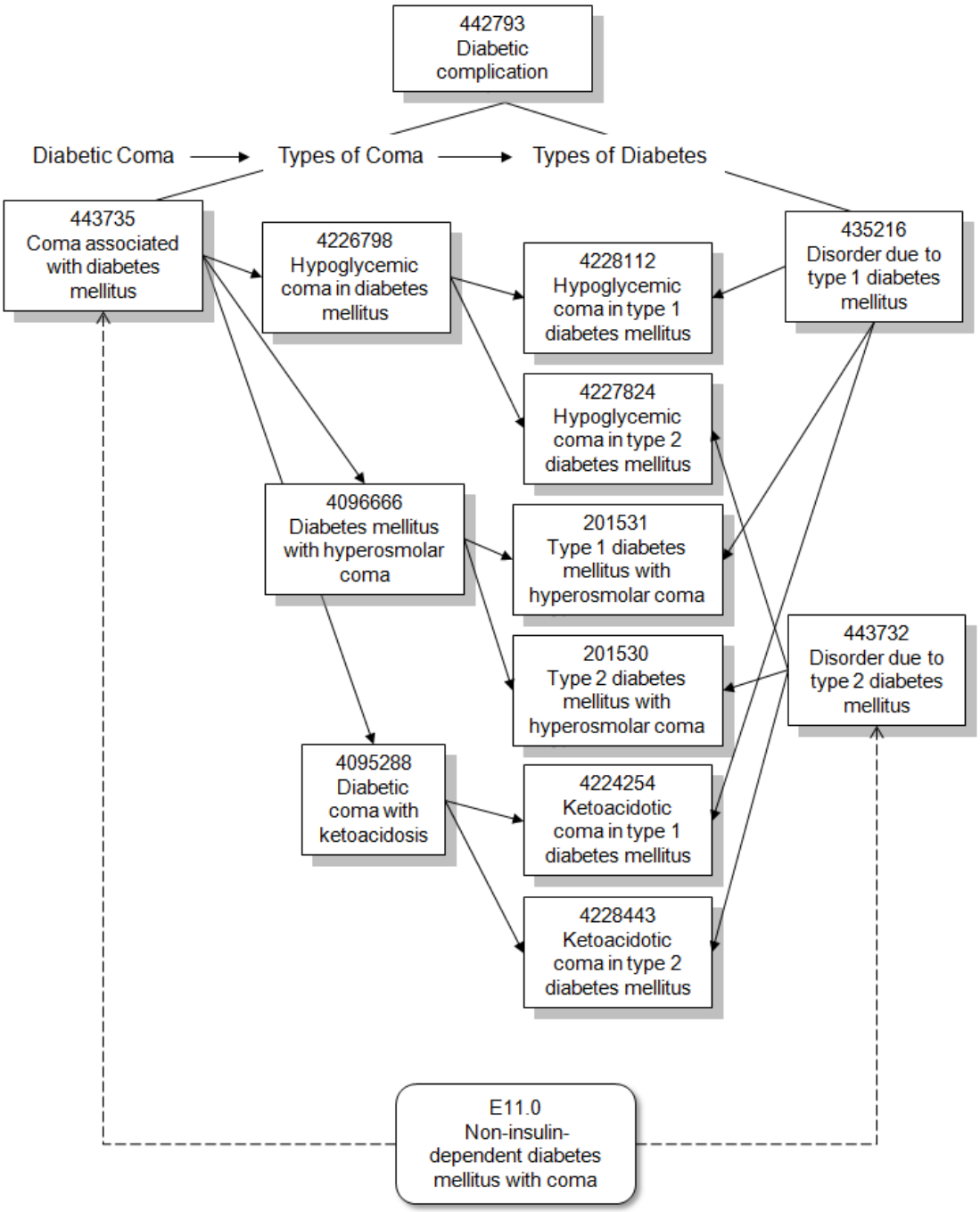


Figura 4.5. Mapeamento de um código ICD10 para um conceito padrão

A hierarquia de destino no SNOMED tem uma topologia diferente para a estrutura organizacional no ICD10. Aqui, diabetes tipo 1 (dependente de insulina, E10) e tipo 2 (independente de insulina, E11) são as principais características distintivas. A combinação de diabetes tipo 2 com a combinação coma E11.0 define diabetes tipo 2 com o coma de complicação. No SNOMED, as complicações do diabetes têm sua própria hierarquia, que se ramifica no conceito de diabetes com coma, ou diabetes tipo 1 ou 2 com complicação. No nível seguinte, distinguem-se as formas distintas de coma

(hipoglicêmico, hiperosmolar e cetoacidótico). No entanto, não há combinação de diabetes tipo 2 com coma. A solução é mapear o E11.0 para o diabetes mellitus tipo 2 44054006 e para o coma associado com diabetes mellitus 420662003. Mesmo que não haja um equivalente distinto no SNOMED, as consultas hierárquicas para o diabetes 2 em combinação com o coma recuperarão os registros corretos.

4.4.2.2. Condições (diagnósticos e achados clínicos)

Condições são registros de uma pessoa sugerindo a presença de uma doença ou condição médica declarada como um diagnóstico, um sinal ou um sintoma, que é observado por um provedor ou relatado pelo paciente. Note que o domínio de condição não faz distinção entre uma mera observação de um sintoma, um diagnóstico totalmente elaborado ou qualquer coisa entre eles. Conceitos de domínio de condição devem ser usados somente nos campos condition_concept_id das tabelas CONDITION_OCCURRENCE e CONDITION_ERA, bem como no campo value_as_concept_id na tabela OBSERVATION (para observações “History of”, “Family history of” etc.).

Os conceitos padrão são retirados do vocabulário SNOMED. Todos os conceitos SNOMED válidos (invalid_reason is null) do domínio de condição são conceitos padrão e, portanto, podem ser usados no campo condition_concept_id nas tabelas CONDITION_OCCURRENCE e CONDITION_ERA.

SNOMED é um vocabulário hierárquico. Portanto conceitos SNOMED também podem ser usados como conceitos de classe: descendentes de qualquer conceito SNOMED na tabela CONCEPT_ANCESTOR podem ser usados como uma correspondência semântica correta em uma consulta.

Regras para o mapeamento de condição: Há muitas condições que não são originárias da função biológica do corpo humano, mas ainda requerem atenção médica. Ou são verdadeiras condições, mas não no momento do levantamento da 'história de' ou na 'história familiar de'. As regras de mapeamento para estas condições são as seguintes:

- Cuidados posteriores após os procedimentos: estes são mapeados para um único conceito 413467001 “Aftercare”, e com um segundo relacionamento “Maps to value” para o procedimento adequado. Por exemplo, ICD10CM Z47.1 “Cuidados posteriores à cirurgia de substituição articular” mapeia para o 413467001 “Aftercare” e para o procedimento 4189532 “Implantation of joint prótese”. Note que “Aftercare” não é uma condição, mas sim do domínio de observação;
- Efeitos tardios ou sequelas de outras condições: se possível, eles são mapeados para conceitos únicos, descrevendo-os como efeitos tardios. Por exemplo, ICD10CM S82.874S concept_id 45589204 “Fratura de pilão não desdentado da tíbia direita, sequela” é mapeada para 197150 “Efeito tardio da fratura das extremidades inferiores”. O domínio é condição;

- História de uma condição: estas são condições, mas não no momento da condição_start_date. Portanto, eles são mapeados para o conceito 4214956 “História de descoberta clínica em questão”, que está no domínio de observação; A condição em si é registrada por meio do relacionamento "Mapear para avaliar". Por exemplo, ICD10CM Z87.820 concept_id “História pessoal de traumatismo cranioencefálico” mapeia para 4214956 “Histórico de achados clínicos em questão” e com “Maps to value” 4132546 “Lesão cerebral traumática”;
- História familiar de uma condição: estes, também, têm um mapeamento duplo para 4167217 "História familiar de descoberta clínica" e um para a condição real através de um relacionamento "Maps to value". Por exemplo, ICD10 / ICD10CM Z80.0 concept_id 45542462 “História familiar de neoplasia maligna de órgãos digestivos” mapeia para 4167217 “Histórico familiar de constatação clínica”, bem como 443568 “Neoplasia maligna do trato gastrointestinal”;
- História do tratamento médico: similarmente, estes são mapeados para o conceito de observação 4207283 “História da terapia medicamentosa”, e através de um link “Maps to value” para a terapia real. Por exemplo, ICD10 Z92.0 concept_id 45605174 “História pessoal de contracepção” vai para 4207283 “História da terapia medicamentosa” história de terapia medicamentosa e 4027509 “Contracepção”. Se o ingrediente exato ou o medicamento clínico / de marca for conhecido, o mapa será direcionado a esses conceitos de medicamentos. Mas isso não é típico;
- Efeito adverso da medicação: se possível, eles são mapeados para um equivalente direto. No entanto, isso geralmente não existe e, em seguida, eles são tratados como a história acima de tratamentos médicos. Por exemplo, tanto ICD10CM T36.8X5A concept_id 45551127 “efeito adverso de outros antibióticos sistêmicos, encontro inicial” e T36.8X5D “efeito adverso de outros antibióticos sistêmicos, encontro posterior” mapear para 437191 “reação adversa a medicamentos antibacterianos”, enquanto CID10CM T36.8X5S concept_id 45560654 “Efeito adverso de outros antibióticos sistêmicos sequela” mapeia para 4207283 “História da terapia medicamentosa”;
- Subdosagem de medicação: essas informações são tratadas de maneira semelhante aos efeitos adversos e os conceitos equivalentes do SNOMED geralmente não existem. Por exemplo, ICD10CM T36.0X6A concept_id 45565479 “Subdosagem de penicilinas, encontro inicial” e T36.0X6D concept_id 45565480 “Subdosagem de penicilinas, encontro subsequente” vá para 40488434 “Dose de medicação muito baixa”, enquanto T36.0X6S concept_id 45565481 “Subdosagem de penicilinas, sequela ”aponta em 4207283“ História da terapia medicamentosa ”;
- Status de ausência do órgão ou transplante / presença de prótese: órgãos ausentes são devidos a um procedimento que os removeu (a menos que sejam condições inatas, que são mapeadas como tal). Portanto, eles são mapeados para

4215685 “Histórico passado de procedimento” e o procedimento apropriado que removeu o órgão por meio de um link “Maps to value”. Por exemplo, ICD10CM Z94.0 concept_id 35225404 “Situação do transplante renal” mapeia para 4215685 “Histórico passado de procedimento” e 4322471 “Transplante de rim”. Z95.5 concept_id 35225418 “Presença de implante de angioplastia coronariana e enxerto” é apontada para 4215685 “História pregressa de procedimento” e 4184832 “Angioplastia coronariana”;

- Conceitos pré-coordenados que listam dois ou mais componentes semânticos através de AND ou OR: esses conceitos são tratados com a seguinte ordem de precedência:
    - Para um conceito de combinação equivalente que também é bem conectado hierarquicamente
    - Para ambos os componentes separadamente
    - Para o ancestral mais comum

Exemplos para estas possibilidades são:

- ICD10 A01 concept_id 45576225 “Febre tifoide e paratifoide” tem uma relação única “Maps to” para 4022808 “Febre tifoide humana E / OU paratifoide”;
- IC10CM F12.22 concept_id 45591098 “Dependência de cannabis com intoxicação” tem dois relacionamentos “Maps to” para 4052690 “intoxicação por cannabis” e 440387 “dependência de cannabis”;
- ICD10 L02.0 concept_id 45596354 “Abscesso cutâneo, furúnculo e carbúnculo da face” tem um único relacionamento “Maps to” de 400082007, “Transtorno da pele da cabeça”;

- Cuidado materno: muitas condições requerem atenção não por causa de uma condição de uma mulher grávida, mas do feto. No entanto, todas essas condições estão sendo mapeadas para a mãe de qualquer maneira. Por exemplo, ICD10 O35.6 concept_id 45567927 “Cuidados maternos para (suspeita) danos ao feto por radiação” tem dois relacionamentos “Maps to” para 199553006 “Feto com dano por radiação” e 289908002 “gravidez”. Ambas as condições são registradas com a mãe;
- Necessidade de imunização: esses conceitos são mapeados para uma observação indicando essa lacuna de imunidade. Um segundo mapeamento com o relationship_id “Maps to value” é então direcionado para a condição (representada como um conceito SNOMED) contra a qual a imunização é inoculada. Observe que ele não está mapeado para a vacina em si (que seria representado como um conceito de RxNorm). Por exemplo, ICD10 Z23 concept_id 45556822 “Necessidade de imunização contra doenças bacterianas únicas” é mapeado para 170536002 “Vacinação necessária” e mapeia para o valor 87628006 “Doença infecciosa bacteriana”;
- Condições que indicam níveis anormais de um teste: estes são divididos em conceitos de medição e resultado. Por exemplo, ICD10 R77.1 concept_id

45553745 “Anormalidade da globulina” tem uma relação de “Maps to” com a medição “Globulina” da medição 4353510 e uma relação “Maps to value” para 4135493 “Anormal”.

4.4.2.3. Medicamentos/drogas

Os conceitos de domínio de exposição a medicamentos capturam registros sobre a utilização de um medicamento quando ingeridos ou introduzidos de uma forma no corpo humano. Uma droga é uma substância bioquímica formulada de tal maneira que, quando administrada a uma pessoa, ela exerce certo efeito fisiológico ou bioquímico.

Os seguintes produtos não são considerados drogas, mas dispositivos:

- Radiofármacos
- Material de contraste para geração de imagens
- Produtos nutricionais e suplementos, incluindo fórmulas infantis. Na realidade, isso resulta na situação ligeiramente arbitrária e, em alguns casos, difícil de verificar que soluções de sais para uso parental são Drogas (hidratar pacientes e manter o equilíbrio iônico), enquanto a adição de nutrientes como glicose ou vitaminas os torna dispositivos (alimentação de pacientes)
- Produtos diretamente derivados do sangue (por exemplo, eritrócitos ou plasma)

Os conceitos do domínio medicamentos devem ser usados no drug_concept_id das tabelas DRUG_EXPOSURE, DRUG_ERA e DOSE_ERA (ambos apenas no nível do ingrediente) ou no campo value_as_concept_id da tabela OBSERVATION ou MEASUREMENT (por exemplo, para medições como "Nível Plasma").

4.4.2.4. Medidas (valores quantitativos)

A tabela MEASUREMENT contém registros de medição, ou seja, valores estruturados (numéricos ou categóricos) obtidos por meio de exame sistemático e padronizado ou teste de uma amostra de uma pessoa. Contém tanto pedidos quanto resultados de tais medidas, como testes de laboratório, sinais vitais, resultados quantitativos de relatórios de patologia, etc. A tabela MEASUREMENT requer algumas convenções, descritas a seguir:

- As medições diferem das observações, na medida em que exigem um teste padronizado ou alguma outra atividade para gerar um resultado quantitativo ou qualitativo. Por exemplo, LOINC 1755-8 concept_id 3027035 'Albumina [Massa / tempo] em urina de 24 horas' é o teste de laboratório para medir um determinado produto químico em uma amostra de urina;
- Mesmo que cada medida tenha sempre um resultado, os campos value_as_number e value_as_concept_id não são obrigatórios. Quando o resultado não é conhecido, o registro de medição representa apenas o fato de que a medição correspondente foi realizada, o que em si já é uma informação útil para alguns casos de uso;

- Conceitos válidos de medição (measurement_concept_id) pertencem ao domínio 'Measurement', mas podem se sobrepor ao domínio 'Observation'. Isso se deve ao fato de que existe uma sobreposição entre o exame ou teste sistemático (medição) e uma simples determinação de fato (observação). Quando o valor da fonte de medição do código não pode ser convertido em um ID de conceito de medição padrão, uma entrada de medição é armazenada com apenas o source_concept_id e measurement_source_value correspondente e um measurement_concept_id de 0;
- As medições são armazenadas como pares de valores de atributo, com o atributo como o conceito de medição e o valor representando o resultado. O valor pode ser um conceito (armazenado em value_as_concept) ou um valor numérico (value_as_number) com uma unidade (unit_concept_id);
- Conceitos válidos para o campo value_as_concept pertencem ao domínio 'Meas Value';
- Para alguns conceitos de medição, o resultado é incluído no teste. Por exemplo, ICD10 concept_id 45595451 “Presença de álcool no sangue, nível não especificado” indica uma medição e o resultado (presente). Nessas situações, a tabela CONCEPT_RELATIONSHIP além do registro “Maps to” contém um segundo registro com o relationship_id definido como “Maps to value”. Neste exemplo, o relacionamento "Maps to" direciona para 4041715 "Medição de etanol de sangue", bem como um registro "Maps to value" para 4181412 "Presente";
- O operator_concept_id é fornecido opcionalmente para medições relativas, em que o valor preciso não está disponível, mas sua relação com um determinado valor está. Por exemplo, isso pode ser usado para limites mínimos de detecção de um teste;
- O significado do conceito 4172703 para '=' é idêntico à omissão de um valor operator_concept_id. Como o uso desse campo é raro, é importante, ao elaborar análises, não esquecer o teste do conteúdo desse campo para valores diferentes de =;
- Os conceitos válidos para o campo operator_concept_id pertencem ao domínio 'Meas Value Operator';
- A unidade é opcional, mesmo que seja fornecido um value_as_number;
- Se os intervalos de referência para o limite superior e inferior do normal, conforme previsto (normalmente por um laboratório), estes são armazenados nos campos range_high e range_low. Os intervalos têm a mesma unidade que o value_as_number;
- A visita durante a qual a observação foi feita é registrada através de uma referência à tabela VISIT_OCCURRENCE. Esta informação nem sempre está disponível;
- O provedor que faz a observação é registrado por meio de uma referência à tabela PROVIDER. Esta informação nem sempre está disponível.

4.4.3. Vocabulários Locais

Existem três abordagens para manipular códigos fonte que não estão no vocabulário OMOP (em ordem de complexidade):

1. Utilizando o SOURCE_TO_CONCEPT_MAP: No vocabulário OMOP existe uma tabela vazia chamada SOURCE_TO_CONCEPT_MAP. É uma estrutura de tabela simples que permite estabelecer mapeamento(s) para cada código-fonte com um conceito padrão no vocabulário OMOP (TARGET_CONCEPT_ID). Esse trabalho pode ser facilitado pela ferramenta USAGI da OHDSI, que verifica a semelhança de texto entre as descrições do código-fonte e o vocabulário OMOP e mapeia os resultados em uma estrutura de tabela SOURCE_TO_CONCEPT_MAP. Exemplos de arquivos SOURCE_TO_CONCEPT_MAP podem ser encontrados em [26]. Esses arquivos SOURCE_TO_CONCEPT_MAP gerados são carregados no SOURCE_TO_CONCEPT_MAP vazio do vocabulário do OMOP antes de incorporar os dados nativos ao CDM, para que o processo de ETL do CDM possa utilizá-los.

2. Adicionando CONCEPT.CONCEPT_IDs: Quando um código-fonte não é suportado pelo vocabulário OMOP, pode-se criar novos registros na tabela CONCEPT, porém os CONCEPT_IDs devem iniciar > 2000000000 para que seja fácil distinguir entre os conceitos de vocabulário OMOP e os conceitos específicos do site. Quando esses conceitos existirem CONCEPT_RELATIONSHIPS podem ser gerados para atribuí-los a terminologias padrão, o USAGI também pode facilitar esse processo.

3. Trabalhe com o *ODYSSEUS Data Services*[27] para adicionar ao Vocabulário OMOP. O vocabulário OMOP está em evolução e novos vocabulários podem ser adicionados.

4.5. Fenotipagem: Definição de uma coorte

Fenótipo é o termo criado pelo pesquisador dinamarquês Wilhelm L. Johannsen (1857 – 1912) e representa as características (parâmetros) que definem um indivíduo, sejam elas morfológicas, fisiológicas ou comportamentais.

A fenotipagem é o processo de identificação de pacientes com uma condição ou característica médica por meio de uma consulta de pesquisa a um sistema RES ou repositório de dados clínicos usando um conjunto definido de elementos de dados e expressões lógicas. O objetivo da fenotipagem é construir coortes identificando pacientes com uma condição médica particular, por exemplo, pacientes com Diabetes Mellitus Tipo 2 (DM2) ou aqueles que sofreram um Infarto do Miocárdio (IM).

Fenotipagem é a seleção dos valores de um conjunto de parâmetros que definem a classificação dos indivíduos participantes de uma coorte. Podemos agrupar o conjunto

[26] https://github.com/OHDSI/ETL-CDMBuilder/tree/master/man/VOCABULARY_ADDITIONS
[27] *ODYSSEUS Data Services* https://odysseusinc.com/

de parâmetros em: eventos de entrada na coorte, critérios de inclusão/exclusão e parâmetros de saída da coorte.

Nesta seção abordamos a definição dos parâmetros que compõem o processo de formação de uma coorte, junto com uma visualização gráfica dos principais elementos da coorte no tempo.

4.5.1. Tabela COHORT_DEFINITION

A tabela COHORT_DEFINITION contém registros definindo uma coorte derivada dos dados através da descrição e sintaxe associadas e mediante instanciação (execução do algoritmo) inserida na tabela COHORT. Coortes são um conjunto de assuntos que satisfazem uma determinada combinação de critérios de inclusão por um período de tempo. A tabela COHORT_DEFINITION fornece uma estrutura padronizada para manter as regras que governam a inclusão de um assunto em uma coorte, e pode armazenar código de programação operacional para instanciar a coorte dentro do CDM.
A tabela requer algumas convenções, descritas a seguir:

- O cohort_definition_syntax não prescreve nenhuma sintaxe específica ou linguagem de programação. Normalmente, seria qualquer SQL, uma linguagem de definição de coorte ou uma descrição de texto livre do algoritmo;
- O subject_concept_id determina em que consistem os sujeitos ou entidades individuais da coorte. Na maioria dos casos, isso seria uma pessoa (paciente). Mas coortes também poderiam ser construídas para provedores, visitas ou qualquer outro domínio. Observe que o domínio não é codificado usando o domain_id alfanumérico como na tabela CONCEPT. Em vez disso, o conceito correspondente é usado. Os conceitos para cada domínio podem ser obtidos na tabela DOMAIN no domínio_concept_id.

4.5.2. A tabela COHORT_ATTRIBUTE

A tabela COHORT_ATTRIBUTE contém atributos associados a cada assunto dentro de uma coorte, conforme definido por um determinado conjunto de critérios por um período de tempo. A definição do atributo de coorte está contida na tabela ATTRIBUTE_DEFINITION. Requer algumas convenções, descritas a seguir:

- Cada registro na tabela COHORT_ATTRIBUTE está vinculado a um registro específico na tabela COHORT, identificado pelos campos correspondentes cohort_definition_id, subject_id, cohort_start_date e cohort_end_date;
- Acrescenta aos registros da coorte co-variáveis calculadas (por exemplo idade, IMC) ou escalas compostas (por exemplo, índice de Charleson);
- A definição unificada ou recurso do atributo de coorte é capturado no atributo_de_definição_id referindo-se a um registro na tabela ATTRIBUTE_DEFINITION;

- O resultado ou valor real do atributo *cohort* (co-variável, valor de índice) é capturado nos campos value_as_number (se o valor for numérico) ou value_as_concept_id (se o valor for um conceito).

4.6. Tipos de análises

Dados observacionais têm potencial para responder a uma miríade de questões importantes na área da saúde:

- Como podemos identificar novos alvos terapêuticos de forma rápida e eficaz?
- Podemos medir o impacto relativo de diferentes intervenções de tratamento?
- Como podemos prever pacientes com um perfil de alto risco para certas doenças antes que apresentem sintomas?
- Como podemos prevenir melhor às condições crônicas?
- Quais são os melhores padrões de cuidado para gerenciar pacientes, especialmente com diferentes combinações de comorbidade?
- Como podemos melhorar o desenho dos estudos clínicos focando nos pacientes com o melhor recrutamento para efetuar o perfil de tamanho?
- Como podemos otimizar a adesão às diretrizes de tratamento e quais fatores influenciam o comportamento dos pacientes?

Cada uma destas questões define uma análise diferente:

- Perfis: linha do tempo que mostra o histórico de um paciente.
    - Quais são todos os eventos associados a um paciente específico ao longo do tempo?
- Estimativas:
    - Qual é o efeito do tratamento A no desfecho X?
- Predições:
    - Em uma população em risco, quais pacientes terão um determinado desfecho?
- Taxas de incidência: proporção e taxa das contagens brutas de pacientes, casos e tempo de risco para determinado evento.
    - Quantos novos desfechos são esperados por um intervalo de tempo?
- Caracterização de populações: geração de estatísticas descritivas da coorte a partir de dados de covariáveis de nível de pessoa.
    - Como podemos melhorar o desenho dos estudos clínicos focando nos pacientes com o melhor recrutamento para efetuar o perfil de tamanho?

4.6.1 Ferramentas de análises

Nesse tópico pretendemos oferecer uma abordagem expositiva da análise de dados de saúde através do conjunto de ferramentas OHDSI. Estas ferramentas auxiliam a

elaboração e análise dos diferentes tipos de estudo, facilitam a exploração dos dados e a geração de evidências. Entre elas podemos citar:

ACHILLES[28] (Caracterização Automatizada de Informações de Saúde em Grande Escala do Sistema de Exploração Longitudinal), é uma ferramenta de visualização baseado em estatísticas resumidas pré-extraídas de conjuntos de dados no formato CDM. Permite a caracterização, avaliação da qualidade e visualização de dados observacionais e fornece aos usuários uma estrutura exploratória e interativa para avaliar a demografia dos pacientes e a prevalência de todas as condições, medicamentos, procedimentos e observações armazenados no conjunto de dados. Possibilita avaliar a qualidade do banco de dados, procurando lacunas que podem significar erros de upload, fazer uma avaliação inicial se o banco de dados conterá um número suficiente de casos de interesse que valham a pena investigar mais. Exibe a prevalência da condição, quantidade de pacientes, distribuição etária, gênero e o tempo de observação. Podemos visualizar os dados a partir da seleção do banco de dados e seleção dos relatórios, que podemos enumerar a seguir:

- *Dashboard:* painel com um sumário da base de dados em análise, população/gênero, idade na primeira observação, pessoas com observações contínuas/mês;
- *Achilles Heel*: mensagens da qualidade dos dados do dados em análise;
- Person: estatística descritiva da população por ano de nascimento, gênero, raça, etnia;
- *Observation Periods*: idade na primeira observação, idade/gênero, tempo de observação, observações contínuas/ano e mês, período/pessoa;
- *Data Density*: total de registros (era de condição, ocorrência da condição, era de drogas, exposição a drogas, período de observação, ocorrência de procedimentos e de visitas) por ano, por pessoa/ano, por conceito/tipo;
- *Conditions/conditions era:* prevalência das condições por número de pessoas e registro por pessoas e tempo (mapa e tabela);
- *Observations*: prevalência das observações por número de pessoas e registro por pessoas (mapa e tabela);
- *Drug eras*: prevalência das drogas por número de pessoas e tempo (mapa e tabela);
- *Drug Exposures*: prevalência de exposição às drogas por número de pessoas e registro por pessoas (mapa e tabela);
- Procedures: prevalência de procedimentos por número de pessoas e registro por pessoas (mapa e tabela);
- *Visits*: prevalência por tipo de visita (internação, consulta, emergência) por número de pessoas e registro por pessoas (mapa e tabela);
- *Death*: prevalência de óbito por idade/gênero/ano, por mês, tipo;

[28] ACHILLES http://www.ohdsi.org/web/achilles

ATHENA[29] Camada intermediária do aplicativo da web para distribuição e navegação nos vocabulários padronizados para o CDM. Site de download dos vocabulários padrão. A ferramenta permite a visualização dos vocabulários, a partir da seleção do domínio (dados demográficos, condição, droga, procedimento, medição, observação, visita, óbito, etc.), os conceitos (padrão, classificação ou não-padrão), a classe (lista das classificações usadas para diferenciar conceitos dentro de um determinado vocabulário. Exemplos: achado clínico, ingredientes, procedimentos, etc.) e o vocabulário (ICD9, ICD10, SNOMED, etc.). Realiza o download dos vocabulários selecionados para serem importados em seu ambiente de construção do CDM. A ferramenta também permite explorar o vocabulário antes de baixá-lo, apresenta os mapeamentos ou um código específico e com qual conceito padrão está associado.

Todos os mapeamentos estão disponíveis na tabela Concept_relationship (que pode ser baixada do ATHENA). Cada valor em uma terminologia de fonte suportada recebe um Concept_id (que é considerado não padrão). Cada Source_concept_id terá um mapeamento para um Standard_concept_id. Por exemplo:

Neste caso, o conceito SNOMED padrão 201826 para diabetes mellitus tipo 2 seria armazenado na tabela Condition_occurrence, pois Condition_concept_id e o conceito ICD10CM 1567956, para diabetes mellitus tipo 2, seriam armazenados como Condition_source_concept_id.

ATLAS[30] é uma das ferramenta publicamente disponível para pesquisadores conduzir análises científicas em dados observacionais padronizados convertidos para o OMOP CDM V5[31]. Permite criar coortes definindo grupos de pessoas com base em uma exposição a um medicamento ou diagnóstico de uma condição específica usando dados de registros de assistência médica. Os perfis dos pacientes podem ser visualizados dentro de uma coorte específica e análises de estimativa de nível populacional permitem a comparação de duas coortes diferentes. Apresenta as seguintes funcionalidades:

- Fontes de dados: opção de seleção da base de dados a ser analisada. Disponibiliza os gráficos de análise da ferramenta Achilles;
- Vocabulário: seleção e importação dos vocabulários necessários para análise dos dados;
- Conjuntos de conceitos: permite definir um novo conceito e listar/exportar;
- Definição de coortes: preparação das coortes para estudo. Define o grupo de pessoas que satisfazem um ou mais critérios de inclusão por um período de tempo. Como consequência desta definição:
    - uma pessoa pode pertencer a múltiplas coortes,
    - pode pertencer a mesma coorte em diferentes períodos de tempo,
    - uma pessoa não pode pertencer mais de uma vez à mesma coorte durante o mesmo intervalo de tempo,

[29] ATHENA http://athena.ohdsi.org
[30] ATLAS http://www.ohdsi.org/web/atlas/#/home
[31] OMOP CDM v5 http://www.ohdsi.org/web/wiki/doku.php?id=documentation:cdm:single-page

    - uma coorte pode ter zero ou mais membros;
- Caracterização de coortes: é definida como o processo de geração de estatísticas descritivas da coorte a partir de dados de covariáveis a nível de pessoa. As estatísticas resumidas dessas covariáveis podem ser contagem, média, sd, var, min, max, mediana, intervalo e quantis. Além disso, as covariáveis durante um período podem ser estratificadas em unidades de tempo para análises de séries temporais, como intervalos fixos de tempo relativos a data de início da coorte (por exemplo, a cada 7 dias, a cada 30 dias, etc.) ou em intervalos absolutos do calendário, como semana, mês, trimestre, ano;
- Caminho da coorte: é definido como o processo de gerar uma sequência agregada de transições entre as coortes de eventos e as pessoas nas coortes alvo;
    - Coortes Alvo: cada uma das coortes-alvo serão analisadas em relação às coortes de eventos;
    - Coortes de Eventos: cada coorte de eventos define o passo em um caminho que pode ocorrer para uma pessoa na coorte de tratamento;
- Taxas de incidência: as taxas de incidência podem ser geradas incluindo as coortes de meta e resultados. A taxa de incidência é reportada como uma proporção e uma taxa. São fornecidas as contagens brutas de pessoas, casos e tempo em risco;
    - Tempo em risco: define a janela de tempo relativa à data de início ou término da coorte com um deslocamento para considerar a pessoa 'em risco' do desfecho em análise;
    - Critérios de estratificação: fornecer critérios de estratificação opcionais para a análise que dividirá a população em grupos únicos em relação aos critérios definidos;
- Perfis: o Atlas fornece a capacidade de pesquisar e explorar perfis individuais de pacientes em um banco de dados. Essa funcionalidade pode ser acessada clicando no item de menu de perfis, selecionando o banco de dados de interesse e inserindo um número de identificação do paciente. Dentro do perfil apresentado, o menu à esquerda lista os registros individuais que podem ser condições, medicamentos, procedimentos, etc. A tabela no canto inferior direito lista os domínios individuais, IDs de conceitos, nomes de conceitos e dias de início e fim no registro de um determinado paciente. O gráfico pode ser redimensionado para aumentar o zoom em uma determinada janela de tempo, por exemplo, nos primeiros 100 dias. Alterando a janela de tempo de interesse, as tabelas a esquerda e inferior direita mudam para refletir a janela de tempo de interesse;
- Estimativa: o Atlas tem a capacidade de realizar estudos de estimativa usando o design de coorte comparativo. O procedimento de estimativa no Atlas usa a

metodologia de escore de propensão[32]. Existem 3 principais modelos de resultados: regressão logística; regressão de Poisson; e riscos proporcionais de Cox;

- Predição: o Atlas incorporou a capacidade de gerar modelos de predição usando métodos de aprendizado de máquina para medicina de precisão e interceptação de doenças, incluindo:
    - *Regularized regression*
    - *Random forest*
    - *k-nearest neighbors*

Usa um conjunto de covariáveis, incluindo, por exemplo, todas as drogas, diagnósticos, procedimentos, bem como idade, índices de comorbidade, etc.
Os modelos de resultados suportados são logísticos, Poisson e sobrevivência (tempo até o evento).

4.7. Considerações finais e conclusões

A iniciativa OHDSI surge como resposta a necessidade de aproveitar o enorme volume de informações disponibilizados pelos sistemas de saúde informatizados e da percepção da carência de integrar estas informações de forma confiável e transparente para poder ter uma pesquisa de qualidade e reprodutível.

A força estatística que traz este volume de dados se contrapõe com a dificuldade de integrar informações que utilizam vocabulários diferentes ou não padronizados para poder ter estudos comparativos realmente efetivos. Isso fomenta a discussão a respeito da importância de passar a utilizar vocabulários padronizados para todas as atividades suscetíveis de informatização a risco de perder a possibilidade de fazer estudos com relevância internacional que permitam o acesso a publicações de alto impacto.

No Brasil, as mais importantes iniciativas estão hoje orientadas a estabelecer vocabulários padronizados. Ao ver o impacto que isto têm na elaboração de uma ferramenta de pesquisa, podemos apreciar a enorme importância deste esforço.

Não precisamos reinventar a pesquisa observacional, as ferramentas já estão disponíveis, em código aberto, de forma gratuita, simples de instalar em ambientes fechados ou prontas na nuvem. Precisamos sim, estudá-las e integrar as bases disponíveis ao modelo OMOP, passando a participar dos centros de pesquisa do mundo, somando esforços com a comunidade mundial, melhorando o que já existe, divulgando o conhecimento e aproveitando toda esta infraestrutura para o ensino da epidemiologia aplicada no mundo real.

[32] Um escore de propensão é a probabilidade de uma unidade (por exemplo, pessoa) ser designada para um tratamento particular, dado um conjunto de covariáveis observadas. Os escores de propensão são usados para reduzir o viés de seleção ao equacionar grupos com base nessas covariáveis.

4.8. Glossário de Termos

Termo - Descrição

Ancestor - O conceito de nível superior em um relacionamento hierárquico. Note que os ancestrais e descendentes podem estar muitos níveis separados uns dos outros.

Ambulatory Payment Classification (APC) - O APC é usado como um método de pagamento de serviços ambulatoriais para o programa *Medicare*, que é análogo aos DRGs para serviços de internação.

Average Wholesale Price (AWP) - Os fabricantes de preços estabelecidos para medicamentos controlados devem ser comprados no atacado para as farmácias e prestadores de serviços de saúde.

Anatomical Therapeutic Chemical (ATC) - é a sigla para a classificação Anatômica Terapêutico Química, que, em conjunto com a Dose Diária Definida - DDD (*Defined Daily Dose*), forma o sistema ATC/DDD, que, desde de 1996, passou a ser reconhecido pela Organização Mundial de Saúde como padrão internacional para os estudos de utilização de drogas. No sistema de classificação ATC, as drogas são divididas em diferentes grupos, de acordo com o órgão ou sistema no qual eles atuam e suas propriedades químicas, farmacológicas e terapêuticas. As drogas são divididas em cinco níveis diferentes, sendo o primeiro dividido em quatorze grupos principais, com um subgrupo farmacológico/terapêutico (segundo nível). Os terceiro e quarto níveis correspondem a subgrupos químicos/farmacológicos/terapêuticos, e o quinto nível, à substância química.

Centers for Disease Control and Prevention (CDC) - Os Centros de Controle e Prevenção de Doenças é uma agência federal dos Estados Unidos sob o Departamento de Saúde e Serviços Humanos. Ele trabalha para proteger a saúde pública e a segurança, fornecendo informações para melhorar as decisões de saúde.

Common Data Model (CDM) - O CDM pretende facilitar a análise observacional de diferentes bases de dados de saúde. O CDM define estruturas de tabela para cada uma das entidades de dados (por exemplo, Pessoas, Ocorrência de Visita, Exposição a Medicamentos, Ocorrência de Condição, Observação, Ocorrência de Procedimentos, etc.). Inclui elementos de dados observacionais que são relevantes para identificar a exposição a vários tratamentos e definir a ocorrência da condição. O CDM inclui os vocabulários padronizados de termos e as tabelas de domínio da entidade.

Concept - Um conceito é a unidade básica de informação. Os conceitos podem ser agrupados em um determinado domínio. Um conceito é um termo único que possui um identificador / nome único e estático, pertence a um domínio e pode existir em relação a

outros conceitos. Os relacionamentos verticais consistem em instruções "é um" que formam uma hierarquia lógica. Em geral, os conceitos acima de um determinado conceito são referidos como ancestrais e os abaixo como descendentes.

Conceptual Data Model - Um modelo de dados conceituais é um mapa de conceitos e seus relacionamentos. Isso descreve a semântica de uma organização e representa uma série de afirmações sobre sua natureza. Especificamente, descreve as coisas importantes para uma organização (classes de entidade), sobre as quais ela está inclinada a coletar informações e características de (atributos) e associações entre pares dessas coisas de significância (relacionamentos).

Current Procedural Terminology, 4th edition (CPT-4) - Uma terminologia que é mantida pela *American Medical Association* (AMA). Ele é usado por hospitais para pacientes ambulatoriais do *Medicare* e por médicos para serviços ambulatoriais.

Data mapping - São os mapeamentos de elementos de dados entre dois modelos de dados, terminologias ou conceitos distintos. O mapeamento de dados é o processo de criação de mapeamentos de elementos de dados entre dois modelos de dados distintos. O mapeamento de dados é usado como primeiro passo para uma ampla variedade de tarefas de integração de dados.

Demographics - A demografia refere-se a características selecionadas de pessoas. Os dados demográficos podem incluir dados como raça, idade, sexo, data de nascimento, local etc.

Descendant - O conceito de nível inferior em um relacionamento hierárquico. Note que os ancestrais e descendentes podem estar muitos níveis separados uns dos outros.

Design Principle - Um arranjo organizado de um ou mais elementos ou princípios para um propósito. Ele identifica os principais princípios e as melhores práticas para ajudar os desenvolvedores a produzir software. Entender completamente as metas das partes interessadas e projetar sistemas com essas metas em mente são as melhores abordagens para entregar resultados com êxito.

Diagnosis- Related Group (DRG) - Os DRG são usados como um método de pagamento de serviços de internação para o programa *Medicare*, que é análogo às APCs para serviços ambulatoriais.

Electronic Health Record (EHR) - Registro eletrônico de saúde refere-se ao prontuário de uma pessoa individual em formato digital. Pode ser composto de registros médicos eletrônicos de vários locais e / ou fontes. O EHR é um registro eletrônico longitudinal de informações de saúde de pessoas geradas por um ou mais encontros em qualquer

ambiente de prestação de cuidados. Incluídos nestas informações estão a demografia pessoal, notas de progresso, problemas, medicamentos, sinais vitais, histórico médico, imunizações, dados laboratoriais e relatórios de radiologia.

Electronic Medical Record (EMR) - Um prontuário eletrônico é um registro médico computadorizado criado em uma organização que presta atendimento, como um hospital ou ambulatório. Registros médicos eletrônicos tendem a fazer parte de um sistema de informações de saúde local independente que permite o armazenamento, a recuperação e a manipulação de registros. Este documento fará referência ao EHR, mesmo que uma fonte de dados específica possa usar internamente a definição do EMR.

Extract Transform Load (ETL) - Processo de obtenção de dados de um armazenamento de dados (*Extract*), modificando-o (*Transform*) e inserindo-o em um armazenamento de dados diferente (*Load*).

Generic Product Identifier (GPI) - Um identificador exclusivo patenteado para um medicamento usado pelo banco de dados de formulários comerciais Medi-Span.

Healthcare Common Procedure Coding System (HCPCS) - Os códigos de nível I do HCPCS são gerenciados pela AMA (*American Medical Association*). Os códigos de nível II do HCPCS são gerenciados pelo CMS (*Centers for Medicare & Medicaid Services*). Os códigos de Nível II incluem: procedimento alfanumérico do HCPCS e códigos modificadores, suas descrições e dados administrativos, de cobertura e de preços aplicáveis do *Medicare*. Esses códigos são usados para serviços ambulatoriais do *Medicare*.

Health Insurance Portability and Accountability Act (HIPAA) - Uma lei federal que foi concebida para permitir a portabilidade do seguro de saúde entre empregos. Além disso, exigiu a criação de uma lei federal para proteger informações de saúde pessoalmente identificáveis; se isso não ocorresse em uma data específica (o que não acontecia), a HIPAA orientou o *Department of Health and Human Services* (DHHS) a emitir regulamentações federais com o mesmo objetivo. O DHHS emitiu os regulamentos de privacidade do HIPAA (a Regra de Privacidade do HIPAA), bem como outros regulamentos no âmbito do HIPAA.

Health Level Seven (HL7) - A HL7 é uma organização global sem fins lucrativos dedicada ao fornecimento de uma estrutura abrangente e padrões relacionados para o intercâmbio, integração, compartilhamento e recuperação de informações eletrônicas de saúde que dão suporte à prática clínica e ao gerenciamento, entrega e avaliação. dos serviços de saúde. As especificações do HL7 baseiam-se principalmente em códigos e vocabulários de uma variedade de fontes.

Instituto do Coração do Hospital das Clínicas da Faculdade de Medicina da Universidade de São Paulo (InCor-HCFMUSP) - O InCor é um hospital público universitário de alta complexidade, especializado em cardiologia, pneumologia e cirurgias cardíaca e torácica. Além de ser um polo de atendimento - desde a prevenção até o tratamento -, o Instituto do Coração também se destaca como um grande centro de pesquisa e ensino. O InCor é parte do Hospital das Clínicas e campo de ensino e de pesquisa para a Faculdade de Medicina da USP.

International Classification of Disease, 9th Revision, Clinical Modifications (ICD-9-CM Or ICD-9) - O sistema oficial de atribuição de códigos para diagnósticos e procedimentos associados à utilização hospitalar nos Estados Unidos.

Logical Data Model - Modelos de dados lógicos são representações gráficas dos requisitos de negócios. Eles descrevem as coisas importantes para uma organização e como elas se relacionam umas com as outras, além de definições e exemplos de negócios. O modelo de dados lógicos pode ser validado e aprovado por um representante comercial e pode ser a base do design do banco de dados físico.

Logical Observation Identifiers Names and Codes (LOINC) - Nomes e identificadores universais de códigos para a terminologia médica relacionada ao Registro de Saúde Eletrônico e auxilia na troca eletrônica e coleta de resultados clínicos (como testes de laboratório, observações clínicas, gerenciamento de resultados e pesquisa).

Medical Dictionary for Regulatory Activities (MedDRA) - MedDRA é uma terminologia médica internacional clinicamente validada, usada pelas autoridades reguladoras e pela indústria biofarmacêutica regulamentada. A terminologia é usada durante todo o processo de regulamentação, desde a pré-comercialização até o pós-marketing, e para entrada, recuperação, avaliação e apresentação de dados.

National Drug Codes (NDC) - Identificadores exclusivos atribuídos a medicamentos individuais. Os NDCs são usados principalmente como um código de inventário e para prescrições.

National Drug File - Reference Terminology (NDF-RT) - Uma terminologia de referência de medicamento não-proprietário que inclui o conhecimento de drogas e classifica as drogas, principalmente por mecanismo de ação e efeito fisiológico.

Primary Care Provider (PCP) - Um prestador de cuidados de saúde designado como responsável pela prestação de cuidados médicos gerais a um doente, incluindo avaliação e tratamento, bem como o encaminhamento para especialistas.

Protected Health Information (PHI) - Informações de saúde protegidas sob HIPAA incluem qualquer informação de saúde identificável individualmente. Identificável refere-se não apenas aos dados explicitamente vinculados a um indivíduo em particular (essa é a informação identificada). Ele também inclui informações de saúde com itens de dados que poderiam ser esperados para permitir a identificação individual. A informação não identificada é aquela a partir da qual todas as informações potencialmente identificáveis foram removidas.

Read Codes - Os *Read Codes* foram desenvolvidos pelo Dr. James Read. Eles contém centenas de milhares de termos, sinônimos e abreviações abrangendo todos os aspectos do atendimento ao paciente, incluindo sinais e sintomas, tratamentos e terapias, investigações, ocupações, diagnósticos, medicamentos e dispositivos médicos. O U.K. *National Health Service Centre for Coding and Classification* (NHS CCC) adquiriu e mantém atualmente os *Read Codes*, agora conhecidos como *Clinical Terms*.

RxNorm - Uma nomenclatura padronizada para drogas clínicas e dispositivos de dispensa de medicamentos, produzida pela *National Library of Medicine*. Em RxNorm, o nome de um medicamento clínico descreve seus ingredientes, pontos fortes e / ou forma de apresentação. O RxNorm fornece nomes normalizados para medicamentos clínicos e vincula seus nomes a muitos dos vocabulários de drogas comumente usados no gerenciamento de farmácia e no software de interação medicamentosa, incluindo os do *First Data Bank, Micromedix, MediSpan, Gold Standard Alchemy e Multum*. Ao fornecer links entre esses vocabulários, o RxNorm pode mediar mensagens entre sistemas que não usam o mesmo software e vocabulário.

SNOMED CT - *Systematized Nomenclature of Medicine - Clinical Terms*: Nomenclatura Sistematizada de Medicina - Termos Clínicos. A versão 3 dos *Clinical Terms* (CTV3) (*Read Codes*) foi incorporada na *Systematized Nomenclature of Medicine - Reference Terminology* (SNOMED RT), resultando na criação do SNOMED - *Clinical Terms* (SNOMED CT). O SNOMED-CT é um de um conjunto de padrões designados para uso em sistemas do Governo Federal dos EUA para o intercâmbio eletrônico de informações clínicas de saúde e também é um padrão exigido em especificações de interoperabilidade do *US Healthcare Information Technology Standards Panel*. Esta terminologia está sendo implementada internacionalmente como padrão dentro de outros países Membros do IHTSDO - *International Health Terminology Standards Development Organisation*.

Referências consultadas

Abrahão, Maria Tereza Fernandes. Método de extração de coortes em bases de dados assistenciais para estudos da doença cardiovascular [tese]. São Paulo:, Faculdade de Medicina; 2016 [citado 2019-04-22] doi:10.11606/T.5.2016.tde-04082016-160129.

Abrahão MTF, Nobre MRC, Gutierrez MA. A method for cohort selection of cardiovascular disease records from an electronic health record system. International Journal of Medical Informatics [Internet] Volume 102, June 2017, Pages 138-149 https://doi.org/10.1016/j.ijmedinf.2017.03.015

Banda JM, Halpern Y, Sontag D, Shah NH. Electronic phenotyping with APHRODITE and the Observational Health Sciences and Informatics (OHDSI) data network. AMIA Jt Summits Transl Sci Proc. 2017, 48-57. Published 2017 Jul 26 Overhage, J. Marc & B Ryan, Patrick & G Reich, Christian & Hartzema, Abraham & E Stang, Paul. (2011). Validation of a common data model for active safety surveillance research. Journal of the American Medical Informatics Association : JAMIA. 19. 54-60. 10.1136/amiajnl-2011-000376.

Ferramenta estatística R https://www.r-project.org/ Acesso: 17/01/2019

Hripcsak G, Duke JD, Shah NH., et al. Observational Health Data Sciences and Informatics (OHDSI): Opportunities for Observational Researchers. Studies in health technology and informatics. vol. 216 (2015): 574-578. MEDINFO'15; August 19–23, 2015; São Paulo, Brazil.

Madigan D, Ryan PB, Schuemie M, Stang PE, Overhage JM, Hartzema AG, et al. Evaluating the Impact of Database Heterogeneity on Observational Study Results. American Journal of Epidemiology [Internet]. 2013 Aug 15 [cited 2015 Nov 2];178(4):645–51. http://aje.oxfordjournals.org/cgi/doi/10.1093/aje/kwt010

Madigan D, Ryan P. Commentary: What Can We Really Learn From Observational Studies?: The Need for Empirical Assessment of Methodology for Active Drug Safety Surveillance and Comparative Effectiveness Research. Epidemiology [Internet]. 2011 Sep [cited 2015 Nov 2];22(5):629–31. http://content.wkhealth.com/linkback/openurl?sid=WKPTLP:landingpage&an=00001648-201109000-00006

Madigan D, Stang PE, Berlin JA, Schuemie M, Overhage JM, Suchard MA, et al. A Systematic Statistical Approach to Evaluating Evidence from Observational Studies. Annual Review of Statistics and Its Application [Internet]. 2014 Jan 3 [cited 2019 Apr 20]; 1(1):11–39. Available from: http://www.annualreviews.org/doi/10.1146/annurev-statistics-022513-115645

OHDSI-Vocabulary-CDM-Tutorial-2018-V1.pdf https://www.ohdsi-europe.org/images/symposium-2018/tutorials/OHDSI Vocabulary-CDM-Tutorial-2018-V1.pdf Acesso: 17/01/2019

OHDSI https://www.ohdsi.org/ Acesso: 17/01/2019

Vashisht R, Jung K, Schuler A, Banda JM, Park RW, Jin S, Li L, Dudley JT, Johnson KW, Shervey MM, Xu H, Wu Y, Natrajan K, Hripcsak G, Jin P, Van Zandt M, Reckard A, Reich CG, Weaver J, Schuemie MJ, Ryan PB, Callahan A, Shah NH. Association of Hemoglobin A1c Levels With Use of Sulfonylureas, Dipeptidyl Peptidase 4 Inhibitors, and Thiazolidinediones in Patients With Type 2 Diabetes Treated With Metformin: Analysis From the Observational Health Data Sciences and Informatics Initiative. JAMA Network Open 2018, 1(4), pp.e181755-e181755.

Yoon D, Ahn EK, Park MY, et al. Conversion and Data Quality Assessment of Electronic Health Record Data at a Korean Tertiary Teaching Hospital to a Common Data Model for Distributed Network Research. Healthc Inform Res. 2016;22(1):54-8.

SBCAS 2019

19° SIMPÓSIO BRASILEIRO DE COMPUTAÇÃO APLICADA À SAÚDE

DE 11 A 14 DE JUNHO DE 2019 EM NITERÓI-RJ

www.sbc.org.br/sbcas2019

REALIZAÇÃO

PATROCÍNIO

APOIO

ORGANIZAÇÃO