Representação de Variabilidade Estrutural de Dados por meio de Famílias de Esquemas de Banco de Dados

Larissa Cristina Moraes Rodrigues

Dissertação apresentada
ao
Instituto de Matemática e Estatística
da
Universidade de São Paulo
para
obtenção do título
de
Mestre em Ciências

Programa: Ciência da Computação

Orientador: Profa. Dra. Kelly Rosa Braghetto

Durante o desenvolvimento deste trabalho a autora recebeu auxílio financeiro da FAPESP
Processo: 2014/18216-2

São Paulo, fevereiro de 2017

Representação de variabilidade estrutural de dados por meio de famílias de esquemas de banco de dados

Esta versão da dissertação contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa da versão original do trabalho, realizada em 09/12/2016. Uma cópia da versão original está disponível no Instituto de Matemática e Estatística da Universidade de São Paulo.

Comissão Julgadora:

- Profª. Drª. Kelly Rosa Braghetto (orientadora) - IME-USP
- Profª. Drª. Solange Nice Alves de Souza - EP-USP
- Profª. Drª. Amanda Sávio Nascimento e Silva - UFOP

Agradecimentos

Queria agradecer primeiramente a Deus, por me acompanhar e me iluminar sempre para que eu conseguisse realizar mais um sonho em minha vida.

Agradeço aos meus pais pela base que me proporcionaram, seja ela emocional ou financeira, e por todo o apoio durante todas as etapas da minha vida. Aos meus irmãos que sempre me ouviram e estiveram ao meu lado torcendo por mim. E a toda a minha família, que sempre torceu por mim e comemorou comigo as minhas conquistas.

Agradeço ao meu marido Fernando pelo amor e companheirismo em todos os momentos juntos, além de todo o apoio emocional dado durante o meu mestrado e também às discussões que me ajudaram durante o trabalho.

Agradeço a minha orientadora Kelly Braghetto pelo suporte e pela dedicação ao meu trabalho. Agradeço também aos professores da banca de qualificação, Fabio Kon e Marcelo Finger, pelos comentários pertinentes a respeito do meu trabalho. E a professora Amanda Nascimento pelos comentários e dicas sobre o meu trabalho.

Agradeço a todos os amigos que fiz no mestrado, principalmente Elaine, Ana Paula e Diego, que estiveram comigo nas disciplinas, nos almoços e também nas apresentações durante o mestrado, sempre me ajudando e dando conselhos bons. E a todas as pessoas que participaram da etapa do estudo exploratório do meu mestrado, muito obrigada pela paciência e colaboração, com certeza esse trabalho não teria os mesmos resultados sem vocês.

Por fim, agradeço a todas as pessoas que direta ou indiretamente colaboraram pra que eu chegasse até aqui e também aquelas que torceram pelo meu sucesso.

Resumo

Rodrigues, L. C. M. **Representação de Variabilidade Estrutural de Dados por meio de Famílias de Esquemas de Banco de Dados**. 2017. 117 f. Dissertação (Mestrado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2017.

Diferentes organizações dentro de um mesmo domínio de aplicação costumam ter requisitos de dados bastante semelhantes. Apesar disso, cada organização também tem necessidades específicas, que precisam ser consideradas no projeto e desenvolvimento dos sistemas de bancos de dados para o domínio em questão. Dessas necessidades específicas, resultam variações estruturais nos dados das organizações de um mesmo domínio. As técnicas tradicionais de modelagem conceitual de banco de dados (como o *Modelo Entidade-Relacionamento* – MER – e a *Linguagem Unificada de Modelagem* – UML) não nos permitem expressar em um único esquema de dados essa variabilidade. Para abordar esse problema, este trabalho de mestrado propôs um novo método de modelagem conceitual baseado no uso de *Diagramas de Características de Banco de Dados* (DBFDs, do inglês *Database Feature Diagrams*). Esse método foi projetado para apoiar a criação de famílias de esquemas conceituais de banco de dados.

Uma família de esquemas conceituais de banco de dados compreende todas as possíveis variações de esquemas conceituais de banco de dados para um determinado domínio de aplicação. Os DBFDs são uma extensão do conceito de *Diagrama de Características*, usado na *Engenharia de Linhas de Produtos de Software*. Por meio dos DBFDs, é possível gerar esquemas conceituais de banco de dados personalizados para atender às necessidades específicas de usuários ou organizações, ao mesmo tempo que se garante uma padronização no tratamento dos requisitos de dados de um domínio de aplicação. No trabalho, também foi desenvolvida uma ferramenta *Web* chamada DBFD Creator, para facilitar o uso do novo método de modelagem e a criação dos DBFDs.

Para avaliar o método proposto neste trabalho, foi desenvolvido um estudo de caso no domínio de dados experimentais de neurociência. Por meio do estudo de caso, foi possível concluir que o método proposto é viável para modelar a variabilidade de dados de um domínio de aplicação real. Além disso, foi realizado um estudo exploratório com um grupo de pessoas que receberam treinamentos, executaram tarefas e preencheram questionários de avaliação sobre o método de modelagem e a sua ferramenta de software de apoio. Os resultados desse estudo exploratório mostraram que o método proposto é reprodutível e que a ferramenta de software tem boa usabilidade, amparando de forma apropriada a execução do passo-a-passo do método.

Palavras-chave: Bancos de Dados, Modelagem conceitual, Diagrama de Características, Variabilidade de dados.

Abstract

Rodrigues, L. C. M. **Representing Structural Data Variability Using Families of Database Schemas**. 2017. 117 f. Thesis (Master) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2017.

Different organizations within the same application domain usually have very similar data requirements. Nevertheless, each organization also has specific needs that should be considered in the design and development of database systems for that domain. These specific needs result in structural variations in data from organizations of the same domain. The traditional techniques of database conceptual modeling (such as *Entity Relationship Model* – ERM – and *Unified Modeling Language* – UML) do not allow to express this variability in a single data schema. To address this problem, this work proposes a new conceptual modeling method based on the use of *Database Feature Diagrams* (DBFDs). This method was designed to support the creation of families of conceptual database schemas.

A family of conceptual database schemas includes all possible variations of database conceptual schemas for a particular application domain. The DBFDs are an extension of the concept of *Features Diagram* used in the *Software Product Lines Engineering*. Through DBFDs, it is possible to generate customized database conceptual schemas to address the specific needs of users or organizations at the same time we ensure a standardized treatment of the data requirements of an application domain. At this work, a Web tool called DBFD Creator was also developed to facilitate the use of the new modeling method and the creation of DBFDs.

To evaluate the method proposed in this work, a case study was developed on the domain of neuroscience experimental data. Through the case study, it was possible to conclude that the proposed method is feasible to model data variability of a real application domain. In addition, an exploratory study was conducted with a group of people who have received training, executed tasks and filled out evaluation questionnaires about the modeling method and its supporting software tool. The results of this exploratory study showed that the proposed method is reproducible and that the software tool has good usability, properly supporting the execution of the method's step-by-step procedure.

Keywords: Databases, Conceptual modeling, Feature Diagram, Data variability.

Sumário

Lista de Abreviaturas

BD	Banco de Dados (*Database*)
ER	Modelo Entidade Relacionamento (*Entity-Relationship Model*)
EER	Modelo Entidade Relacionamento Estendido (*Enhanced Entity-Relationship Model*)
UML	Linguagem de Modelagem Universal (*Unified Modeling Language*)
LPS	Linha de Produto de Software (*Software Product Line*)
DBFD	Diagrama de Características de Banco de Dados (*Database Feature Diagram*)
BPMN	Business Process Model and Notation
SQL	Script Query Language
SGBD	Sistema Gerenciador de Banco de Dados
XML	Extensible Markup Language
SBBD	Simpósio Brasileiro de Banco de Dados

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Contexto e Motivação

Diferentes organizações dentro de um mesmo domínio de aplicação costumam ter requisitos de dados bastante semelhantes. Apesar disso, cada organização também tem necessidades específicas, que precisam ser consideradas no projeto e desenvolvimento dos sistemas de bancos de dados para o domínio em questão. Por essa razão, alguns domínios de aplicação podem apresentar um alto grau de variabilidade estrutural em seus dados.

Um exemplo disso acontece em grandes projetos de pesquisa, que coletam diversos tipos de dados experimentais. Um projeto de pesquisa desse tipo geralmente tem vários laboratórios de pesquisa participantes, cada um deles conduzindo tipos específicos de experimentos e armazenando seus dados experimentais em seu próprio banco de dados local. Nesse cenário heterogêneo, o projeto de pesquisa deve garantir que: (i) todas as informações essenciais sobre os experimentos sejam coletadas por todos os laboratórios participantes; e, (ii) todos os dados coletados pelos laboratórios participantes estejam organizados em uma estrutura padronizada, para permitir o compartilhamento de dados entre os membros do projeto. Em outras palavras, é desejável que todos os bancos de dados locais tenham uma parte comum em seus esquemas, e que as variações existentes entres seus esquemas seja de alguma forma controlada por um modelo de dados padrão especificado para o projeto.

Um cenário similar ocorre em uma companhia matriz e suas filiais. Dentro dessa hierarquia, todas as filiais vendem produtos ou serviços de um mesmo catálogo e seguem uma mesma política de vendas, portanto, devem conter estruturas em comum nos seus bancos de dados locais. Ao mesmo tempo, cada filial tem suas particularidades (por exemplo, algumas podem vender os serviços A, B e D, enquanto outras vendem somente os serviços A e C). Portanto, as estruturas de dados contidas nos bancos de dados locais variam de filial para filial. É importante que o armazenamento desses dados nos bancos de dados locais aconteça de forma padronizada, para facilitar o gerenciamento que matriz faz das filiais e a manutenção de uma visão centralizada dos seus dados.

Representações padronizadas para dados de um domínio de aplicação costumam ser criadas por meio de esquemas conceituais de banco de dados. Entretanto, domínios com variabilidade em suas estruturas de dados impõem desafios à modelagem conceitual. As técnicas tradicionais de modelagem conceitual de bancos de dados (como o *Modelo Entidade Relacionamento* - ER [EN10] e a *Linguagem Unificada de Modelagem* - UML [uml]) não nos permitem representar em um mesmo modelo as diferentes variações de esquema do domínio. Cada possível variação de esquema deve ser definida como um modelo separado. Isso torna difícil manter e evoluir os esquemas conceituais de banco de dados do domínio, principalmente considerando suas estruturas comuns (ou frequentes), que devem ser replicadas em todos os modelos.

A representação de dados relacionados mas que podem ter estruturas heterogêneas pode ser feita por meio de modelos de dados semiestruturados. Assim como ocorre no modelo ER e na UML, em modelos de dados semiestruturados, objetos de dados semanticamente relacionados podem ser agrupados em classes e suas hierarquias. Entretanto, nos modelos de dados semiestruturados, os objetos de dados de uma mesma classe não precisam ter uma mesma estrutura: eles podem conter

atributos diferentes. Além disso, nesses modelos, a estrutura dos objetos de dados não precisa seguir esquemas predefinidos, como ocorre no modelo ER e na UML. Dois exemplos bastante usados de notações para dados semiestruturados são a XML (eXtensible Markup Language) [BPSM+98] e o JSON (JavaScript Object Notation) [jsoa].

A flexibilidade dos modelos de dados semiestruturados é bastante conveniente para a representação de dados em vários cenários comuns na atualidade, como, por exemplo, na troca e exibição de dados na web. Mas alguns domínios de aplicação (como os citados no início deste capítulo) ainda demandam dados com estruturações mais padronizadas. Linguagens como XMLSchema [xml] e JSON Schema [jsob] permitem definir esquemas para a validação de documentos de dados semiestruturados quanto à estrutura. Mas nos esquemas definidos com esse tipo de linguagem, conseguimos apenas especificar quais tipos de objetos de dados e atributos são obrigatórios e quais são opcionais. Não conseguimos, por exemplo, especificar as dependências que podem existir entre diferentes tipos de objetos de dados, nem regras sofisticadas para a definição da obrigatoriedade de tipos de objetos e atributos.

Assim, tanto os modelos para dados estruturados quanto os modelos para dados semiestruturados não são expressivos o suficiente para descrever a variabilidade estrutural que deve ser admitida nos dados em função dos diferentes tipos de requisitos que se pode observar em um domínio de aplicação a ser modelado.

1.2 Objetivos

Considerando a motivação apresentada na seção anterior, este trabalho de mestrado tem como objetivo principal possibilitar a criação de representações padronizadas de dados para domínios de aplicação que possuem algum nível de variabilidade estrutural em seus dados. Para isso, neste trabalho nós sustentamos que é possível adaptar técnicas e modelos de Linhas de Produtos de Software (LPSs), utilizados para representar variabilidade de funcionalidades de sistemas de software, para serem aplicados na modelagem conceitual de dados.

Os objetivos específicos deste trabalho são:

- Criar um novo tipo de modelo conceitual capaz de representar variabilidade estrutural dos dados;
- Definir um método para guiar projetistas de bancos de dados no uso do novo tipo de modelo conceitual;
- Desenvolver uma ferramenta computacional de apoio à aplicação do método e ao uso do novo tipo de modelo;
- Avaliar o modelo e o método propostos, bem como a ferramenta de apoio desenvolvida.

1.3 Contribuições

As principais contribuições deste trabalho de mestrado são:

- o desenvolvimento de um novo tipo de modelo conceitual – os Diagramas de Características de Banco de Dados (DBFDs) – para representar variabilidade estrutural de dados;
- a definição do conceito de famílias de esquemas conceituais de banco de dados e o especificação de um método para a criação dessas famílias;
- a criação de uma ferramenta computacional de apoio ao uso de DBFDs, que também gera de forma automática *scripts* para a criação ou evolução de bancos de dados (físicos) a partir de esquemas conceituais derivados das famílias de esquemas modeladas a partir do novo método;

- a modelagem de uma família de esquemas conceituais para representar dados de experimentos de neurociência.

Para representar variabilidade estrutural dos dados, desenvolveu-se um novo tipo de modelo conceitual, os *Diagramas de Características de Banco de Dados (DBFDs)*, que são uma extensão dos *Diagramas de Características* usados nas LPSs. Um DBFD representa uma família de esquemas conceituais de banco de dados, que compreende todas as possíveis variações de esquemas conceituais de banco de dados para um determinado domínio de aplicação. Em um DBFD, os requisitos de dados do domínio são modelados como *módulos de dados* e *ligações* entre os módulos de dados. Cada módulo de dados modela (usando o modelo ER) objetos de dados que estão fisicamente ou semanticamente relacionados. As ligações (e suas *anotações* associadas) definem como os módulos podem ser conceitualmente combinados de forma a satisfazer os requisitos de dados do domínio modelado.

Por meio dos DBFDs, é possível gerar esquemas conceituais de banco de dados personalizados para atender às necessidades específicas de usuários ou organizações ao mesmo tempo em que é garantida a padronização no tratamento dos requisitos de dados do domínio de aplicação. Todos os esquemas gerados a partir de um mesmo DBFD são membros da mesma família de esquemas, que tem uma estrutura base comum (i.e., módulos de dados obrigatórios). E todas as variações que podem aparecer nos esquemas de uma família (i.e., módulos opcionais) são totalmente controladas pelo DBFD correspondente.

Para derivar um esquema conceitual personalizado a partir de um DBFD, basta escolher quais módulos de dados opcionais farão parte do esquema, lembrando que os módulos de dados obrigatórios estarão em todos os esquemas personalizados gerados a partir do mesmo DBFD e que a escolha dos opcionais deve respeitar as restrições de composição definidas pelas ligações entre os módulos. Quando surge um novo requisito de dados para o domínio de aplicação, é possível evoluir o DBFD por meio da criação de novos módulos de dados, ligando-os com os módulos já existentes e descrevendo as ligações por meio das anotações.

Para guiar os projetistas de bancos de dados na criação de DBFDs, este trabalho definiu um método que descreve todas as etapas envolvidas na criação e manutenção de um DBFD, desde o levantamento dos requisitos de dados do domínio até a derivação de esquemas conceituais personalizados e a evolução do DBFD. Uma ferramenta de software de código aberto, chamada DBFD Creator, foi desenvolvida para apoiar o uso do método e dos DBFDs. Além de gerar uma visualização gráfica para um esquema conceitual personalizado derivados de um DBFD, essa ferramenta também gera *scripts* SQL que permite a criação do banco de dados físico correspondente.

Diferente dos trabalhos existentes na área de Linhas de Produtos de Software que fazem uso intensivo de modelos de dados para criação de famílias de aplicação de software, o novo modelo apresentado possibilita criar famílias de esquemas conceituais, sem considerar artefatos de software. Outros trabalhos da área de Banco de Dados lidam com a evolução de esquemas de dados físicos, diferentemente deste trabalho de mestrado que trabalha com a evolução de esquemas conceituais de banco de dados.

Como consequência dessas contribuições, esperamos facilitar o gerenciamento e o compartilhamento de dados entre organizações de um mesmo domínio de aplicação que possuem variações em seus requisitos de dados e que estejam associadas.

Parte das contribuições listadas aqui foram tema do artigo entitulado *"Creating Families of Conceptual Database Schemas Using Database Feature Diagrams (DBFDs)"*, publicado nos anais no XXX Simpósio Brasileiro de Banco de Dados (SBBD'15) [MB15]. O artigo descreve o método de modelagem de banco de dados desenvolvida neste trabalho de mestrado e a exemplifica com o estudo de caso sobre os dados de experimentos de neurociência.

1.4 Organização do Trabalho

No Capítulo 2, serão apresentados os principais conceitos relacionados aos de famílias de esquemas conceituais de bancos de dados: modelagem de banco de dados, com ênfase em modelagem

conceitual, e o paradigma de linhas de produtos de software. E para finalizar esse capítulo, são descritos os métodos utilizados para a etapa de validação deste trabalho de mestrado.

O Capítulo 3 contém trabalhos relacionados a esta proposta de mestrado, sendo dividido em três partes. A primeira apresenta abordagens que tratam a modelagem conceitual de banco de dados de forma flexível ou evolutiva. A segunda trata de trabalhos que abordam o uso de linhas de produtos de software na área de banco de dados. E a terceira apresenta trabalhos relacionados a modelagem conceitual de dados semiestruturados.

No Capítulo 4 são apresentados mais detalhes sobre a proposta desenvolvida neste trabalho de mestrado, com os conceitos básicos e o passo-a-passo a ser seguido para reproduzir a nova método de modelagem criada. Em seguida, o Capítulo 5 apresenta a ferramenta computacional criada pra auxiliar o uso do método criado.

O Capítulo 6 apresenta um estudo de caso, onde foi aplicado o método de modelagem conceitual criado em um domínio de aplicação com alto grau de variabilidade estrutural de dados: o domínio de experimentos de neurociência. E o Capítulo 7 apresenta um estudo exploratório onde foram utilizados diferentes tipos de questionários para avaliar o método proposto e a ferramenta de software desenvolvida.

Finalmente, no Capítulo 8 discutimos algumas conclusões obtidas neste trabalho. Analisamos as vantagens e desvantagens do método proposto e falamos de trabalhos futuros que podem ser realizados para aprofundar ou aperfeiçoar o que foi feito até o presente momento.

Capítulo 2

Fundamentação Teórica

O trabalho de mestrado descrito neste texto propõe o uso de famílias de esquemas conceituais de banco de dados para modelar variabilidade estrutural de dados. A criação dos esquemas conceituais de banco de dados envolve métodos e técnicas para modelagem de bancos de dados, que são descritos na Seção 2.1.

O conceito de famílias de esquemas de dados criado neste trabalho se baseia no de *famílias de artefatos de software*, um tópico explorado pelo paradigma de Linhas de Produtos de Software, que é introduzido na Seção 2.2.

Para validar o novo método de modelagem proposta no trabalho, foram feitas avaliações qualitativas, seguindo paradigmas encontrados na área de Engenharia de Software, como apresentado na Seção 2.3.

2.1 Modelagem de Banco de Dados

Um banco de dados (BD) é uma coleção de dados relacionados que podem ser armazenados fisicamente de diferentes formas, como, por exemplo, em papel ou em um disco rígido, por meio do uso de um computador. O ciclo de vida de um banco de dados consiste nas fases de definição, construção, manipulação e compartilhamento dos dados.

Uma etapa inicial importante da fase de definição de um banco de dados é a análise de requisitos de dados, onde serão especificadas as características dos dados a serem armazenados e as necessidades dos usuários com relação a esses dados. Todos os requisitos de dados devem ser documentados, facilitando a interação entre os usuários e o banco de dados.

A partir da análise de requisitos, começa a fase de definição do banco de dados, que consiste na modelagem conceitual do banco de dados, para a posterior modelagem física do mesmo. A modelagem se baseia em *modelos de dados*, que são conjuntos de conceitos que podem ser usados para criar um *esquema* que descreve a estrutura de um banco. Essa fase de definição é muito importante para que a construção do banco seja eficaz e eficiente [TLNJ11].

Um modelo conceitual de dados descreve a estrutura do banco de dados em alto nível, em uma representação que mesmo usuários não especialistas em banco de dados compreendem bem. Os modelos físicos de dados descrevem os detalhes sobre como os dados são armazenados fisicamente no computator. Existem modelos que estão entre os modelos conceituais e os físicos – os *modelos de dados representativos* (também chamados de *modelos de dados de implementação*). Esses modelos ocultam alguns dos detalhes sobre o armazenamento físico, mas podem ser implementados diretamente em um sistema de computador [EN10].

Existem diferentes modelos de dados para a criação do modelo conceitual, entre eles, o *Modelo Entidade-Relacionamento*, o *Modelo Entidade-Relacionamento Estendido* e a *Linguagem de Modelagem Universal*, que serão apresentados posteriormente.

2.1.1 Modelagem Conceitual com o Modelo Entidade-Relacionamento (MER)

O Modelo Entidade Relacionamento (ER) e suas variações, como o Modelo Entidade-Relacionamento Estendido (EER), são frequentemente usados para o projeto conceitual de aplicações de banco de dados.

O modelo ER se baseia em três conceitos básicos: *entidades*, *atributos* e *relacionamentos*. Uma entidade representa um elemento do mundo real, como um `pesquisador` ou um `experimento`. Um atributo corresponde a alguma propriedade de interesse da entidade, como o nome do pesquisador. Um atributo pode ser: *simples*, como `Nome`; *composto*, como `Endereço`, que é dividido em `Rua`, `Número` e `Cep`; de *valor único*, como `Idade`; *multivalorado*, como `Telefone`; e *derivado*, como `Idade`, que pode ser calculado a partir de `DataNasc` e da data de hoje [EN10].

Para um grupo de entidades similares damos o nome de *tipo de entidade*. Um tipo de entidade é definido por seu nome e seus atributos. Por exemplo, `Pesquisador` é um tipo de entidade que contém os atributos `Cpf`, `Nome`, `DataNasc`. Chave candidata é um ou mais atributos que identificam unicamente uma entidade, podendo ser a chave primária ou alternativa. O tipo de entidade `Pesquisador` tem como atributo chave `Cpf`, que é um atributo cujos valores são distintos para cada entidade do conjunto de entidades [EN10].

Um relacionamento se dá entre duas ou mais entidades e representa uma associação entre estas, como o relacionamento `trabalhaEm` entre um `Pesquisador` e um `Laboratório`. Um conjunto de associações ou relacionamentos entre entidades é definido como um *tipo de relacionamento* entre tipos de entidade. O tipo de relacionamento `trabalhaEm` tem o atributo `Horas` para indicar quantas horas cada pesquisador trabalha no laboratório [EN10].

A *cardinalidade* de um tipo de relacionamento especifica o número máximo de instâncias do relacionamento em que uma entidade pode participar. Por exemplo, o tipo de relacionamento `participaDe` entre `Pesquisador` e `ProjetoDePesquisa` tem cardinalidade N:N, pois um pesquisador pode participar de vários projetos de pesquisa e de um projeto de pesquisa participam vários pesquisadores. A *participação* especifica o número mínimo de instâncias do relacionamento em que cada entidade deve participar. No tipo de relacionamento `trabalhaEm`, a participação de `Pesquisador` é *total*, pois todo pesquisador deve trabalhar em um laboratório, enquanto que no tipo de relacionamento `gerencia` (entre `Pesquisador` e `Laboratório`) a participação de `Pesquisador` é *parcial*, pois nem todo pesquisador gerencia um laboratório [EN10].

Os diagramas ER são utilizados como uma notação diagramática associada ao MER [Che76]. O *Modelo Entidade-Relacionamento Estendido* (EER, do inglês *Enhanced Entity Relationship*) introduz construtores adicionais que possibilitam a representação de conceitos como o de *especialização*, *generalização* e *categorização*. A Figura 2.1 apresenta um exemplo de diagrama EER para o esquema de um laboratório de pesquisa [EN10].

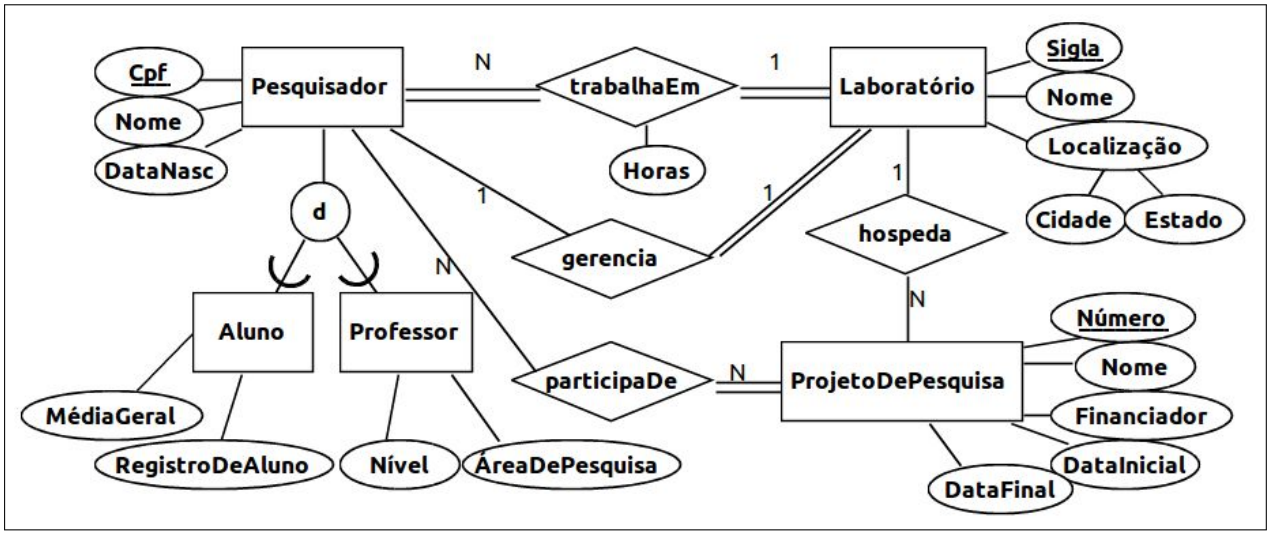

Figura 2.1: *Diagrama EER para um laboratório de pesquisa.*

Especialização é o processo de definir um conjunto de *subclasses* de um tipo de entidade; esse tipo de entidade é chamado de *superclasse* da especialização. Os tipos de entidade que são subclasses da especialização herdam os atributos e tipos de relacionamento da superclasse. Por exemplo, na Figura 2.1, `Aluno` e `Professor` são subclasses de `Pesquisador`. Generalização é o processo reverso da especialização, ou seja, as características comuns de diferentes tipos de entidades são identificadas e generalizadas em uma única superclasse. Uma especialização é disjunta quando toda entidade da superclasse pode ser especializada em no máximo uma das subclasses e é sobreposta quando pode ser especializada em mais de uma subclasse. Uma especialização total indica que toda entidade da superclasse deve ser membro de pelo menos uma das subclasses e é parcial quando permite que uma entidade não pertença a nenhuma das subclasses [EN10].

Na categorização, uma subclasse, chamada *categoria*, vai representar uma coleção de entidades distintas. Por exemplo, podemos dizer que um laboratório pode ser coordenado por um pesquisador ou uma empresa. Nesse caso, será criada a categoria `Coordenador` para representar as entidades `Pesquisador` e `Empresa`, que serão superclasses da categorização. Uma categoria total contém a união de todas as entidades em suas superclasses, enquanto que uma categoria parcial pode conter um subconjunto da união [EN10].

2.1.2 Modelagem Conceitual Baseada em Modelo de Objetos

Existem técnicas de modelagem de objetos, como a Linguagem de Modelagem Universal (UML, do inglês *Unified Modeling Language*), que são populares em projeto tanto de banco de dados quanto de software. A UML utiliza vários tipos de diagramas para ir além do projeto de dados e especificar detalhadamente o projeto de módulos de software e suas interações. Na UML, um dos tipos de diagrama mais utilizados é o *diagrama de classes*, como o da Figura 2.2, que representa um esquema de um projeto de pesquisa de forma similar à Figura 2.1. Nos diagramas de classe da UML, uma *classe*, quando utilizada para modelar dados, pode ser considerada semelhante a um tipo de entidade do ER, enquanto que as *associações* correspondem aos tipos de relacionamento do ER.

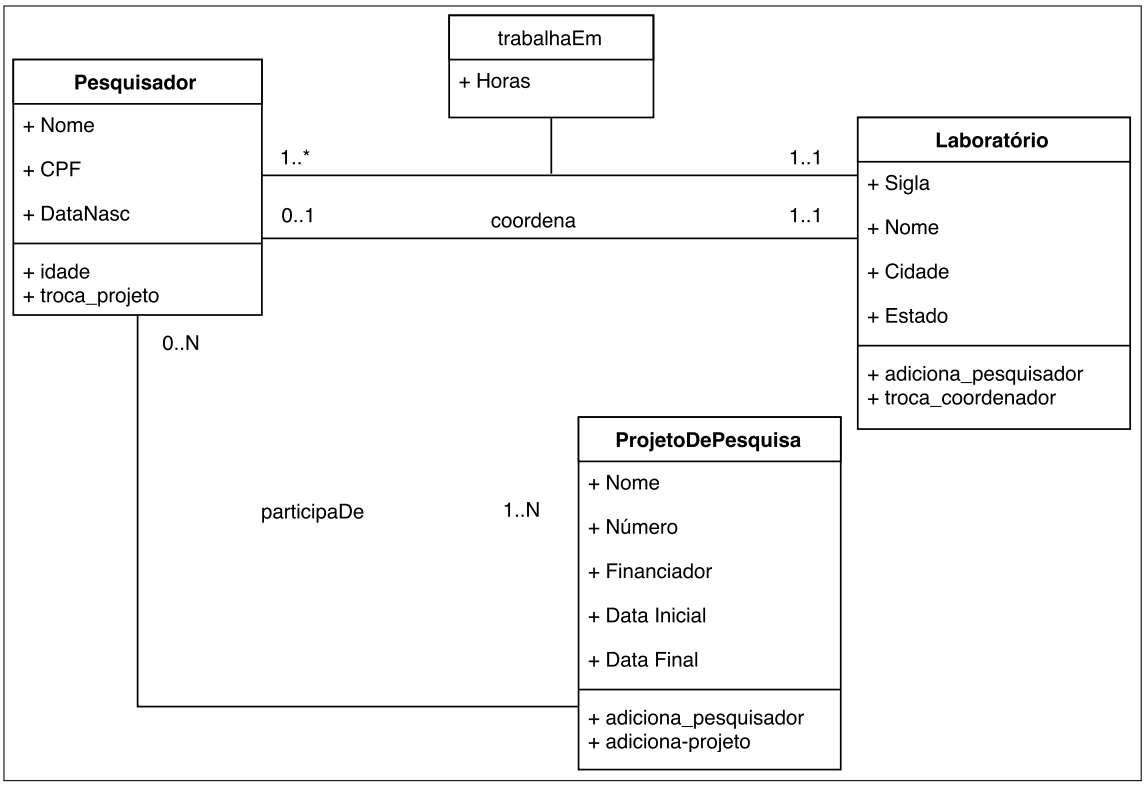

Figura 2.2: *Diagrama de Classes UML para um Projeto de pesquisa.*

2.1.3 Modelagem Física

Para que um esquema conceitual possa ser usado para criar um banco de dados real, é preciso que ele seja mapeado em um esquema de *modelo físico* ou de *implementação* que, por sua vez, será mapeado em comandos em uma linguagem de definição de dados (DDL - *data definition language*), como a linguagem SQL (*Structured Query Language*) [TLNJ11].

Os modelos mais usados para a modelagem física dos dados são modelos de implementação: o *Modelo Relacional* e o *Modelo de Dados de Objetos*. O Modelo Relacional representa o banco de dados como uma coleção de relações, onde uma relação é vista como uma tabela de valores, em que cada linha na tabela representa uma coleção de dados relacionados (podendo representar tanto uma entidade quanto um relacionamento). A chave primária de uma tabela será um atributo ou uma combinação de atributos que identificam de forma única uma linha da tabela. E a chave estrangeira é um atributo que é chave primária em outra relação.

Já o Modelo de Dados de Objetos utiliza muitos dos conceitos que foram desenvolvidos originalmente nas linguagens de programação orientadas a objeto. Bancos de dados relacionais que incorporam algumas características de bancos de dados de objetos são chamados de *objeto-relacionais*. Algumas características do Modelo de Dados de Objeto que são comumente incluídas nesses bancos são: construtores de tipos para especificar objetos complexos, um mecanismo para especificar a identidade de objetos, mecanismos de herança e mecanismos para o encapsulamento de operações.

O mapeamento de um modelo conceitual para um modelo de implementação pode ser feito de forma automática seguindo regras de mapeamento. Por exemplo, existe um algoritmo de sete passos para o mapeamento de modelos ER para modelos relacionais [EN10]. Os passos a serem seguidos, descritos de maneira simplificada, são:

1. Para cada tipo de entidade E, deve ser criada uma relação R_E com todos os atributos de E. Deve ser escolhida uma chave candidata de E para ser a chave primária da relação R_E.

2. Para cada tipo de relacionamento binário 1:1 entre os tipos de entidades $E1$ e $E2$, deve-se escolher uma das relações correspondentes, por exemplo R_{E2}, e incluir como chave estrangeira em R_{E2} a chave primária da outra relação (R_{E1}). Os atributos do relacionamento devem ser incluídos na relação com chave estrangeira.

3. Para cada tipo de relacionamento binário 1:N entre os tipos de entidades $E1$ e $E2$, deve ser escolhida a relação correspondente ao tipo de entidade do lado N, por exemplo R_{E2}, para incluir como chave estrangeira nela a chave primária da outra relação (R_{E1}). Os atributos do relacionamento também são incluídos na relação que recebeu a chave estrangeira.

4. Para cada tipo de relacionamento binário N:N entre os tipos de entidades $E1$ e $E2$, é necessário criar uma nova relação R_{rel} para representar o relacionamento e incluir como chaves estrangeiras em R_{rel} as chaves primárias de R_{E1} e R_{E2}. Os atributos das duas chaves irão compor a chave primária de R_{rel}. Os atributos do relacionamento devem ser incluídos na relação R_{re}.

5. Para um tipo de relacionamento de grau maior que 2, deve-se criar uma nova Relação R_{rel} para representar o relacionamento e incluir como chaves estrangeiras em R_{rel} as chaves primárias das relações correspondentes aos tipos de entidades que participam do relacionamento. A chave primária de R_{rel} deve ser a composição dessas chaves estrangeiras.

6. Para cada tipo de entidade fraca F deve ser criada uma relação R_F com todos os atributos de F. Também deve ser incluído como chaves estrangeiras em R_F as chaves primárias das relações correspondentes aos tipos de entidade fortes de F. A chave primária de R_F deve ser a composição da chave parcial de F com essas chaves estrangeiras.

7. Para cada atributo multivalorado A de um conjunto de entidades E, deve-se criar uma relação R_A com o atributo A e incluir como chave estrangeira em R_A a chave primária da relação

R_E correspondente a E. A chave primária de R_A será composta do atributo A mais a chave primária de R_E.

Portanto, por meio dos passos de levantamento de requisitos, modelagem conceitual e modelagem física constrói-se um banco de dados para um determinado domínio de aplicação. A partir de um modelo inicial de banco de dados, o próximo passo é evoluir o modelo a cada vez que existir um novo requisito de dados. Evoluir o banco de dados sem tornar o modelo inicial inconsistente é um dos desafios da área de banco de dados em termos operacionais, pois ainda não existem boas ferramentas computacionais para executar essa tarefa. Este trabalho de mestrado trabalha com a evolução em nível de modelo conceitual de banco de dados.

2.2 Linhas de Produtos de Software

Interesses divergentes de desenvolvedores e clientes e o alto grau de complexidade dos produtos de software são fatores complicadores no processo de desenvolvimento de software. Com o aumento da complexidade dos produtos de software, é comum haver repetição de código ou apenas pequenas modificações em parte do código para atender às necessidades de produto para usuários específicos. Isso torna difícil a manutenção e o gerenciamento dos produtos de software. As Linhas de Produtos de Software propõem soluções para lidar com esses problemas, com base na ideia de reúso de software no processo de desenvolvimento [HTO03].

Nesse contexto, uma linha de produto de software é definida como um conjunto de produtos, ou uma *família de produtos*, de software com alto grau de similaridade entre si que atendem às necessidades específicas de um determinado conjunto de usuários.[Har02]

O conceito de linhas de produtos foi completamente introduzido no começo dos anos 90, com uma das primeiras contribuições sendo a descrição do método FODA (*Feature-Oriented Domain Analysis*) [KCH+90]. Na mesma época, várias empresas começaram a trabalhar com esse conceito, como por exemplo, a Philips, introduzindo o método *Building-Block* [vdLM95].

A Engenharia de Linhas de Produtos de Software faz uma divisão entre desenvolvimento para reúso e desenvolvimento com reúso, correspondentes às fases de *Engenharia de Domínio e Engenharia de Aplicação*, como mostrado na Figura 2.3.

Na Engenharia de Domínio são definidas as funcionalidades comuns e variáveis para a família de produtos, que são utilizadas para construir uma infraestrutura para a linha de produto, como um repositório de artefatos reutilizáveis [Har02]. Diferentemente do que ocorre em muitas abordagens de reúso tradicionais que focam em artefatos de código, a infraestrutura da linha de produtos engloba todos os artefatos que são relevantes ao longo do ciclo de vida de desenvolvimento de software [LSR07].

No contexto de software, os artefatos podem ser modelos de requisitos, modelos de domínio, modelos de arquitetura, componentes de software, documentação, planos de testes, casos de testes, planos de trabalho e descrições de processos [Nor02].

Para prover a personalização de produtos, a infraestrutura definida na engenharia de domínio deve conter meios de satisfazer os requisitos de diferentes usuários. Para isso, os artefatos que podem diferir nas aplicações da linha de produtos são modelados usando variabilidade [VDLP05].

O conceito de *variabilidade de software* é fundamental para prover a reutilização de software e refere-se à capacidade de um sistema ou artefato de software ser modificado, personalizado ou configurado para ser utilizado em um contexto específico [KKL+98]. A variabilidade de um domínio pode ser expressa por meio de *características*, do inglês *features*, que são propriedades relevantes do sistema.

Uma característica pode ser do tipo obrigatória, opcional ou alternativa. As características obrigatórias de uma LPS são aquelas que estão presentes em todos os produtos da LPS, formando o núcleo da LPS. Uma característica opcional é aquela que pode ou não estar presente em um produto derivado da LPS. E uma característica alternativa indica que somente uma entre um grupo de duas ou mais características pode ser selecionada. [KCH+90, LKL02].

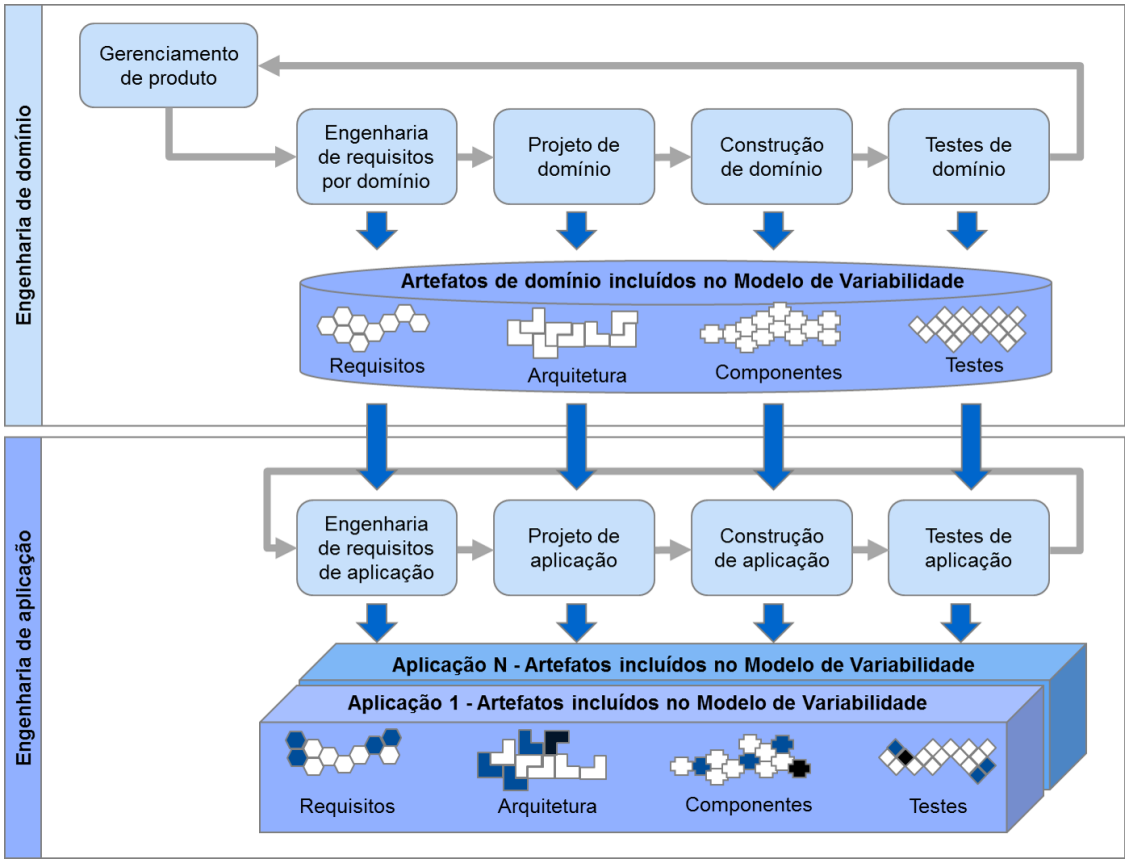

Figura 2.3: *Modelo dos ciclos de vida de LPS. Traduzido de [VDLP05].*

A fase de Engenharia de Domínio normalmente é dividida em três outras fases, que são: *análise do domínio*, *projeto do domínio* e *implementação do domínio* [Har02]. Na etapa de Análise do Domínio da fase de Engenharia de Domínio, as características de uma LPS são comumente documentadas em um *Diagrama de Características*. Um Diagrama de Características é uma forma de representar todas as possíveis configurações para um produto específico que pode ser construído a partir da LPS. Mais detalhes sobre isso serão apresentados na Seção 2.2.1.

Na Engenharia de Aplicação, utiliza-se a infraestrutura criada na fase anterior como base para derivar produtos específicos [CB11]. A variabilidade explicitamente modelada na infraestrutura fornece a base para derivar produtos individuais [LSR07]. A fase de Engenharia de Aplicação é dividida nas fases de *projeto* e de *implementação de aplicações individuais*.

Ao descrever os artefatos de software na fase de Engenharia de Domínio por meio do Diagrama de Características, é importante mantê-los flexíveis o suficiente para que os detalhes de implementação de um produto específico possam ser determinados facilmente na fase de Engenharia de Aplicação [CN01]. Para isso, são criados *pontos de variação*, que são locais do artefato de software em que uma decisão de projeto pode ser tomada, e *variantes*, que são alternativas de projeto associadas a esses pontos.

2.2.1 Abordagens Clássicas de Engenharia de Domínio

Para se criar uma família de esquemas conceituais que possa representar os dados de um determinado domínio de aplicação, o primeiro passo é fazer um levantamento de requisitos, que se assemelha à fase de Análise do Domínio, da Engenharia de Domínio. Esta seção apresenta alguns métodos da Engenharia de Domínio que têm potencial para serem utilizados no contexto de modelagem de dados. Esses métodos produzem como resultado das análises diagramas intuitivos, fáceis de serem compreendidos por usuários não-especialistas.

No método *FODA* (do inglês *Feature-Oriented Domain Analysis*) [KCH+90], o objetivo é iden-

tificar as características comuns e variáveis de aplicações similares de um mesmo domínio. Essas características são capacidades das aplicações do domínio do ponto de vista do usuário final. As características em FODA são divididas em *obrigatórias* (ou *comuns*), *alternativas* e *opcionais*.

Em FODA, são realizadas diversas análises, como análise de requisitos e análise de características, resultando em um modelo do domínio que contém as diferenças entre aplicações relacionadas [Har02]. A partir do modelo do domínio, são criados componentes reutilizáveis que possibilitam o desenvolvimento de múltiplas aplicações do domínio.

O FODA faz parte de uma abordagem baseada em modelo, que cobre tanto a fase de Engenharia do Domínio quanto de Engenharia de Aplicação. A parte de Engenharia do Domínio consiste em *análise do domínio*, que pode ser coberta pelo FODA, *projeto do domínio* e *implementação do domínio* [Har02].

O processo de análise de domínio pode ser dividido em três etapas [AR07]:

1. **Análise do contexto:** O objetivo é estabelecer os limites do domínio, a relação com outros domínios e o escopo da análise. No processo de levantamento dos requisitos é necessário coletar informações das diferentes atividades realizadas por diversas fontes do domínio. O resultado final dessa análise é documentado em diagramas de fluxo de dados e diagramas de estruturas.

2. **Modelagem do domínio:** O modelo de contexto gerado é analisado para gerar os modelos do domínio, que podem ser dos tipos: *modelo de informação*, que contêm as entidades do domínio e as relações entre elas; *modelo de características*, que são divididos em *diagramas de características*, *definições das características*, *regras* e *composição e lógica para características*; *modelo operacional*, que descreve o controle e fluxo de dados do domínio de aplicação e os relacionamentos entre objetos do modelo de informação e do modelo de características; e *dicionário da terminologia do domínio*.

3. **Modelagem da arquitetura:** Um modelo de arquitetura de alto nível é criado a partir dos modelos do domínio. Esse modelo será instanciado para criar aplicações individuais.

O *modelo RSEB* (do inglês *Reuse-Driven Software Engineering Business*) [JGJ97] é um método de reúso sistemático e dirigido a modelo. São compostos conjuntos de aplicações relacionadas a partir de conjuntos de componentes reutilizáveis. O RSEB utiliza a UML [RJB99] para especificar sistemas de aplicações, sistemas de componentes reutilizáveis e arquiteturas em camadas. A variabilidade entre os sistemas é expressa com pontos de variação e variantes anexadas [Har02].

O *modelo FeatuRSEB* (do inglês *Feature RSEB*) [GFd98] é uma extensão do modelo RSEB que utiliza um modelo de características similar ao do método FODA para catalogar ou indexar a comunalidade e a variabilidade capturada nos modelos do RSEB. De fato, os modelos FODA e RSEB têm muito em comum, pois ambos são métodos dirigidos a modelos que oferecem vários modelos correspondendo a diferentes pontos de vista do domínio [Har02]. A diferença entre o diagrama de características do método FODA e o do método FeatuRSEB está no tipo das relações dos dois diagramas [BHST04].

A Figura 2.4 exemplifica um diagrama de características para um sistema de telefones celulares baseado na notação do método FeatuRSEB. Nesse diagrama de características, podemos identificar os componentes de telefones móveis que são obrigatórios, ou seja, que constituem o módulo base do diagrama, que são: `tela` e `chamada`. O componente `tela` tem como opções o `básico`, o `colorido` e de `alta resolução`, e somente uma dessas opções poderá ser escolhida. A relação entre `tela` e seus filhos é um *XOR* (OU-Exclusivo).

`GPS` e `mídia` são componentes opcionais para os telefones. O componente `mídia` tem duas opções, que são `câmera` e `MP3`, e pelo menos uma delas deve ser escolhida. A relação entre `mídia` e seus filhos é um *OR*.

Outras relações presentes no diagrama são: a relação *exclui*, que indica que se houver o componente `GPS`, então não é possível ter o componente `tela` do tipo `básico`; e a relação *requer*, que indica que se houver o componente de `mídia` do tipo `câmera`, então é necessário ter o componente `tela` de `alta resolução`.

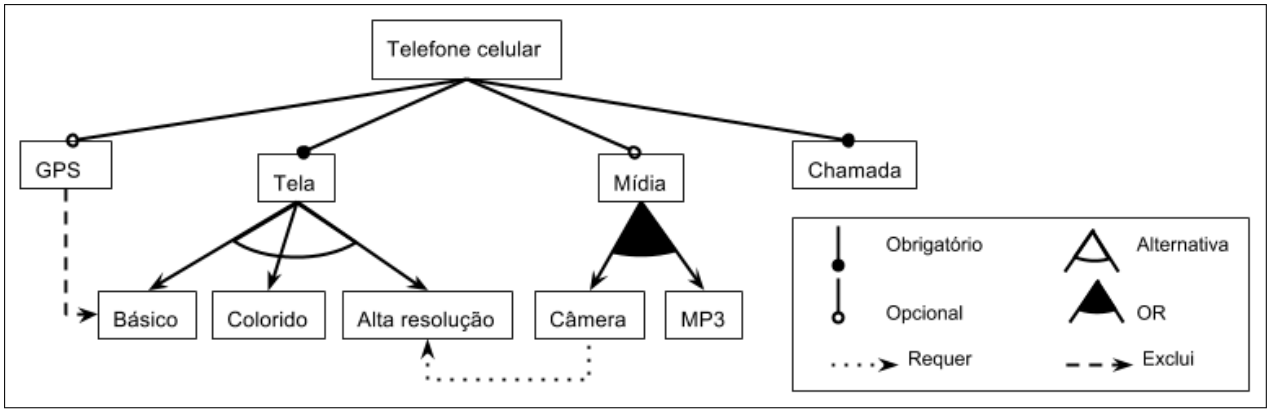

Figura 2.4: *Diagrama de características para sistema de telefones celulares. Adaptado de [BSRC].*

Existem ainda outros métodos, como: *o FORM* (do inglês *Feature-Oriented Reuse Method*) [KKL+98], que é uma extensão do FODA para incluir a fase de Engenharia de Aplicação, *o ODM* (do inglês *Organization Domain Modeling*) [SCK+96], que tem como foco a fase de Engenharia de Domínio de sistemas legados, o *PuLSE* (do inglês *Product Line Software Engineering*) [BFK+99], que é dividido em três fases e cada uma delas é associada com a evolução da infraestrutura da linha de produtos, e *o FAST* (do inglês *Family-Oriented Abstraction, Specification, and Translation*) [WL99], que cobre todo o processo de Engenharia de Linhas de Produtos de Software.

2.3 Técnicas de Avaliação

As pesquisas científicas do tipo quantitativa e qualitativa são muito utilizadas para avaliar um sistemas ou uma metodologia. Esses tipos de pesquisa são chamadas de pesquisa empírica, pois avaliam e observam a experiência real de um usuário ao utilizar um sistema ou uma metodologia.

Existem diversas definições que distinguem a pesquisa quantitativa da pesquisa qualitativa. Segundo [KPP+02], a pesquisa quantitativa é baseada na medida normalmente numérica de poucas variáveis objetivas, na comparação de resultados e no uso intensivo de técnicas estatísticas. Variáveis objetivas são aquelas cujos diferentes observadores conseguem obter o mesmo resultado em observações distintas. Pode-se dizer que a essência da pesquisa quantitativa em Ciência da Computação é verificar o quão melhor é usar um novo sistema ou método considerando as alternativas já existentes. Já a pesquisa qualitativa é realizada por meio de observações de várias perspectivas dos usuários enquanto eles utilizam o sistema.

Outra definição [DSFG13] é de que as duas visões são diferentes pelo fato da qualitativa ser mais idealista e subjetiva, enquanto que a quantitativa é realista e objetiva. Na pesquisa qualitativa, é necessária uma interpretação mais subjetiva ao invés da quantificação de dados [dM01].

Segundo [Dem01], quando o pesquisador deseja investigar algo mais subjetivo, como a opinião sobre algo, esta é uma análise qualitativa, pois não se trata da coleta de dados quantificáveis, mas sim de aspectos individuais de cada participante do estudo. Na pesquisa quantitativa, os dados são mais facilmente ordenáveis e mensuráveis, ou seja, são mais palpáveis para se trabalhar, sendo o método científico mais utilizado.

Na pesquisa quantitativa basta uma amostra representativa para que seja feita uma generalização segura dos resultados, enquanto que na qualitativa, é necessário ser mais cuidadoso na análise dos dados, para que as conclusões sobre a pesquisa não fiquem erradas, ou enviesadas.

Nos estudos tanto qualitativos quanto quantitativos é comum o uso de questionários [MM07] como principal material para a validação. Um questionário é um conjunto de questões elaboradas de forma a gerar dados necessários para atingir os objetivos de um projeto de pesquisa [Par91]. O uso de questionários envolve as seguintes fases: elaboração das perguntas e respostas, escolha da população, avaliação das respostas, análise dos resultados [PK01].

Segundo Oliveira [dO01], os questionários devem apresentar as seguintes características: ser a

espinha dorsal de qualquer levantamento, reunir todas as informações necessárias e possuir linguagem adequada. Quando for necessário desenvolver um questionário novo, deve-se ter cuidado na elaboração das perguntas e respostas [PK01]. Recomenda-se que as questões sejam simples e diretas, não sejam frases negativas e que não contenham mais de uma pergunta. Para as respostas, é importante haver o balanço entre os extremos opostos das alternativas e também uma alternativa neutra. Para isso, recomenda-se o uso da escala de *Likert*, onde a pessoa que está respondendo o questionário especifica seu nível de concordância com uma afirmação [Bow14, GB05].

Após aplicar o questionário em um grupo de participantes da pesquisa, é necessário reportar os seus resultados. Normalmente, descreve-se a distribuição das respostas de cada questão ou das questões mais importantes do questionário. É importante escolher as análises estatísticas mais apropriadas para cada tipo de questão. Por exemplo, para questões categóricas, pode resumir o resultado reportando qual foi a resposta mais frequente. Já para questões ordinais, o resultado pode ser resumido reportando a mediana, o primeiro e o terceiro quartil e possivelmente até o valor mínimo e o valor máximo.

Existe uma outra forma de analisar os resultados de um questionário, onde se agrega as respostas de várias questões em uma só medida ou número, como é o caso dos questionários de usabilidade de software SUMI (Software Usability Measurement Inventory) [KC93], QUIS (Questionnaire for User Interaction Satisfaction) [SP92] e SUS (System Usability Scale) [Bro96].

2.4 Considerações Finais do Capítulo

O método de modelagem de dados introduzido neste trabalho de mestrado segue as etapas de modelagem apresentadas na Seção 2.1, iniciando com a análise de requisitos, seguida da modelagem conceitual conceitual e do mapeamento para um modelo físico. Pode-se considerar que as etapas de análise de requisitos e modelagem conceitual são análogas ao processo de Engenharia de Domínio da Engenharia de Linhas de Produtos de Software, enquanto que a modelagem física é análoga à Fase de Engenharia de Aplicação.

Para a modelagem conceitual do banco de dados, este trabalho optou pelo Modelo EER, por ele apresentar todas as estruturas necessárias e pela facilidade de entendimento dos seus diagramas. Para a modelagem física, foi utilizado o modelo relacional, combinado à linguagem SQL para gerar os bancos de dados reais.

Para representar a variabilidade de um domínio de aplicação, o trabalho utiliza extensões dos diagramas de características do método FeatuRSEB, por serem diagrama completos, que contêm todos os tipos de elementos necessários para a proposta.

Pode-se notar que o método proposto herdou alguns conceitos de modelagem de banco de dados, e outros de engenharia de linhas de produtos de software. Portanto, a ideia do trabalho de mestrado foi aplicar um paradigma de Engenharia de Software na área de Banco de Dados. E realizamos a etapa de validação para avaliar qual a eficiência dessa combinação de conceitos de duas áreas diferentes.

Na etapa de validação do trabalho de mestrado, optou-se por realizar a análise qualitativa, para que se tivesse maior flexibilidade na interpretação dos resultados dos testes aplicados e também porque as variáveis estudadas não eram numericamente mensuráveis. A validação pode ainda ser caracterizada como um estudo exploratório, pois há interferência direta do pesquisador, e estuda-se um método que foi definido recentemente para identificar os possíveis problemas que não foram previstos na concepção do método. O principal material para a validação foi o uso de questionários.

Capítulo 3

Trabalhos Relacionados

Neste capítulo serão apresentados alguns trabalhos relacionados ao projeto de mestrado apresentado neste texto. Esses trabalhos foram separados em três tópicos. O primeiro trata de trabalhos que apresentam abordagens flexíveis ou evolutivas para modelagem conceitual de bancos de dados. O segundo aborda trabalhos que aplicaram o paradigma de linhas de produtos de software em bancos de dados. E, por último, apresentamos trabalhos relacionados à modelagem conceitual de dados para dados semiestruturados.

3.1 Abordagens Flexíveis para Modelagem Conceitual de Bancos de Dados Estruturados

Como discutido no Capítulo 1, um desafio encontrado com frequência na modelagem de bancos de dados é o de expressar em um único esquema as variações dos requisitos de dados de um grupo de usuários dentro de um determinado domínio de aplicação. Os trabalhos encontrados na literatura que se relacionam a esse problema são os que propõem o uso de esquemas de dados mais flexíveis, para acomodar mais facilmente as alterações que podem ocorrer no banco de dados ao longo do tempo, quando o banco de dados já existe fisicamente.

Uma das abordagens consiste em criar uma extensão do modelo entidade-relacionamento (ER) de forma que em um mesmo diagrama ER seja representada uma versão anterior e a versão atual dos requisitos do banco de dados. Portanto, em um mesmo diagrama é possível identificar quais foram as alterações feitas de uma versão para a outra. No trabalho de Liu et al.[LCC94] é apresentado o *diagrama EVER* (do inglês *EVolutionary ER*), que estende as construções gráficas usadas nos diagramas ER a fim de possibilitar a especificação das relações de derivação entre versões do esquema, as relações entre os atributos e as condições para se manter versões consistentes do banco de dados.

Outra abordagem consiste em fazer uma previsão das modificações que poderão ocorrer no modelo conceitual de banco de dados e a partir disso fazer um mapeamento das mesmas para o modelo físico do banco de dados. O trabalho de Roddick et al. [RCR93] apresenta uma taxonomia de mudanças aplicáveis ao modelo entidade relacionamento juntamente com seus efeitos no modelo relacional subjacente, efeitos esses expressos em termos de uma segunda taxonomia. O conjunto de operações atômicas propostas para serem aplicadas ao modelo relacional deve resultar em um banco de dados consistente e, na medida do possível, reversível.

Uma terceira abordagem diz respeito a utilizar modelos mais genéricos em qualquer etapa da modelagem do banco de dados, tanto conceitual quanto física, e com todo tipo de modelo (e.g., o entidade-relacionamento, o relacional e o orientado a objetos). No trabalho de Hainaut et al. [HCMD92] é apresentado o *TRAMIS*, que é baseado em um único modelo genérico para expressar as estruturas de banco de dados em diferentes estágios da modelagem. Esse modelo único permite uma definição imparcial dos processos de design, bem como alta flexibilidade nas estratégias de design usadas pelos projetistas de banco de dados. Isso permite que uma transformação conceitual

também possa ser usada em um esquema físico. O modelo TRAMIS é uma extensão do modelo entidade-relacionamento para expressar estruturas tanto conceituais quanto físicas.

Os trabalhos que exploram essas três abordagens têm como principal objetivo propor modelos de dados flexíveis para expressar diferenças entre versões temporais de um banco de dados. Por outro lado, o trabalho de mestrado apresentado nesse texto busca lidar com as diferenças de requisitos de dados dentro de um mesmo domínio de aplicação em nível de modelagem conceitual, propondo um modelo onde possam ser especificadas essas diferenças ao mesmo tempo em que se destaca os requisitos comuns a todos os usuários do domínio. É importante ressaltar que essas diferenças de requisitos não precisam estar atreladas a mudanças que ocorrem no domínio com o passar do tempo.

Existem ainda abordagens puramente voltadas para bancos de dados físicos, como , por exemplo, os *bancos de dados particionados*, os *bancos de dados temporais* e os *bancos de dados evolutivos*. No particionamento de dados, o objetivo principal é dividir o banco de dados físico em várias partições, para que ele seja tratado de forma mais granular, de modo que uma alteração ou variação em uma partição não afete o desempenho do banco de dados como um todo [NCWD84]. Em um banco de dados temporal, são gravados estados do banco em determinados momentos no tempo para que seja possível rastrear as informações armazenadas nele em um dado instante [TCG+93]. E, finalmente, um banco de dados evolutivo é um banco que evolui de forma conjunta à evolução das aplicações que o acessam. Essa técnica consiste em aplicar integração contínua e refatoração no desenvolvimento do banco de dados, envolvendo os especialistas em banco de dados e também os desenvolvedores da aplicação [AS06].

Essas três abordagens mais físicas não se preocupam com a evolução do modelo conceitual do banco de dados, o que as difere do método proposto neste trabalho de mestrado, que tem entre os seus objetivos facilitar a evolução do modelo conceitual conforme novos requisitos de dados surgem no domínio de aplicação.

3.2 Uso de Linhas de Produtos de Software em Bancos de Dados

Os trabalhos da área de Linhas de Produtos de Software apresentam semelhança com os requisitos encontrados nos domínios de aplicação lidados neste trabalho de mestrado. A estratégia de se modelar a variabilidade em termos de artefatos de software pode ser extendida para modelar variabilidade em estruturas de dados.

Os artefatos criados e reusados nas Linhas de Produtos de Software (LPSs) geralmente são modelos de requisitos, componentes de software e planos de testes. Embora muitos produtos de software façam uso intensivo de dados que são mantidos em bancos de dados, há poucos trabalhos que se dedicam a propor maneiras de se lidar com os bancos de dados nas LPSs. Posteriormente são apresentados três trabalhos que propõem abordagens usando o paradigma de LPS com foco em bancos de dados.

O trabalho de Bartholdt et al. [BOR09] define uma abordagem para a modelagem e integração de dados em LPSs que usa desenvolvimento dirigido por modelos (*MDSD – Model Driven Software Development*). Nessa abordagem, a modelagem dos dados é feita por meio de diagramas da UML2 XMI (*Unified Modeling Language – XML Metada Interchange*), de forma integrada à modelagem das características da LPS.

Na abordagem proposta no trabalho de Zaid e Troyer [AZDT11], a modelagem dos dados também é feita conjuntamente à das características dos produtos de software da linha: todo item de dado que aparece no modelo deve estar associado a uma característica de software. E uma característica que tem um item de dado associada a ela é chamada no modelo proposto pelo autores de *característica de persistência*. Segundo os autores, a abordagem pode ser aplicada sobre qualquer tipo de modelo conceitual de dados.

No trabalho de Siegmund et al. [SKR+09], a ideia é decompor os esquemas em termos de características e para isso são criados dois tipos de decomposição: a decomposição física e a decomposição virtual. Na decomposição física, elementos do esquema de dados são armazenados em arquivos se-

parados, sendo um arquivo por característica. Já na decomposição virtual, os elementos do esquema conceitual não são fisicamente separados, mas são apenas anotados, ou seja, os elementos pertencentes a uma característica são deixados em um único esquema com outros elementos, mas são destacados.

Os trabalhos descritos anteriormente se baseiam na criação de diagramas de características de dados a partir de modelos conceituais de dados associados a diagramas de características de software. Já o trabalho de Khedri e Khosravi [KK13] não se baseia em diagramas de características, mas sim na técnica *delta-oriented programming*. Na abordagem proposta pelos autores, os dados são sempre descritos no nível de implementação (como comandos de definição de dados da SQL). O modelo de dados de um produto de software é gerado por meio da adição ao módulo núcleo de um módulo de dados *delta* para cada uma das características escolhidas para o produto.

Esses trabalhos mostram que é viável estender os métodos e ferramentas da Engenharia de Linhas de Produto de Software para a aplicação em Bancos de Dados. Entretanto, diferentemente do que ocorre no trabalho de mestrado apresentado neste texto, as abordagens descritas nesses trabalhos não tinham como alvo a criação de famílias de bancos de dados, mas sim a criação de famílias de aplicações que fazem uso de bancos de dados. Portanto, para esses trabalhos, o banco de dados é apenas mais um artefato de apoio, e não o produto final. Além disso, as famílias de esquemas conceituais criadas por meio dos métodos propostos neste trabalho são extensíveis. Os usuários finais podem incorporar novas características ao modelo de uma família, para que ela consiga lidar com novos tipos de requisitos não considerados na família inicial.

3.3 Modelos Conceituais para Dados Semiestruturados

Devido ao crescente número de aplicações web em uso na atualidade, existe um grande volume de dados semiestruturados sendo processado. Dados semiestruturados são organizados em entidades semânticas e não apresentam uma estrutura formal para tipos estritos. Em vez disso, são organizados de forma irregular e parcial. A linguagem XML está sendo utilizada como um padrão para o armazenamento e troca de informações estruturadas e semiestruturadas contidas na Internet. Porém, os esquemas de XML apresentam apenas uma representação lógica dos dados semiestruturados, sem se preocupar com a semântica dos dados [ABS00]. Para abordar esse problema, vários trabalhos propõem diferentes tipos de modelos conceituais para representar a semântica de dados semiestruturados [GS12].

As abordagens para criar modelos conceituais para dados semiestruturados se baseiam nos modelos conceituais para dados estruturados mais utilizados, que são os modelos Entidade Relacionamento (ER) e a Linguagem de Modelagem Unificada (UML). Abordagens como o EReX (*ER extended for XML*) [Man04], ERX (*Entity Relationship for XML*) [Psa00], XER (*Extensible Entity Relationship Modeling*) [SMD03] e XSEM [Nec07] estendem o modelo ER para acomodar a faceta dos dados semiestruturados a nível conceitual. A principal diferença entre esses modelos e os seus modelos conceituais de base é em relação a representação hierárquica dos dados. Na proposta de Necasky [Nec07], por exemplo, uma abordagem de dois níveis foi usada para representar as relações hierárquicas. No primeiro nível, o esquema conceitual baseia-se no modelo ER estendido, e no segundo nível, são desenhadas organizações hierárquicas de partes do esquema conceitual global.

No entanto, poucas tentativas foram realizadas para modelar os dados semiestruturados usando o paradigma orientado a objetos. O modelo ORA-SS (*Object-Relationship-Attribute Model for Semistructured Data*) [WLLD01] se propôs a criar os modelos conceituais partindo de sua estrutura hierárquica, e suporta parcialmente as características de orientação a objetos. As abordagens propostas baseadas em UML [LLY06, CSF00] suportam o paradigma orientado a objetos de forma abrangente e preenchem a lacuna entre o projeto de software orientado a objetos e os esquemas de dados semiestruturados. Essas propostas estendem as definições e notações do estereótipo UML.

Sarkar propôs um modelo conceitual baseado em semântica de grafos para dados semiestruturados, denominado Modelo de Dados Semiestruturado Orientado a Gráficos (GOOSSDM, do inglês

Graph Object Oriented Semi-Structured Data Model) [Sar12]. Esse modelo é baseado em um paradigma orientado a objetos e suporta a representação da estrutura hierárquica, juntamente com relações não-hierárquicas, conteúdo misto, ordenação, restrições de participação etc.

Nos trabalhos citados anteriormente, portanto, foram definidos diferentes tipos de modelos conceituais para lidar com a falta de expressividade de semântica existente em modelos usados para dados semiestruturados, como o modelo XML. Embora esses modelos possam expressar relações hierárquicas entre os objetos de dados de um domínio de aplicação, eles não são capazes de representar regras sofisticadas de depêndencia entre os objetos de dados. Portanto, não é possível usar os modelos conceituais de dados semiestruturados existentes para expressar os requisitos de dados dos domínios de aplicação tratados neste trabalho.

Capítulo 4

Família de Esquemas Conceituais de Banco de Dados

O desafio de se criar esquemas conceituais de banco de dados flexíveis que representem de forma padronizada a variabilidade estrutural dos dados de um domínio de aplicação colabora para a dificuldade existente na manutenção dos esquemas de dados e possível compartilhamento desses esquemas entre diferentes organizações do mesmo domínio. As técnicas tradicionais de modelagem conceitual de banco de dados não permitem que as diferentes variações de esquemas conceituais para um dado domínio de aplicação sejam representadas em um mesmo esquema conceitual.

A ideia de se criar uma família de esquemas conceituais de dados se baseia em um paradigma da Engenharia de Software chamado de Engenharia de Linhas de Produtos de Software (LPSs), também conhecido por famílias de aplicações. Como as abordagens clássicas de LPS não lidam particularmente com a variabilidade de dados e, mais especificamente, em nível de modelagem conceitual, este trabalho propõe o uso de um novo tipo de diagrama, o Diagrama de Características de Banco de Dados, que estende os Diagramas de Características de forma a possibilitar a representação da variabilidade de dados do domínio.

4.1 Diagramas de Características de Banco de Dados

Da mesma forma que os diagramas de características apoiam a criação de produtos de software personalizados e extensíveis, eles podem ser utilizados na modelagem de banco de dados para permitir que o projetista de banco de dados crie esquemas personalizáveis que possam ser facilmente estendidos para atender às necessidades específicas de algum usuário. Entretanto, a fim de possibilitar o uso dos diagramas de características no projeto conceitual de banco de dados, é necessário redefinir o conceito de *característica* (do inglês, *feature*), relacionando-o aos tipos de elementos em modelos conceituais, como: *tipo de entidade*, *atributo* e *tipo de relacionamento*.

Dessa forma, estamos propondo a criação dos Diagramas de Características de Banco de Dados – chamados *Database Feature Diagrams* (DBFDs) em inglês [1] – uma extensão dos diagramas de características do método featuRSEB dedicada à modelagem conceitual de banco de dados. Um DBFD é sobretudo composto de três tipos de elementos: *módulos*, *ligações* e *anotações*.

Em Linhas de Produtos de Software, a seleção de características no diagrama resulta em um novo produto de software personalizado contendo essas características. De maneira análoga, a seleção de módulos de dados no diagrama DBFD resulta em um esquema conceitual de banco de dados personalizado contendo esses módulos e as ligações que podem existir entre eles.

[1] Neste texto, estamos empregando um nome em inglês – *Database Feature Diagram* (DBFD) – para o novo método de modelagem proposto pois esse foi o nome atribuído a ele em um artigo publicado no *Simpósio Brasileiro de Banco de Dados* (SBBD) 2015.

4.1.1 Conceitos Básicos

Um *módulo de dados* é o artefato reutilizável em um DBFD (correspondente ao conceito de 'característica' dos diagramas de características clássicos) e pode ser definido como uma *partição do modelo conceitual de banco de dados*, agrupando objetos de dados que estão fisicamente ou semanticamente relacionados. As *ligações* expressam as dependências e restrições existentes entre os módulos. Já as *anotações* são uma característica especial que foi introduzida nos DBFDs para aperfeiçoar a expressividade das ligações.

A Figura 4.1 mostra um exemplo de DBFD para o domínio de pesquisas científicas. Na notação gráfica, um módulo de dados é representado como um retângulo contendo seu nome dentro, enquanto as ligações são arcos rotulados partindo de um módulo de origem para um módulo de destino. O rótulo de uma ligação é seu nome de identificação e, por esse motivo, deve ser único no DBFD. As anotações são expressas textualmente e aparecem anexadas ao diagrama gráfico. No caso da Figura 4.1, não são apresentadas as anotações.

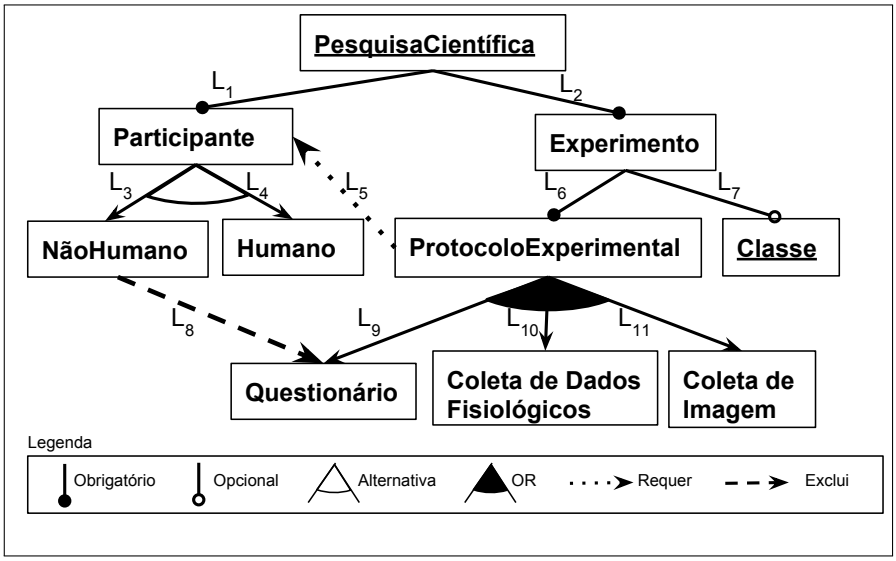

Figura 4.1: *Exemplo de Diagrama de Características de Banco de Dados (DBFD).*

Normalmente, quando um módulo de dados é selecionado no DBFD, um conjunto de objetos de dados conceituais vai ser criado no esquema conceitual. Se isso não acontece, então esse módulo é chamado *módulo de dados vazio* e é graficamente representado de forma similar aos outros módulos, mas com seu nome sublinhado (por exemplo, `PesquisaCientífica`). Um DBFD tem também um módulo especial, o *módulo raiz*, que se refere à família de esquemas conceituais modelada no diagrama por completo e é desenhado no topo do DBFD (por exemplo, `PesquisaCientífica`). O módulo raiz deve ser visto como um módulo vazio, pois ele não gera nenhuma modificação no esquema conceitual.

As Figuras 4.2 e 4.3 mostram os diagramas Entidade-Relacionamento Estendidos (EER) que estão contidos nos módulos de dados `Participante` e `Humano`, respectivamente. O módulo de dados Pesquisa Científica não contém nenhum diagrama EER associado, por ser um módulo de dados vazio. Para exemplificar o módulo de dados vazio, podemos observar a Figura 4.4, que contém o módulo `Experimento` com o diagrama EER contido nele e o módulo de dados `Classe`, que é um módulo vazio. A ligação L_7 terá uma anotação para adicionar o atributo `classificação` no tipo de entidade `Experimento`, que pertence ao módulo `Experimento`, sem que sejam adicionados novos tipos de entidade ou de relacionamento ao diagrama EER.

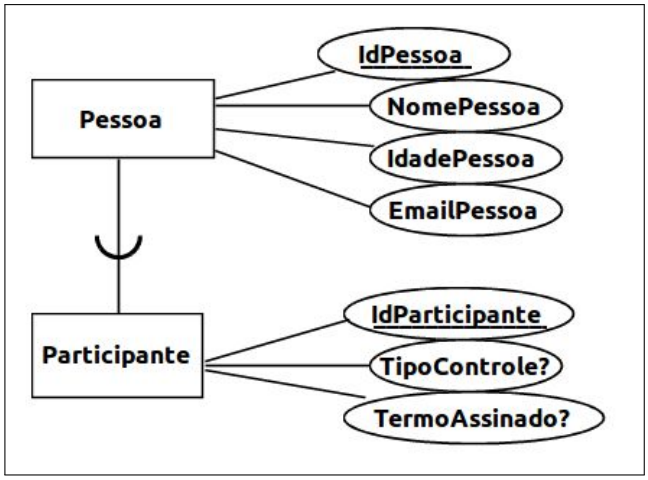

Figura 4.2: *Módulo Participante.*

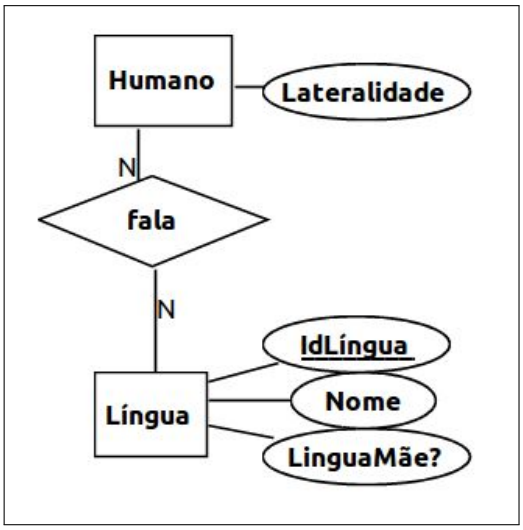

Figura 4.3: *Módulo Humano.*

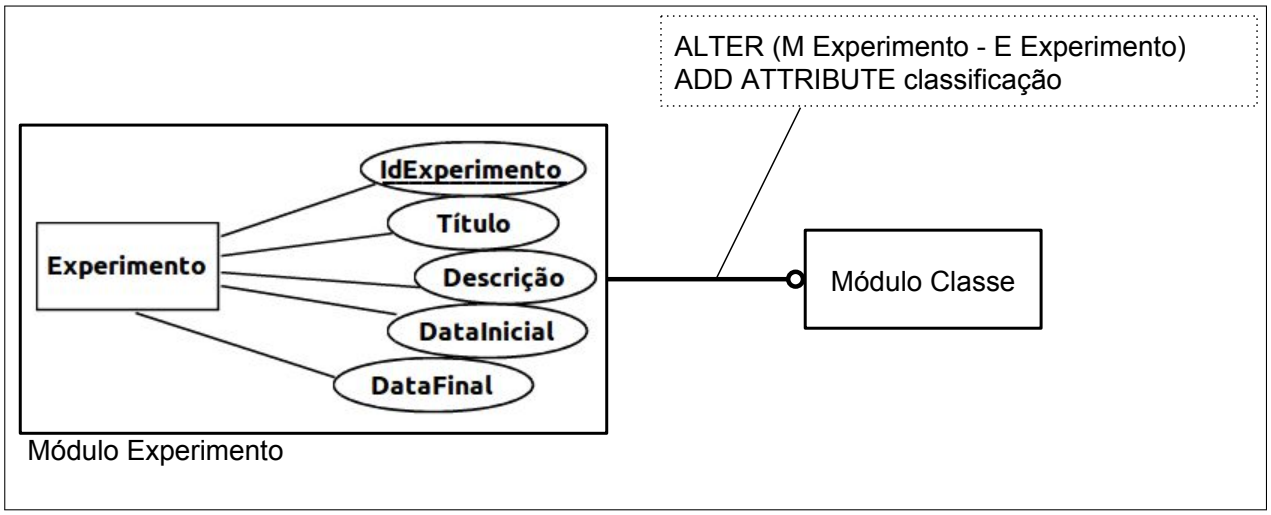

Figura 4.4: *Módulo Experimento e Classe.*

4.1.2 Ligações nos DBFDs

As ligações nos DBFDs correspondem ao conceito de relações nos diagramas de características das LPSs. Para que o termo *relação* (dos diagramas de características) não fosse confundido com o termo *relacionamento* (do modelo EER), neste trabalho o termo *relação* foi substituído pelo termo *ligação*.

DBFDs podem expressar os seguintes tipos de ligações:

- *Ligação associativas* – liga um módulo pai aos módulos filhos usados para compô-lo;
- *Ligação de restrição* – cruza a árvore para ligar um par de módulos, ou seja, é uma relação que não é necessariamente parental.

Existem quatro tipos de ligações associativas (todas elas ilustradas na Figura 4.1):

- *Composição obrigatória* – denota um módulo filho que é necessário. É graficamente representada por uma reta terminando com um círculo preenchido, ligando o pai ao filho obrigatório. Exemplo: *PesquisaCientífica* obrigatoriamente contém *Experimento*;
- *Composição opcional* – denota um módulo filho que é opcional. É graficamente representada por uma reta terminando com um círculo vazio, ligando o pai ao filho opcional. Exemplo: *Experimento* pode ter *Classe*;
- *Composição OU* – denota que pelo menos um dos módulos filhos deve ser selecionado. É desenhada como um ângulo preenchido no módulo pai unindo os arcos de ligação dos filhos. Exemplo: o módulo *ProtocoloExperimental* deve conter *Questionário*, ou *Tarefa*, ou *Estímulo*, ou qualquer combinação desses três;
- *Composição OU-Exclusivo*, também chamada de *composição alternativa* – denota que um e somente um dos módulos filhos deve ser selecionado. É desenhada como um ângulo vazio no módulo pai unindo os arcos de ligação dos filhos. Exemplo: o módulo *Participante* contém *Humano* ou *NãoHumano*, mas não pode conter os dois.

Existem dois tipos de ligações de restrição que podem ser definidas a partir de um módulo de origem A para um módulo de destino B:

- *Requer* – denota que se o módulo A é selecionado, então o módulo B também deve ser selecionado. É desenhada com uma seta pontilhada unindo dois módulos. Exemplo: *ProtocoloExperimental* requer *Participante*;
- *Exclui* – denota que se o módulo A é selecionado, então o módulo B não pode ser selecionado (isto é, os módulos A e B não podem ser parte do mesmo esquema conceitual). É desenhada com uma seta tracejada unindo dois módulos. Exemplo, *NãoHumano* exclui *Questionário*.

Os tipos de ligações e suas respectivas representações gráficas são mostrados na Figura 4.5.

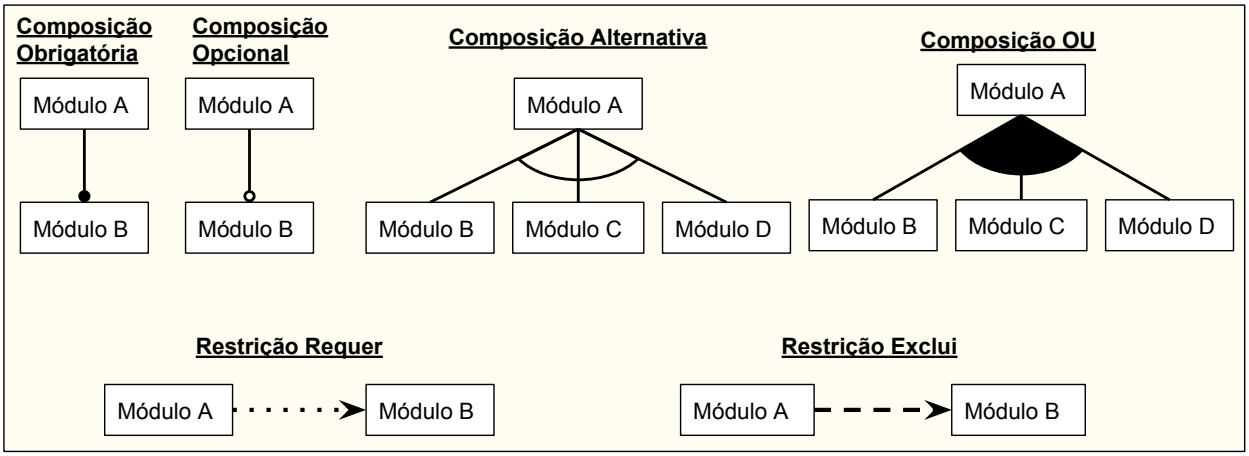

Figura 4.5: *Representação das ligações no DBFD.*

4.1.3 Anotações nos DBFDs

As *anotações* são um recurso exclusivo dos DBFDs, isto é, elas não existem nos diagramas de características clássicos. Uma anotação está sempre associada a uma ligação e tem como propósito descrever as modificações que devem ser realizadas nos esquemas conceituais dos módulos envolvidos na ligação a fim de criar esquemas conceituais personalizados.

Um módulo de dados no DBFD é um diagrama no modelo Entidade-Relacionamento Estendido (EER) e os tipos de objetos de dados que podem ser criados ou modificados nos módulos de dados por meio das anotações são os mesmos existentes no modelo EER: *tipos de entidade*, *tipos de relacionamento*, *atributos*, *especializações* e *categorizações*.

Anotações são declarações textuais que seguem um formato bem definido e sua sintaxe é inspirada nos comandos de modificação de esquemas da linguagem SQL. Cada anotação denota uma modificação atômica, ou seja, a inclusão, atualização ou exclusão de um único objeto de dados no esquema conceitual de um módulo de dados. Cada ligação pode ser associada a várias anotações, mas uma anotação pertence somente a uma ligação. Como exemplo, são mostradas a seguir algumas anotações para o DBFD da Figura 4.1. Essas anotações foram feitas sobre as ligações do DBFD que possuem rótulos de L_1 até L_{11}.

Exemplo 1: Anotação para inserção de atributos em tipos de entidades ou tipos de relacionamentos.

```
L1: ALTER (MODULE Participante - ENTITY Participante) ADD ATTRIBUTE Atributo1;
L1: ALTER (MODULE Humano - RELATIONSHIP fala) ADD ATTRIBUTE Atributo2;
```

Exemplo 2: Anotação para remoção de atributos em tipos de entidades ou tipos de relacionamentos.

```
L2: ALTER (MODULE Experimento - ENTITY Experimento) DROP ATTRIBUTE DataFinal;
L2: ALTER (MODULE Experimento - RELATIONSHIP fala) DROP ATTRIBUTE Atributo2;
```

Exemplo 3: Anotação para criação de tipos de relacionamento entre tipos de entidades.

```
L6: ADD RELATIONSHIP temProtocolo BETWEEN (MODULE Experimento-ENTITY Experimento)
AND (MODULE ProtocoloExperimental-ENTITY Protocolo) M:N TOTAL:PARTIAL ATTR=Atr1;
```

Exemplo 4: Anotação para criação de especialização.

```
L4: ADD SPECIALIZATION tipo FROM SUPERCLASS (MODULE Participante -
ENTITY Participante) TO SUBCLASS (MODULE Humano - ENTITY Humano) DISJOINT;
```

Exemplo 5: Anotação para criação de categorização.

```
L4: ADD CATEGORY tipo TO SUBCLASS (MODULE Participante - ENTITY Participante)
FROM SUPERCLASS (MODULE Humano - ENTITY Humano);
```

A sintaxe das anotações está definida na gramática (na forma de *Backus-Naur* estendida) mostrada na Figura 4.6.

```
anotação = rótulo_ligação, ":", (atributo | relacionamento |
                                 especialização | categorização), ";"

atributo = "ALTER", (módulo_entidade | módulo_relacionamento), modificação_de_atributo

modificação_de_atributo = adicionar_atributos | remover_atributos

módulo_entidade = "( M ", nome_módulo, " - E ", nome_entidade, ")"
módulo_relacionamento = "( M ", nome_módulo, " - R ", nome_relacionamento, ")"

adicionar_atributos = "ADD ATTRIBUTE", lista_de_atributos

lista_de_atributos = definição_atributo, {",", definição_atributo}
definição_atributo = valor_único | multivalorado
valor_único = tipo_atributo, [ "KEY" ]
multivalorado = "{", tipo_atributo, "}"
tipo_atributo = (simples | composto)
simples = nome_novo_atributo
composto = nome_novo_atributo, "(", tipo_atributo, {",", tipo_atributo}, ")"

remover_atributos = "DROP ATTRIBUTE", nome_atributo, {",",  nome_atributo}

relacionamento = "ADD RELATIONSHIP", nome_novo_relacionamento, "BETWEEN", módulo_entidade,
     "AND", módulo_entidade, cardinalidade, [participação], ["ATTR =", lista_de_atributos]
cardinalidade = "1:1" | "1:N" | "N:1" | "M:N"
participação = "PARTIAL:PARTIAL" | "PARTIAL:TOTAL" | "TOTAL:PARTIAL" | "TOTAL:TOTAL"

especialização = "ADD SPECIALIZATION", nome_especialização, "FROM SUPERCLASS", módulo_entidade,
                 "TO SUBCLASS", módulo_entidade, {",", módulo_entidade}
                 ["DISJOINT" | "OVERLAPPING"]

categorização = "ADD CATEGORY", nome_categoria, "TO SUBCLASS", módulo_entidade,
                 "FROM SUPERCLASS" módulo_entidade, {",", módulo_entidade}

rótulo_ligação = ? rótulo de uma ligação do DBFD ?
nome_módulo = ? nome de um módulo de dados da ligação anotada ?
nome_entidade = ? nome de um tipo de entidade do módulo associado ?
nome_relacionamento = ? nome de um tipo de relacionamento do módulo associado ?
nome_novo_relacionamento = ? nome de um tipo de relacionamento que não existe no esquema ?
nome_atributo = ? nome de um atributo do objeto de dados associado ?
nome_novo_atributo = ? nome de um atributo que ainda não existe no esquema ?
nome_especialização = ? nome de uma especialização no esquema de banco de dados ?
nome_categoria = ? nome de uma categoria no esquema de banco de dados ?
```

Figura 4.6: *Gramática das anotações dos DBFDs.*

Algumas explicações finais precisam ser feitas sobre as anotações:

1. Os atributos adicionados a tipos de entidades ou tipos de relacionamentos podem ser simples ou compostos, e de valor único ou multivalorados. Além disso, um atributo de valor único pode ser definido como um atributo chave.

2. Um tipo de relacionamento, especialização ou categorização definido por meio de uma anotação não precisa envolver tipos de entidades dos dois módulos de dados ligados pela relação sobre a qual a anotação foi feita. Portanto, uma anotação pode ser utilizada para relacionar tipos de entidades de um mesmo módulo.

3. O grau dos tipos de relacionamento criados por meio de anotações é sempre dois.

Podemos observar que os pontos de variações nos DBFDs serão diferentes dos definidos pelos diagramas de características de LPS. Em um DBFD, teremos dois níveis de variabilidade, ou seja, dois pontos de variação para criação do modelo conceitual. O primeiro nível é relação aos módulos de dados, que podem ser combinados de diversas formas para se criar modelos diferentes. O segundo é representado pelas anotações, que determinam a variabilidade em um nível mais granular, dos diagramas EER contidos nos módulos.

4.2 Projetando uma Família de Esquemas Conceituais de Banco de Dados

Os Diagramas de Características de Banco de Dados (DBFDs) introduzidos na Seção 4.1 podem ser usados para modelar a variabilidade que pode existir nos esquemas conceituais de banco de dados de organizações associadas de um mesmo domínio de aplicação. Um DBFD denota uma *família de esquemas conceituais de banco de dados* que compreende as possíveis variações de esquemas conceituais levantadas por um grupo de usuários para um dado domínio.

Para se criar uma família de esquemas conceituais de banco de dados, podem ser seguidas duas abordagens diferentes: na primeira abordagem, inicia-se definindo quais serão os módulos de dados e os diagramas EERs contidos neles, e então acrescentaremos as ligações e as anotações entre eles; na segunda abordagem, partiremos de um esquema conceitual completo e a partir disso dividimos em módulos de dados e definimos as ligações e anotações. Chamamos as duas abordagem de Abordagem de Construção de DBFD e Abordagem de Extração de DBFD, respectivamente. Nas próximas seções as duas abordagens serão explicadas detalhadamente, com a apresentação de um diagrama BPMN para cada uma delas [bpm]. Os diagramas BPMN são utilizados por apresentar uma notação gráfica simples que permite que organizações comuniquem a forma padrão de realização dos procedimentos internos.

4.2.1 Abordagem de Construção de DBFD

O diagrama BPMN da Figura 4.7 especifica os passos envolvidos na criação de uma família de esquemas conceituais de banco de dados segundo a abordagem de construção de DBFD, onde é criado um DBFD do zero, iniciando pela definição dos módulos de dados e dos diagramas EER contidos neles, para então criar as ligações e anotações entre os módulos de dados. Nessa abordagem, não se projeta um esquema conceitual completo do domínio de aplicação.

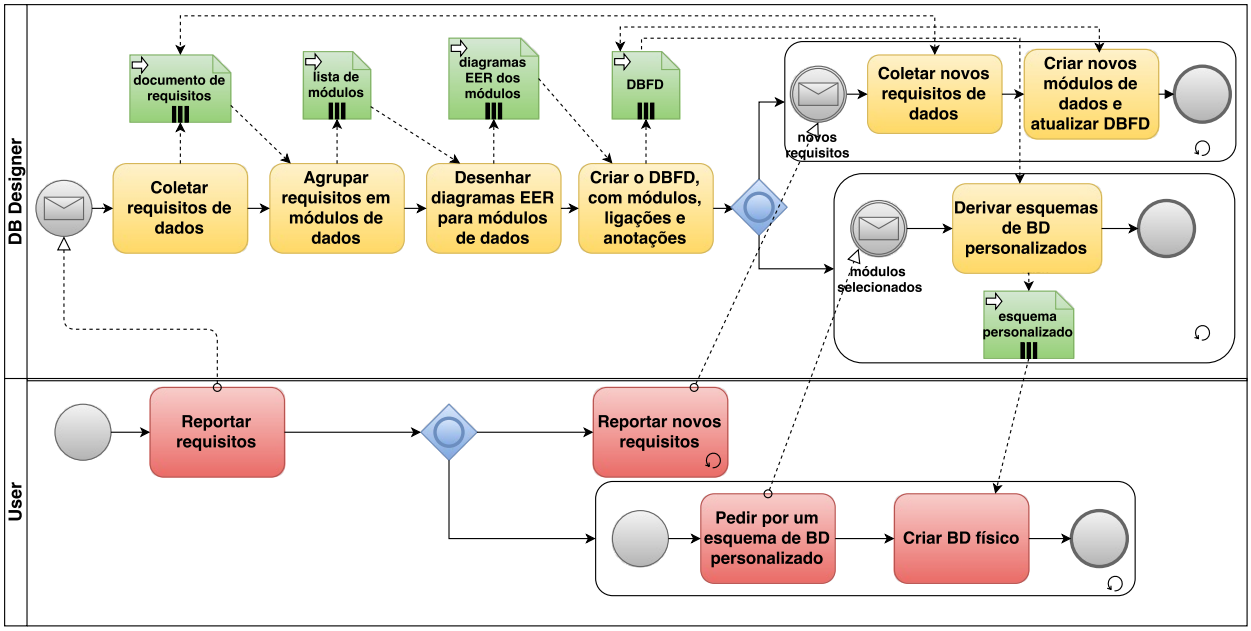

Figura 4.7: *Diagrama BPMN seguindo abordagem de construção de DBFD.*

A criação de uma família de esquemas conceituais de banco de dados na abordagem B começa com a identificação dos módulos de dados pertencentes ao domínio de aplicação considerado. A fim de fazer isso, primeiro o projetista do banco de dados deve coletar os requisitos de dados reportados pelos usuários de diferentes organizações do domínio de aplicação. Nesse estágio, outra tarefa primordial é identificar as semelhanças e as variações existentes nas organizações em relação aos seus requisitos de dados. Então, o projetista deve agrupar os requisitos de dados de acordo com os conceitos de dados a que eles se referem.

Cada grupo de requisitos de dados relacionados vai gerar um módulo de dados. Eventualmente, será necessário dividir um módulo de dados em mais módulos para separar requisitos que são comuns para todos os usuários de outros requisitos que são específicos para apenas alguns usuários. E para cada módulo de dados, um modelo conceitual EER deve ser desenhado. Neste ponto, é importante lembrar que os módulos de dados são partições do esquema de banco de dados, então eles não podem ter intersecções (ou seja, um objeto de dados que aparece em um dado módulo não pode aparecer em outro módulo do DBFD).

Depois da identificação e definição dos módulos de dados, o próximo passo é a criação do DBFD. Para criar o DBFD, o projetista deve primeiro identificar as dependências que existem entre os módulos de dados e então expressá-las por meio de ligações *consiste-em* e *de restrição* (definidas na Seção 4.1.2). Uma vez que as relações de um DBFD são definidas, elas podem ser enriquecidas com as anotações (como descrito na Seção 4.1.3).

Quando necessário, módulos de dados vazios podem ser introduzidos no DBFD para expressar modificações de dados que devem ser feitas em relação a outros módulos não-vazios a fim de satisfazer um dado conjunto de requisitos de dados que não geram novos tipos de entidades no esquema conceitual.

Cada módulo de dados que se refere aos requisitos de dados que são comuns para todos os usuários do domínio de aplicação deve ser definido como um módulo de dados base no DBFD. Um módulo de dados base define o que é comum em todos os esquemas conceituais da família e deve ser um descendente obrigatório do módulo raiz do DBFD [2]. O conjunto de módulos base é definido como o núcleo do DBFD. As composições opcionais ou alternativas dos módulos de dados no DBFD são projetadas para representar o que pode ser selecionado e combinado de diferentes maneiras para criar esquemas conceituais personalizados.

[2]Um descendente obrigatório do módulo raiz é um módulo que está conectado à raiz por meio de um caminho contendo apenas composições obrigatórias.

4.2.2 Abordagem de Extração de DBFD

O diagrama BPMN da Figura 4.8 especifica os passos envolvidos na criação de uma família de esquemas conceituais de banco de dados segundo a abordagem de extração de DBFD, partindo do diagrama EER completo para então extrair o DBFD, definindo os módulos de dados, as ligações e as anotações.

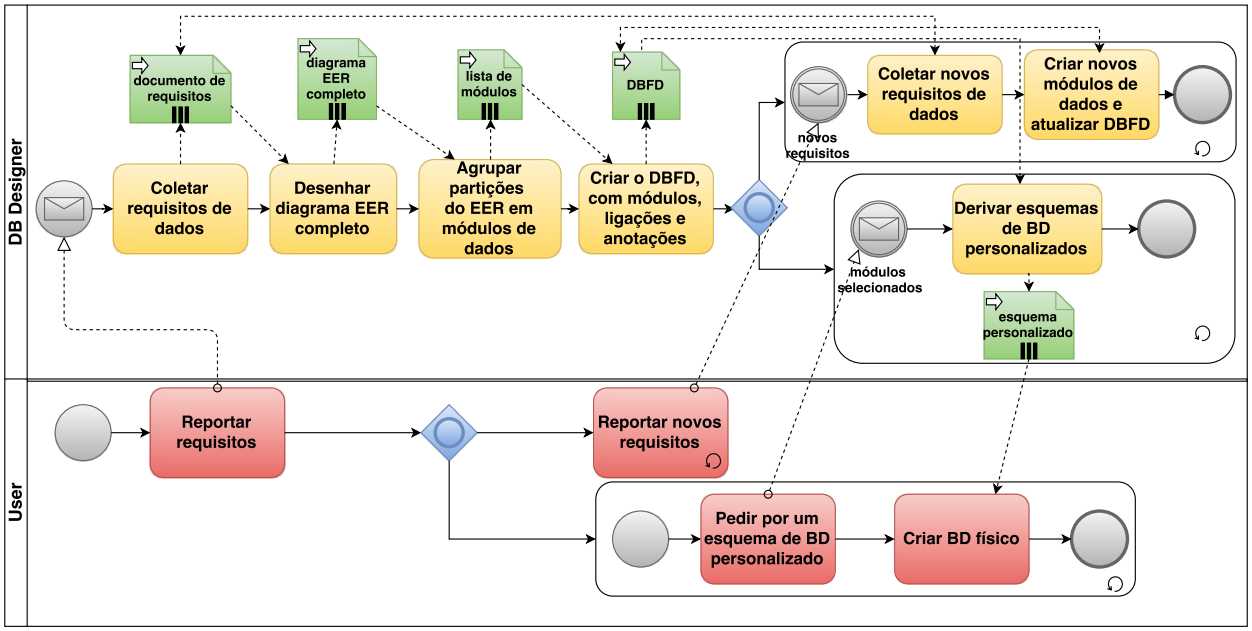

Figura 4.8: *Diagrama BPMN seguindo abordagem de extração de DBFD.*

A criação de uma família de esquemas conceituais de banco de dados pela abordagem A começa com a criação de um diagrama EER completo do domínio de aplicação considerado. A fim de fazer isso, primeiro o projetista do banco de dados deve coletar os requisitos de dados reportados pelos usuários de diferentes organizações do domínio de aplicação. Então, o projetista desenha um diagrama EER completo para o domínio.

O próximo passo consiste na identificação dos módulos de dados. O projetista deve agrupar os objetos de dados de acordo com os conceitos e requisitos de dados a que eles se referem. Considerando que o diagrama EER já está desenhado para todos os requisitos do domínio levantados, será necessário dividir o diagrama em diversas partições para criar os módulos de dados. Cada grupo de requisitos de dados relacionados vai gerar um módulo de dados. Nesse estágio, é primordial identificar as semelhanças e as variações existentes nas organizações do domínio em relação aos seus requisitos de dados, para se poder definir quais módulos serão obrigatórios e quais serão opcionais.

Ao fazer a divisão do diagrama EER em módulos de dados, existirão tipos de relacionamentos, especializações, categorizações e atributos que só farão sentido quando dois módulos de dados forem conectados. Esses tipos de elementos não deverão ser inseridos em nenhum dos módulos, pois serão criados através de anotações definidas sobre as ligações entre os módulos. Módulos de dados vazios podem ser introduzidos no DBFD para expressar modificações de dados que devem ser feitas em relação a outros módulos não-vazios a fim de satisfazer um dado conjunto de requisitos de dados que não geram novos tipos de entidades no esquema conceitual.

Depois da identificação e definição dos módulos de dados, o próximo passo é incluir as ligações *associativas* ou *restritivas* entre os módulos de dados. Uma vez que as ligações de um DBFD estejam definidas, elas podem ser enriquecidas com as anotações. Vale lembrar que as anotações poderão representar tipos de relacionamentos, especializações e categorizações entre tipos de entidades que ficaram em módulos de dados distintos quando o modelo conceitual completo foi particionado. Além disso, deve-se identificar quais atributos devem ser adicionados ou removidos somente quando houver a ligação entre dois módulos de dados.

4.2.3 Evolução e Derivação do DBFD

É importante notar que uma família inicial de esquemas de dados pode ser facilmente estendida toda vez em que um novo requisito de dados aparecer no domínio de aplicação. Para isso, é necessário apenas adicionar novos módulos de dados no DBFD da família de esquemas e conectá-los aos módulos pré-existentes, por meio de ligações e anotações.

Para derivar um esquema conceitual personalizado a partir do DBFD, é necessário selecionar os módulos de dados opcionais, que serão acrescentados ao núcleo do diagrama para compor o esquema específico. Um esquema conceitual personalizado pode ser mapeado para um esquema físico e então implementado para ser utilizado por uma organização específica do domínio de aplicação representado pelo DBFD.

Tanto a evolução do DBFD quanto a derivação são feitos da mesma maneira nas duas abordagens apresentadas. Portanto, as abordagens se diferenciam somente na forma de criação do DBFD inicial.

4.3 Exemplo Completo de Aplicação do Método

Após apresentar todos os conceitos existentes nos Diagramas de Características de Banco de Dados (DBFDs) e também os dois possíveis métodos que devem ser seguidos para se criar um DBFD, iremos apresentar um exemplo completo de aplicação do método seguindo a abordagem de construção de DBFD (Seção 4.2.1).

Inicialmente, vamos considerar o documento de requisitos do domínio de casas inteligentes (do inglês *smart homes*) apresentado posteriormente.

Casa Inteligente – Requisitos

Considere um sistema de automatização residencial para uma "casa inteligente" que contém o registro de usuários que podem modificar ou acessar as funções inteligentes da casa. Cada usuário tem um nome, código e senha. A casa contém cômodos, onde são encontrados diversos sensores e atuadores. Cada cômodo tem um código e um nome.
Os atuadores podem ser de diferentes tipos: climatizadores, interruptores de luz, sirenes e câmeras. No sistema, cada atuador tem uma descrição textual de sua localização no cômodo da casa e um *status*, que pode ser ligado ou desligado. Um interruptor de luz tem um tipo, que pode ser dimmer ou normal, e uma voltagem. O climatizador tem um modelo e uma potência.
Nesse nosso sistema, um climatizador deverá ser um ar condicionado ou um aquecedor. O ar condicionado tem funções, como ventilar ou refrigerar, enquanto que o aquecedor tem um *timer*. Cada sirene tem um tipo de som, um modelo e um nível de intensidade sonora. E a câmera tem um modelo, código de série e um *driver*.
Os sensores podem ser dos tipos: termostato, detector de presença ou medidor de luminosidade. Para todo sensor, o sistema de automatização mantém uma descrição da localização, o código, o tipo e o modelo. Periodicamente, o sistema coleta e armazena dados capturados por meio dos sensores. O termostato captura a temperatura em graus Celsius. O detector de presença registra a informação de presença. E o medidor de luminosidade mede a intensidade da luminosidade (baixa, média ou alta) especificando a informação do período do dia (manhã, tarde ou noite).
O sistema de automatização conhece as regras que podem existir para tornar a casa inteligente. A regra pode mudar qualquer configuração que o atuador tenha, por exemplo, ligar o ar condicionado na função ventilar. Uma regra é configurada por um usuário em uma interface amigável e é guardada como texto no banco de dados. Portanto, uma regra contém o código do atuador, o código do sensor e a lógica para a regra.
Na casa inteligente, o climatizador é ativado pelo termostato, ou seja, quando a temperatura aumenta, o ar condicionado é ligado, ou quando a temperatura diminui muito, é ligado o aquecedor. Portanto, quando temos um climatizador na casa, é necessário um termostato. Normalmente, uma casa tem somente um aquecedor ou ar condicionado, de acordo com o clima da região onde a casa se encontra.
Já o interruptor é ligado ou desligado quando detecta a mudança de luminosidade da casa. Ou seja, quando a casa está escura, o interruptor é ligado, e quando a casa está clara, o interruptor é desligado. Portanto, para ter um interruptor inteligente é necessário um sensor de luminosidade.
Os cômodos da casa podem ter dois atuadores que funcionam como dispositivos de segurança, que são as câmeras que podem ser acessadas online pelos usuários e as sirenes, que são disparadas quando for detectado a presença de uma pessoa e o sistema de alarme estiver ligado.

A partir do documento de requisitos anterior, o próximo passo é definir os módulos de dados e os diagramas EER contidos em cada módulo de dados. Foi criado um módulo de dados chamado `Painel de Controle`, que contém os dados de usuários do sistema de automatização, as regras criadas pelos usuários e os cômodos que essas regras modificam. Outros dois módulos definidos são os módulos `Sensor` e `Atuador`. E para cada módulo destes, foram definidos os possíveis filhos, que são `Detector de Presença`, `Sensor de Luminosidade` e `Termostato` para o `Sensor`, e `Sirene`, `Câmera`, `Interruptor` e `Climatizador` para o `Atuador`. E, ainda, foram criados dois módulos relacionados ao `Climatizador`, que são `Ar Condicionado` e `Aquecedor`.

Para cada módulo de dados definido, foram desenhados os diagramas EER correspondentes. As Figuras 4.9, 4.10, 4.11 e 4.12 mostram os diagramas criados para os módulos de dados na ordem listada anteriormente.

A partir da definição dos módulos de dados, são criadas as ligações associativas e restritivas entre os módulos, resultando em um DBFD. A Figura 4.13 mostra o DBFD criado. Note que os módulos de dados que compõem o núcleo do diagrama estão destacados dos demais módulos.

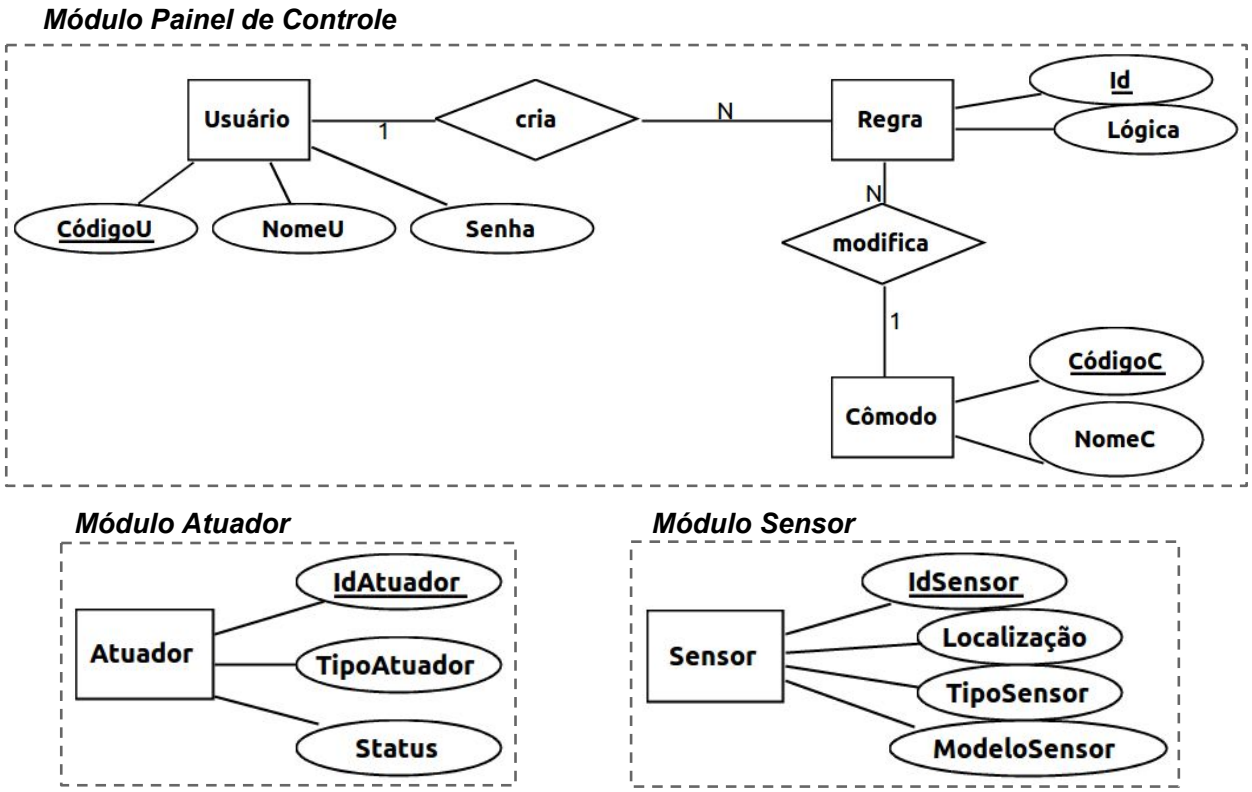

Figura 4.9: *Diagramas EER para Painel de Controle, Sensor e Atuador.*

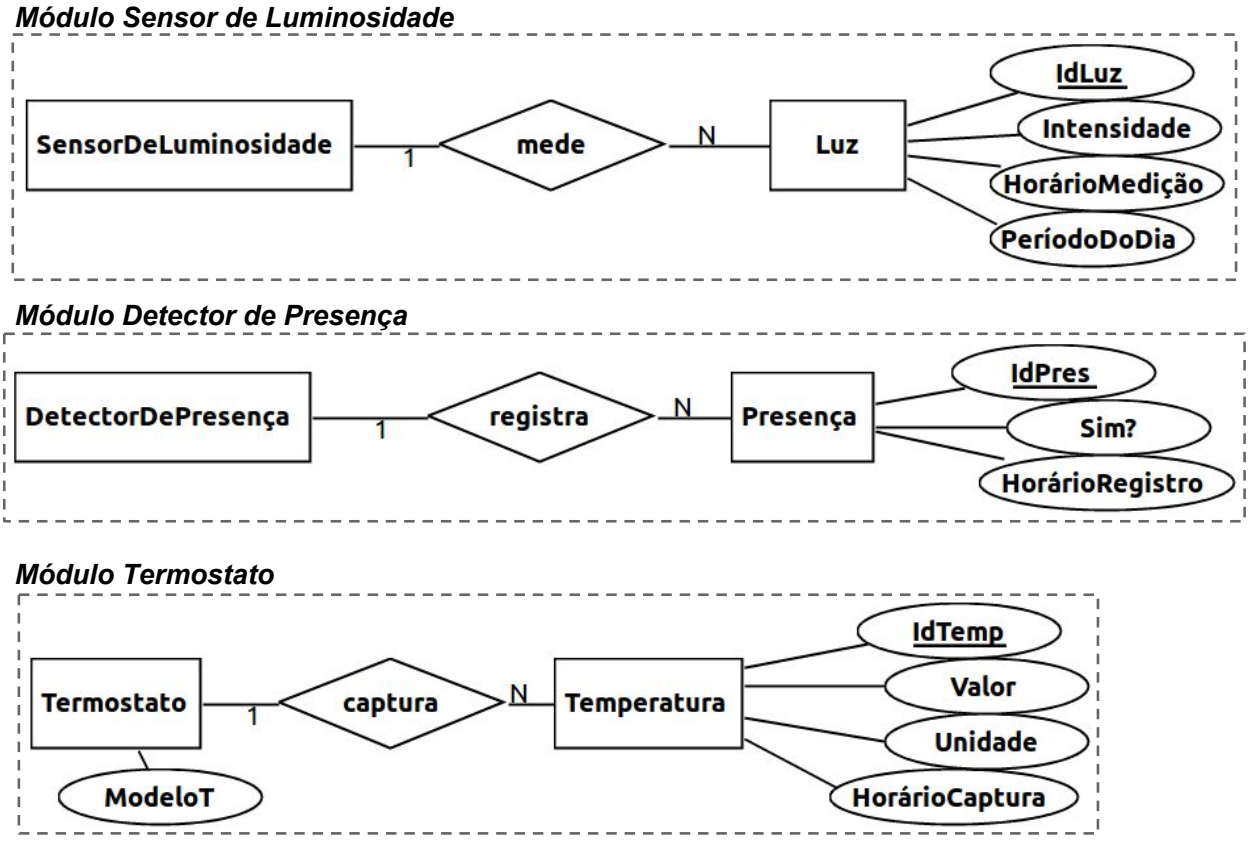

Figura 4.10: *Diagramas EER para os tipos de sensores.*

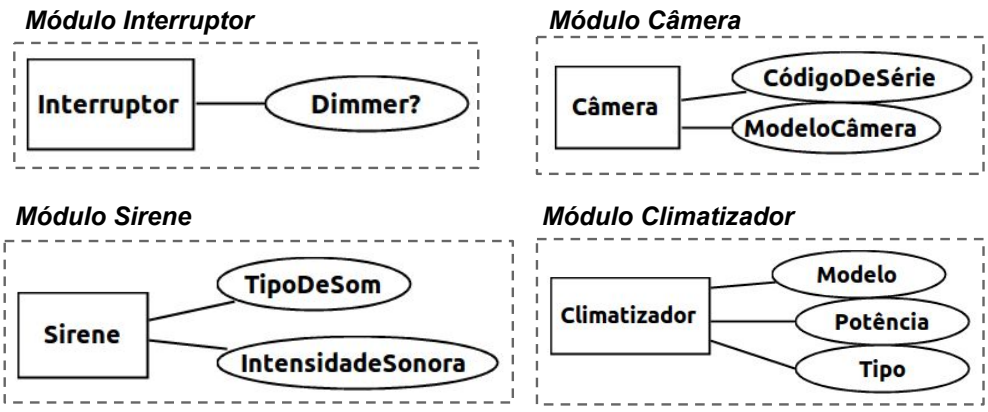

Figura 4.11: *Diagramas EER para os tipos de atuadores.*

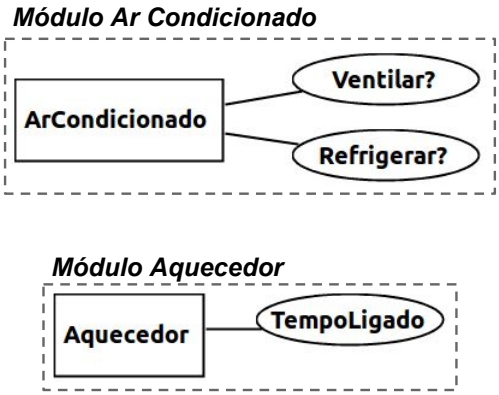

Figura 4.12: *Diagramas EER para tipos de climatizadores.*

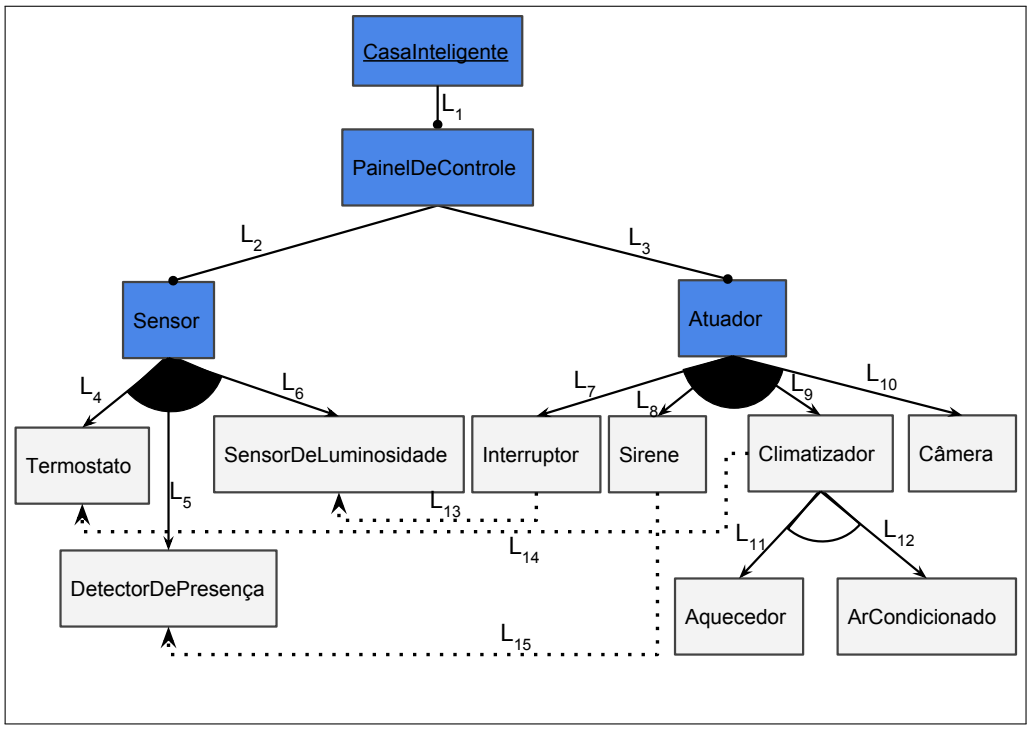

Figura 4.13: *DBFD para o domínio de casas inteligentes.*

Por último, é necessário escrever as anotações para as ligações entre os módulos de dados. As ligações e suas anotações são apresentadas posteriormente.

L_2 = ADD RELATIONSHIP contémSensor BETWEEN (M PainelDeControle - E Cômodo) AND (M Sensor - E Sensor) CARDINALITY=1:N ATTR=localizaçãoDoSensor;

L_3 = ADD RELATIONSHIP contémAtuador BETWEEN (M PainelDeControle - E Cômodo) AND (M Atuador - E Atuador) CARDINALITY=1:N ATTR=localizaçãoDoAtuador;

L_4 = ADD SPECIALIZATION tipoSensor BETWEEN SUPERCLASS (M Sensor - E Sensor) AND SUBCLASS (M Termostato - E Termostato);

L_5 = ADD SPECIALIZATION tipoSensor BETWEEN SUPERCLASS (M Sensor - E Sensor) AND SUBCLASS (M DetectorDePresença - E DetectorDePresença);

L_6 = ADD SPECIALIZATION tipoSensor BETWEEN SUPERCLASS (M Sensor - E Sensor) AND SUBCLASS (M SensorDeLuminosidade - E SensorDeLuminosidade);

L_7 = ADD SPECIALIZATION tipoAtuador BETWEEN SUPERCLASS (M Atuador - E Atuador) AND SUBCLASS (M Interruptor - E Interruptor);

L_8 = ADD SPECIALIZATION tipoAtuador BETWEEN SUPERCLASS (M Atuador - E Atuador) AND SUBCLASS (M Sirene - E Sirene);

L_9 = ADD SPECIALIZATION tipoAtuador BETWEEN SUPERCLASS (M Atuador - E Atuador) AND SUBCLASS (M Climatizador - E Climatizador);

L_{10} = ADD SPECIALIZATION tipoAtuador BETWEEN SUPERCLASS (M Atuador - E Atuador) AND SUBCLASS (M Câmera - E Câmera);

L_{11} = ADD SPECIALIZATION tipoClimatizador BETWEEN SUPERCLASS (M Climatizador - E Climatizador) AND SUBCLASS (M Aquecedor - E Aquecedor);

L_{12} = ADD SPECIALIZATION tipoClimatizador BETWEEN SUPERCLASS (M Climatizador - E Climatizador) AND SUBCLASS (M ArCondicionado - E ArCondicionado);

L_{13} = ADD RELATIONSHIP ligadoPor BETWEEN (M Interruptor - E Interruptor) AND (M SensorDeLuminosidade - E SensorDeLuminosidade) CARDINALITY=N:1;

L_{14} = ADD RELATIONSHIP ativadoPor BETWEEN (M Climatizador - E Climatizador) AND (M Termostato - E Termostato) CARDINALITY=N:1;

L_{15} = ADD RELATIONSHIP acionadoPor BETWEEN (M Sirene - E Sirene) AND (M DetectorDePresença - E DetectorDePresença) CARDINALITY=N:1;

Após os passos seguidos, construímos um DBFD, com seus módulos de dados, ligações e anotações, para o domínio de casas inteligentes. Esse DBFD poderá ser evoluído com a adição de novos módulos, ligações entre os novos módulos e módulos já existentes no diagrama, e anotações para as novas ligações. A partir desse DBFD, é possível gerar esquemas conceituais personalizados, selecionando os módulos opcionais que devem estar no esquema personalizado, além dos módulos base.

4.4 Considerações Finais do Capítulo

Este capítulo apresentou um novo método de modelagem de dados, que permite criar famílias de esquemas conceituais de banco de dados. O método é baseado em um novo tipo de diagrama, chamado de *Diagrama de Características de Banco de Dados* (DBFD). Com esse tipo de diagrama, é possível representar a variabilidade estrutural de dados que existe em alguns domínios de aplicação, como é o caso de grandes projetos de pesquisa e empresas multinacionais.

O próximo capítulo apresenta uma ferramenta de software, a DBFD Creator, que foi desenvolvida para facilitar a aplicação do método apresentado neste capítulo e, com isso, apoiar sua validação e avaliação.

Capítulo 5

Software Gerenciador de DBFDs

Diagramas de Características de Banco de Dados (DBFD) podem ser construídos e mantidos com o auxílio de uma ferramenta de software chamada **DBFD Creator**. A principal motivação para a construção dessa ferramenta neste trabalho de mestrado foi auxiliar usuários a reproduzir o passo-a-passo do método de modelagem introduzido no Capítulo 4. O DBFD Creator é uma ferramenta simples e de uso intuitivo, que guia os usuários na construção de DBFDs.

O código da ferramenta é aberto e está disponível no sítio Web https://bitbucket.org/laracmoraes/dbfd_creator para download. Quem se interessar pode criar extensões abertas da ferramenta e disponibilizá-las na plataforma *Bitbucket*.

O primeiro passo para construção da ferramenta de software foi realizar um estudo de ferramentas de código aberto que pudessem ser estendidas para construção de diagramas de características de banco de dados. Foram estudados dois tipos de ferramentas: ferramentas para Linhas de Produtos de Software e para modelagem conceitual de Banco de Dados. Esse estudo é apresentado na Seção 5.1. A Seção 5.2 apresenta as funcionalidades da ferramenta DBFD Creator, com alguns exemplos das telas da ferramenta.

5.1 Estudo de Ferramentas Abertas

Primeiramente, foram estudadas ferramentas de software para Linhas de Produtos de Software que pudessem ser adaptadas para a criação dos DBFDs. De acordo com [MS10], as ferramentas que tem como foco a modelagem de características são: Gears [gea], Pure::Variant [pur], FeatureIDE [TKB+14] e S.P.L.O.T. [spl]. Porém, as ferramentas Gears e Pure::Variant não são ferramentas de código aberto, e optou-se por se utilizar somente ferramentas de código aberto neste trabalho de mestrado.

As ferramentas FeatureIDE e S.P.L.O.T. são similares, pois são ferramentas de código aberto que foram criadas com o foco em desenvolvimento de famílias de produtos de software para uso em projetos acadêmicos. Existem duas grandes diferenças entre as duas ferramentas. A primeira delas é que a ferramenta S.P.L.O.T. é uma ferramenta online, enquanto que o FeatureIDE é um plugin do Eclipse, sendo necessário fazer o *download* e instalação tanto do plugin quanto da plataforma Eclipse para sua utilização. E a outra grande diferença é que o plugin FeatureIDE possibilita criar os diagramas de características de forma gráfica, enquanto que na ferramenta S.P.L.O.T. é necessário criar cada característica, relação e restrição com campos no formato de texto.

A intenção ao utilizar uma ferramenta de linhas de produtos de software era para criar o diagrama de características de base para a criação de um DBFD. Portanto, a ferramenta escolhida foi um *plugin* do Eclipse – o FeatureIDE [TKB+14], que é simples, intuitivo e possibilita criar os diagramas de forma visual e gráfica. E, além disso, o FeatureIDE é baseado no modelo de diagrama de características do FeatuRSEB[GFd98], mesmo método que embasou o novo tipo de diagrama proposto neste trabalho.

Em relação à criação de diagramas EER (Entidade Relacionamento Estendido), houve certa dificuldade para encontrar uma ferramenta de código aberto que suportasse modelagem conceitual

pura. Entre as ferramentas estudadas estão: Lucid Chart [luc], Draw IO [dra], ERDPlus [erd] e EERCase [EER]. As ferramentas Lucid Chart e Draw IO são ferramentas de propósito geral, que possibilitam a criação de diversos tipos de diagramas, como EER, UML, BPMN, entre outros.

A ferramenta ERDPlus permite criar diagramas ER, esquemas relacionais, esquemas estrela e também declarações SQL, ou seja, é possível criar desde o modelo conceitual até o físico. Já a ferramenta EERCase foi desenvolvida exclusivamente para possibilitar a criação de diagramas EER. A ferramenta escolhida foi o EERCase, por ser simples, tratar especificamente de modelos conceituais e possibilitar a criação de diagramas que seguem a notação utilizada neste trabalho de mestrado, que é a notação apresentada por Navathe e Elmasri [EN10].

5.2 Funcionalidades do DBFD Creator

A ferramenta DBFD Creator foi desenvolvida em formato de aplicação web, utilizando o arcabouço *Django* [dja] com a linguagem *Python* [pyt], e teve como base de implementação a reutilização das ferramentas de código aberto FeatureIDE [TKB+14] e EERCase [EER]. A Figura 5.1 mostra a página inicial da ferramenta DBFD Creator, onde há uma explicação simples do método de modelagem conceitual proposto neste trabalho de mestrado e também um passo-a-passo para reproduzir o método na ferramenta.

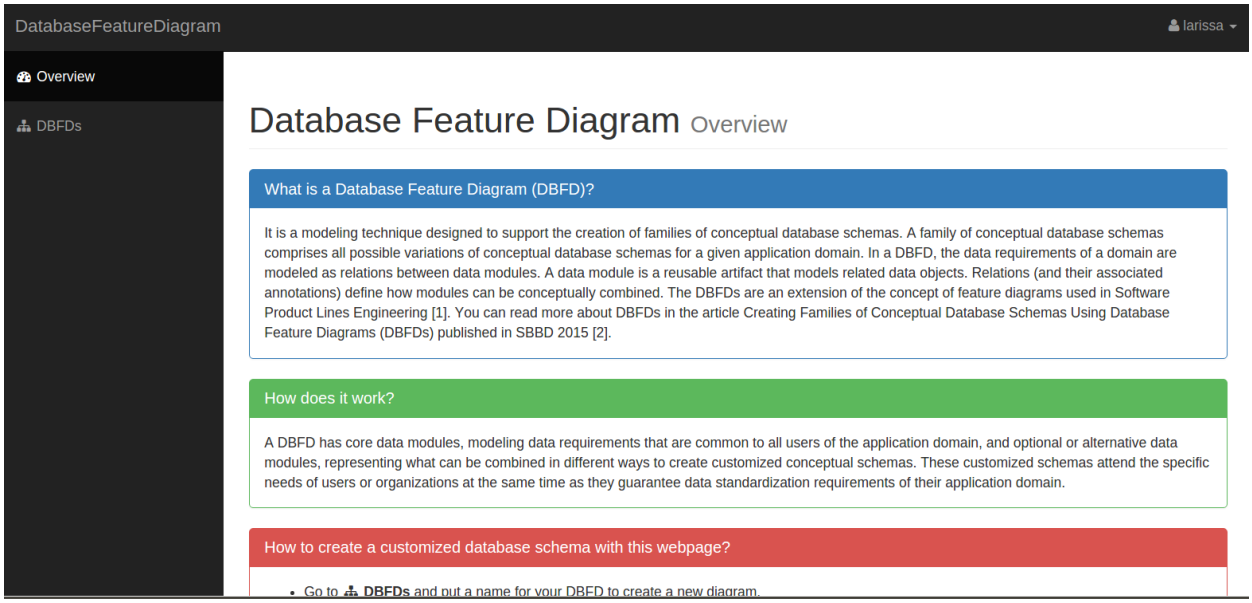

Figura 5.1: *Página Inicial do DBFD Creator.*

As funcionalidades existentes no DBFD Creator são:

1. Importar um diagrama de características para adicionar os módulos de dados, ligações e restrições;
2. Adicionar novos módulos de dados, determinando o tipo de ligação com módulos de dados existentes;
3. Adicionar novas ligações restritivas;
4. Remover módulos de dados existentes e suas respectivas ligações;
5. Importar um diagrama EER para um módulo de dados;
6. Adicionar ou remover uma anotação para uma ligação;
7. Selecionar uma configuração em um DBFD criado;

8. Gerar um diagrama EER completo para a configuração selecionada;
9. Gerar um *script* SQL inicial para a configuração selecionada;
10. Gerar um *script* SQL de atualização da configuração atual para a configuração salva anteriormente.

Primeiramente, é necessário dar um nome para a família de esquemas conceituais que está sendo criada na ferramenta. O segundo passo para construção do DBFD é criar os módulos de dados e as ligações existentes entre eles. Para isso, existem duas possibilidades: adicionar os módulos de dados diretamente no DBFD Creator ou importar um diagrama de características inicial.

Para esse trabalho de mestrado, o *plugin* FeatureIDE foi utilizado para se criar um diagrama de características inicial. Após criar o diagrama de características no *plugin*, o mesmo pode ser salvo no formato Velvet, que é um formato textual simples, para ser importado na ferramenta DBFD Creator.

Após importar o diagrama, os módulos de dados e ligações estarão criados no DBFD Creator. Vale ressaltar que não é obrigatória a utilização do *plugin* FeatureIDE, ou seja, além de iniciar o processo de criação do DBFD importando o diagrama de características criado no *plugin*, existe a possibilidade de se criar o DBFD desde o início no DBFD Creator, adicionando os módulos e as ligações entre eles diretamente na ferramenta.

Portanto, na tela apresentada na Figura 5.2 poderão ser adicionadas módulos de dados e ligações associativas ou restritivas entre os módulos de dados. Outra tarefa que poderá ser executada é a remoção de módulos e ligações. Essa tarefa só é permitida enquanto não for criada nenhuma configuração do DBFD. A Figura 5.2 mostra essas funcionalidades na tela de construção do DBFD.

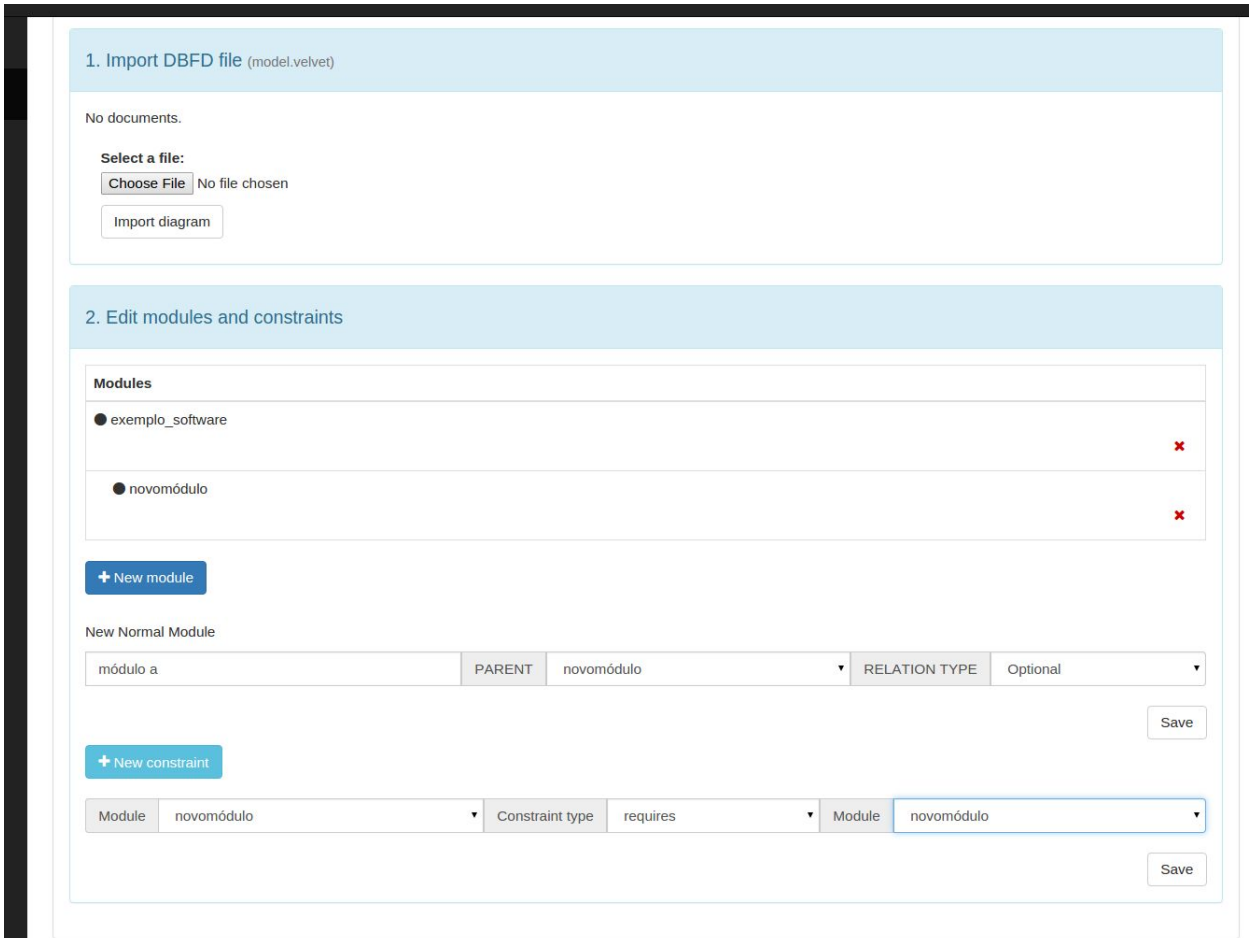

Figura 5.2: *Tela de criação dos módulos e ligações.*

O próximo passo na ferramenta é de importar um diagrama EER para cada módulo de dados criado, clicando no ícone de *upload*, como mostrado na Figura 5.3. A ferramenta EERCase deve ser usada para construir os diagramas EER contidos nos módulos de dados. Cada módulo de dados existente no DBFD deve ter um arquivo com extensão " .eer" associado, que contém o esquema conceitual referente ao módulo. A única exceção ocorre para módulos de dados vazios ou raiz.

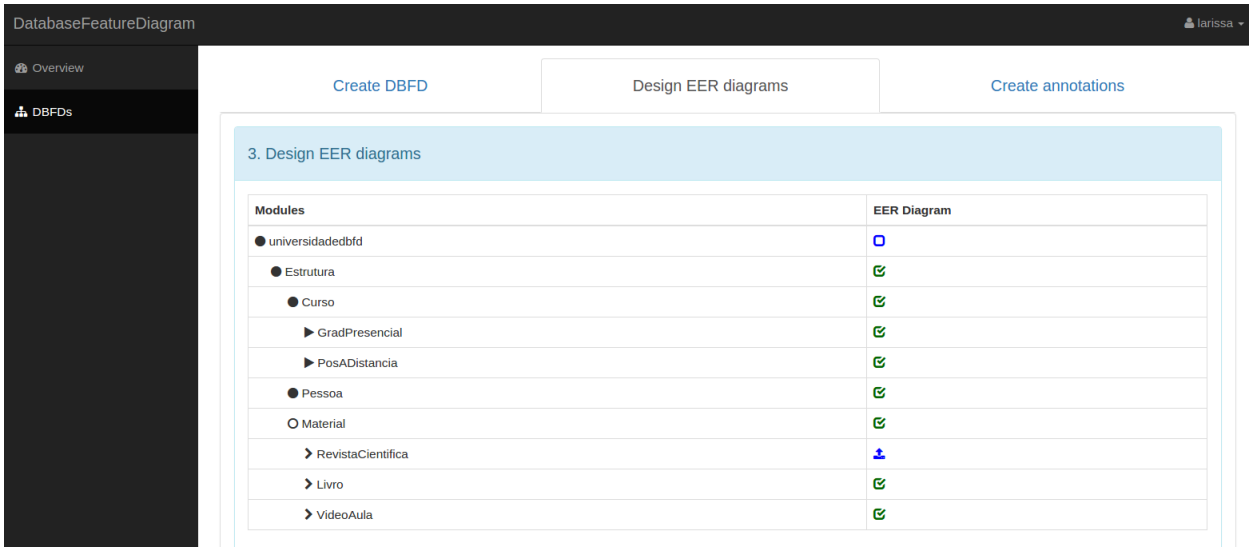

Figura 5.3: *Tela de upload do diagrama EER.*

Após importar um diagrama EER para cada módulo de dados, é possível iniciar a criação das anotações, como mostrado nas Figuras 5.4 e 5.5. Para cada ligação existente entre dois módulos no DBFD, podem ser escritas uma ou mais anotações com a função de: adicionar ou remover um atributo de um tipo de entidade ou de relacionamento, criar um novo tipo de relacionamento entre tipos de entidades e adicionar uma especialização ou uma categorização entre dois tipos de entidades.

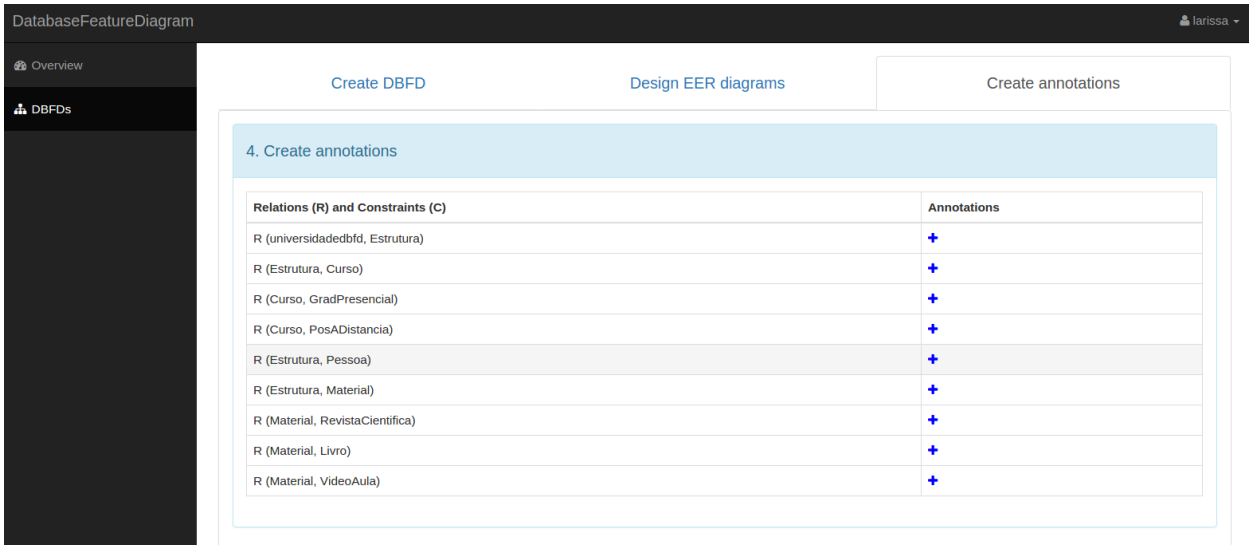

Figura 5.4: *Tela de resumo das anotações.*

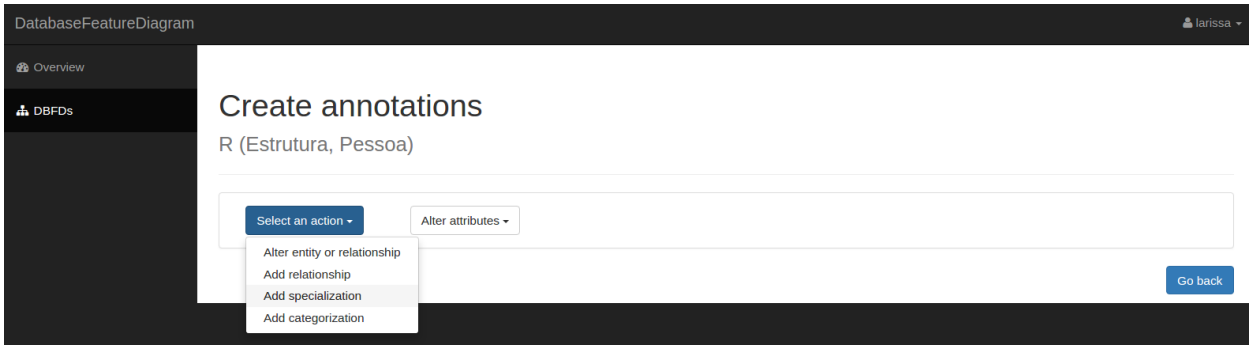

Figura 5.5: *Tela de criar anotações.*

Portanto, resumidamente, os primeiros passos na ferramenta DBFD Creator são:

1. Adicionar e/ou importar os módulos de dados e ligações entre os módulos de dados;
2. Associar um diagrama EER para cada módulo de dados existente, exceto no caso do módulo raiz ou de um módulo vazio;
3. Escrever anotações para as ligações do diagrama.

Após finalizar os três passos anteriores, o DBFD já está criado, ou seja, foi construída uma família de esquemas conceituais de banco de dados. Nesse momento, o botão *Create Configuration* é habilitado para que o usuário possa ir para a próxima etapa.

A partir desse momento, podem ser criadas configurações para o diagrama, gerando esquemas conceituais ou *scripts* SQL personalizados. Uma configuração pode ser entendida como uma derivação da família de esquemas para um esquema personalizado específico. Para não gerar inconsistências, a partir do momento que uma configuração é criada sobre um diagrama, esse diagrama não poderá mais ser alterado. Quando for necessário alterar o diagrama, é criada uma cópia do diagrama e as alterações acontecerão sobre essa cópia, formando uma nova família de esquemas. A Figura 5.6 mostra um exemplo da página de criação de uma configuração do DBFD, onde existe a opção de se mostrar a figura com o esquema conceitual completo personalizado ou gerar o *script* SQL de criação ou atualização para implementação em um SGBD (Sistema Gerenciador de Banco de Dados) gerando o banco de dados físico.

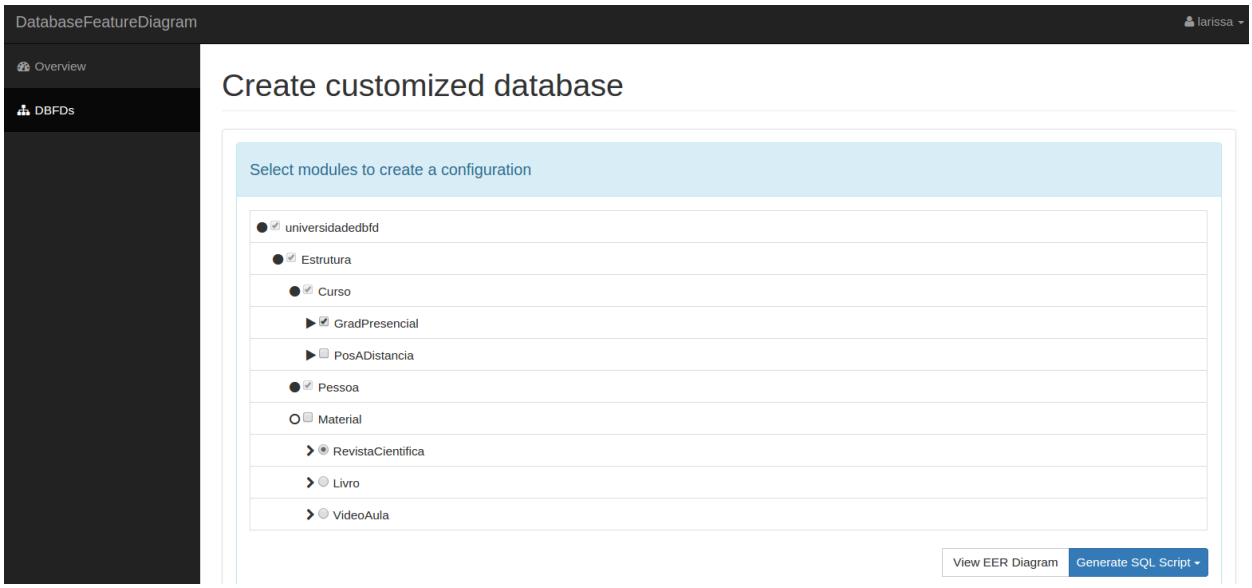

Figura 5.6: *Página do DBFD Creator de criação de uma configuração.*

5.3 Considerações Finais do Capítulo

Foi desenvolvida uma ferramenta Web simples e intuitiva, o DBFD Creator, que possibilita a criação e manutenção de diagramas que representam famílias de esquemas conceituais de bancos de dados e também a geração de configurações dessas famílias, por meio da seleção dos módulos de dados desejados.

Vale ressaltar que a ferramenta de software foi criada inicialmente com funcionalidades simples, para auxiliar na validação do método de modelagem proposto no trabalho. Portanto, outras versões podem ser criadas para enriquecer a ferramenta com novas funcionalidades.

Algumas melhorias que podem ser feitas na ferramenta são: desenvolver uma interface para criação dos módulos de dados e ligações diretamente na ferramenta; disponibilizar a visualização do diagrama EER contido em cada módulo de dados e listar as anotações de cada ligação na página que contém todas as ligações.

Para a etapa de validação deste trabalho de mestrado, foi criado um manual completo da DBFD Creator, que pode ser encontrado no Apêndice A.

No próximo capítulo será apresentado um estudo de caso que foi realizado em um domínio de aplicação real: o domínio de experimentos de neurociência.

Capítulo 6

Estudo de Caso

A fim de avaliar o método proposto no Capítulo 4, foi realizado um estudo de caso que visa criar esquemas conceituais de banco de dados para um domínio do mundo real: dados experimentais de neurociência. Os dados do estudo de caso foram coletados no escopo das atividades do *Centro de Pesquisa, Inovação e Disseminação em Neuromatemática* (NeuroMat) [neu].

O Neuromat tem como principal objetivo integrar modelagem matemática com pesquisa básica e aplicada na fronteira da neurociência. Do NeuroMat participam diversos laboratórios de pesquisa em neurociência que coletam dados a partir de diferentes tipos de experimentos de eletrofisiologia e neuroimagem, envolvendo tanto humanos, quanto outros tipos de animais. Matemáticos, estatísticos, cientistas da computação e neurocientistas, do Instituto de Matemática e Estatística (IME) da Universidade de São Paulo (USP) e de outras instituições brasileiras e estrangeiras associadas ao projeto, trabalham juntos na construção de novos modelos matemáticos e ferramentas de software que ajudem a avançar a compreensão que se tem hoje do funcionamento do cérebro e o tratamento de suas patologias.

6.1 Descrição do Domínio de Neurociência

A neurociência corresponde ao estudo científico do sistema nervoso, visando compreender sua estrutura, seu desenvolvimento, seu funcionamento, sua evolução, a relação entre o comportamento e a mente, e também suas alterações. Essa área tem apresentado um crescente desenvolvimento, envolvendo um número cada vez maior de cientistas interessados em realizar experimentos que os auxiliem a melhor entender o funcionamento do cérebro humano. Os experimentos em neurociência investigam a correlação entre os sistemas cerebrais e a atividade mental alterada ou saudável.

No domínio de neurociência existem diferentes tipos de experimentos realizados, como, por exemplo, os comportamentais, cognitivos, eletrofisiológicos e de neuroimagem. Os experimentos eletrofisiológicos e de neuroimagem, em particular, sempre envolvem a coleta de dados em formato digital.

Experimentos eletrofisiológicos envolvem o estudo de potenciais elétricos gerados pelo cérebro por meio da coleta de sinais da atividade cerebral ou muscular de humanos ou outros animais submetidos a condições experimentais específicas (como, por exemplo, a exposição a um estímulo sonoro ou a execução de uma tarefa motora). Exemplos de experimentos de eletrofisiologia são a Eletroencefalografia (EEG), a Estimulação Magnética Transcraniana (TMS) e a Eletromiografia (EMG).

Experimentos de neuroimagem consistem na utilização de aparelhos compostos por um magneto supercondutor e bobinas, que transmitem ondas de radiofrequência, formando juntos um campo magnético intenso, ativando os átomos de hidrogênio. Quando desligados os pulsos de rádiofrequência, esses átomos liberam energia que é captada por antenas receptoras e convertida em sinais digitais que são traduzidos em imagens estruturais de alta definição do cérebro. Exemplos desse tipo de experimento são a Ressonância Magnética (MRI) e Ressonância Magnética Funcional (fMRI) [PFH+08].

Todas as definições sobre um experimento, incluindo definições sobre os objetivos, a descrição dos grupos de sujeitos que serão testados, as condições experimentais às quais os grupos serão submetidos e os tipos de coletas de dados que serão realizados, são chamadas pelos cientistas de *protocolo experimental* ou *desenho do experimento*. Cada tipo de experimento envolve uma preparação específica para sua realização, como, por exemplo, configurações no equipamento de coleta do sinal e a colocação de sensores (eletrodos) em locais previamente definidos do corpo dos sujeitos do experimento. As coletas de dados podem se referir tanto aos sinais ou imagens capturados dos sujeitos e registrados em meio digital com o auxílio de equipamentos como o de EEG ou RMI, quanto a medições ou anotações manuais sobre o comportamento observado dos sujeitos. Essas informações são importantes para que um cientista possa fazer uma análise correta dos dados coletados a partir dos experimentos.

Além disso, há outras informações "ortogonais" à realização do experimento que também são muito importantes para definir a qualidade dos dados coletados. Exemplos disso são as informações sobre o laboratório onde os dados foram coletados, sobre o cientista responsável pelo experimento, a data de realização do experimento e até mesmo publicações ou outros resultados decorrentes do estudo dos dados coletados.

No contexto de neurociência, tanto as informações do protocolo experimental quanto as informações ortogonais à realização dos experimentos variam de laboratório para laboratório de pesquisa. Portanto, para que os laboratórios consigam trocar informações a respeito de seus experimentos e também possam compartilhar seus dados com a comunidade científica, é necessário que haja uma estrutura padronizada para a representação e o armazenamento dos dados.

Porém, os cientistas frequentemente armazenam digitalmente os dados de seus experimentos como arquivos comuns (de texto ou planilhas), sem nenhum formato padrão, mantidos no sistema de arquivos de um computador. Essa estrutura de armazenamento dificulta a manutenção, a recuperação (busca), o compartilhamento e o reúso dos dados, principalmente quando o volume de dados coletados começa a crescer.

A representação e o armazenamento digital de dados de experimentos em neurociência impõe diversos desafios. O principal deles é a grande variabilidade que pode existir nas estruturas dos protocolos experimentais. Além disso, não há um consenso na comunidade científica sobre quais são os tipos de dados que são indispensáveis para se reportar um experimento em neurociência. Ainda não existem propostas que lidem apropiadamente com esses desafios e que tenham se tornado uma referência para a comunidade de neurociência.

6.2 Representação e Armazenamento de Dados em Neurociência

Na última década, a comunidade de neurociência despertou para a importância da criação e manutenção de bancos de dados que apoiem as pesquisas conduzidas nessa área. Consórcios e projetos foram criados [Nem, Car], com o objetivo de desenvolver padrões para a representação de dados e ontologias no domínio da neurociência, e também repositórios que integrem dados provenientes de fontes heterogêneas e os disponibilizem para a comunidade científica (em algumas vezes de forma aberta, em outras vezes por meio de acesso controlado).

Apesar da crescente quantidade de iniciativas relacionadas à coleta e organização de dados em neurociência, ainda há muitas limitações nas soluções que vêm sendo empregadas atualmente. Uma avaliação informal feita sobre bancos de dados bastante conhecidos na comunidade de neurociência (como os listados na Wikipédia [wik]) identificou algumas deficiências frequentes.

Além de muitas vezes não serem de acesso público e não apresentarem quantidade significativa de dados, esses bancos de dados são muito específicos para determinados tipos de paradigmas de experimentos, como, por exemplo, cognição e memória, e com isso não têm a flexibilidade necessária para acomodar experimentos mais variados. E, também, a maioria desses bancos de dados funcionam como uma "federação" de conjuntos heterogêneos de dados, ou seja, eles agrupam dados coletados em diferentes projetos – com qualidade variável e, muitas vezes, armazenados em estruturas diferentes – e não provêm uma visão unificada dos dados. Isso dificulta o reúso dos dados para a descoberta

automática de novos conhecimentos.

Portanto, mesmo considerando os avanços já alcançados, existe ainda um grande desafio que a área de neurociência precisa transpor para que possa se beneficiar de maneira mais plena do uso de sistemas computacionais: estabelecer padrões para a representação e armazenamento de dados dados coletados ou gerados em experimentos, bem como para os dados de proveniência desses experimentos. Esses padrões possibilitariam a interoperalibidade nas ferramentas de software desenvolvidas para o domínio, facilitariam a criação e manutenção dos repositórios e o compartilhamento e reúso dos dados.

6.3 Aplicando os DBFDs na Modelagem de Dados Experimentais de Neurociência

Para levantar os requisitos do domínio de dados experimentais de neurociência, foram realizadas reuniões com os pesquisadores e a equipe de desenvolvimento de software do Neuromat [neu], por meio das quais foi possível definir os componentes existentes nos protocolos experimentais de neurociência e também caracterizar as coletas de dados em experimentos envolvendo Eletroencefalografia (EEG), Eletromiografia (EMG) e Estimulação Transcraniana Magnética (TMS). Os requisitos de dados são apresentados na Seção 6.3.1.

6.3.1 Descrição dos Requisitos de Dados

No contexto do pesquisa científica, todo projeto de pesquisa tem um título, um resumo e palavras chave que o define. Um projeto é financiado por uma ou mais agências financiadoras de pesquisa, que tem um nome e a cobertura da agência, que pode ser estadual, nacional ou internacional. Todo projeto financiado tem um número de processo para consultá-lo na agência de fomento. Um projeto de pesquisa pertence a uma área de pesquisa, com nome e descrição.

Uma universidade é dividida em departamentos, e cada departamento tem pessoas associadas, que podem ser do tipo aluno, professor ou pesquisador. E ainda, um departamento pode ter diversos laboratórios de pesquisa, com um professor responsável. Um laboratório de pesquisa pode hospedar vários projetos de pesquisa. Todos os projetos de pesquisa têm um coordenador e pesquisadores que são associados a um departamento.

Um projeto de pesquisa realiza experimentos que são projetados e conduzidos pelos pesquisadores. Os experimentos são identificados por um título, a descrição, data inicial, data final e o nome do pesquisador responsável. Os experimentos seguem um paradigma, com nome e definição, que se refere aos métodos de pesquisa utilizados pelos cientistas para alcançar seus objetivos experimentais. Experimentos bem sucedidos têm seus resultados disseminados para a comunidade científica na forma de publicação como artigos, relatórios, padrões, etc.

Experimentos em neurociência são aplicados sobre grupos de sujeitos (por exemplo, participantes voluntários dos estudos). Os sujeitos têm idade, gênero e podem ser diagnosticados com doenças que possuem código internacional (CID), nome e descrição. Sujeitos podem ser humanos ou animais não-humanos. Alguns laboratórios realizam experimentos envolvendo somente sujeitos humanos, enquanto outros realizam experimentos envolvendo somente não-humanos, e há também os laboratórios que realizam experimento com os dois tipos de sujeitos. Cada um desses dois tipos de sujeitos tem seus próprios atributos. Por exemplo, a língua nativa é um atributo exclusivo de sujeitos humanos, enquanto que espécie, gênero e família é a forma de classificar não-humanos. Os sujeitos humanos devem assinar um termo de consentimento para poder participar de um experimento.

Cada grupo de sujeitos de um experimento é associado a um protocolo experimental, que pode ser visto como uma "receita" para executar um experimento. Um protocolo experimental pode ser modelado como um fluxo de trabalho (do inglês, *workflow*), onde cada passo tem uma descrição, o número de repetições e a ordem que aquele passo é executado no protocolo.

Um passo corresponde a uma tarefa a ser executada pelo sujeito, à apresentação de um estímulo, a uma pausa dada pelo pesquisador ao sujeito, ao início de um bloco de passos ou à coleta de dados.

Um estímulo pode ser um arquivo de mídia (imagem, vídeo ou áudio) ou uma estimulação magnética transcraniana (TMS, do inglês *transcranial magnetic stimulation*). Existem diferentes tipos de dados que podem ser coletados nos experimentos de neurociência, como: preenchimento de questionários, sinais de eletroencefalograma (EEG) e eletromiografia (EMG), imagens de ressonância magnética (RMI), dados comportamentais, etc.

O estímulo TMS é usado para estimular pequenas regiões do cérebro. Durante um procedimento de TMS, um gerador de campo magnético, chamado de bobina, é posicionado próximo à cabeça do paciente, produzindo correntes elétricas por indução eletromagnética. A posição da bobina indica o ponto de atuação do estímulo. Uma bobina tem uma configuração, com as informações de tipo do pulso, intensidade do estímulo, intervalo entre pulsos e um limiar motor. O modelo da bobina é descrito por material, tamanho e formato. Uma restrição a ser considerada é a de que quando houver estímulo TMS é necessário uma coleta de dados do tipo EMG para captar as alterações musculares geradas pelos estímulos.

Uma coleta de dados produz arquivos de coleta, com descrição e nome, que serão utilizados nas análises dos experimentos. A coleta de dados mais simples é feita com questionários, que contêm um código e um nome e são compostos por perguntas, com enunciado, tipo de resposta e opções de resposta. Na coleta de dados do tipo Neuroimagem, o sujeito é colocado em uma máquina de ressonância que gera várias imagens em sequência. Além do arquivo de imagem de RMI é importante saber a informação da posição em que deve estar o sujeito durante o experimento e também o tempo de scan da ressonância, além das configurações da máquina utilizada.

Os experimentos de EEG e EMG são similares pelo fato de registrarem atividades elétricas, do cérebro e das membranas de células musculares, respectivamente, e por isso são classificados como experimentos de eletrofisiologia. Para o procedimento de EEG, são colocados eletrodos em determinados pontos do couro cabeludo de um sujeito, para medir as flutuações de tensão resultantes da corrente iônica dentro dos neurônios do cérebro. Já no procedimento de EMG, são fixados eletrodos, que podem ser de agulha ou superficiais, na musculatura e esses eletrodos enviam sinais ao eletromiógrafo de acordo com as trocas iônicas de nível celular.

Para dados tanto de EEG quanto de EMG, temos as informações dos filtros e amplificadores usados no equipamento de aquisição dos sinais. Como as diferenças de potencial flutuam em função do tempo, os sinais de EEG registrados têm uma certa largura de banda. Por isso, os canais de gravação de EEG são equipados com filtros passa-baixas e passa-altas, de modo que a resposta da frequência possa ser restrita à faixa de frequência de interesse. As informações que devem ser armazenadas sobre o filtro são: tipo do filtro, corte de passa-altas (em Hz), corte de passa-baixas (em Hz) e ordem, que é o atraso máximo usado para criar a amostra de saída.

A amplitude do sinal de EEG e EMG é de apenas alguns micro-volts. Portanto, o sinal precisa ser amplificado milhares de vezes antes de ser digitalizado. O amplificador tem as seguintes informações: ganho, número de canais, taxa de rejeição do modo comum, impedância da entrada e a unidade da impedância.

A configuração do eletrodo usado tanto no EEG quanto no EMG envolve o modelo do eletrodo e a localização do mesmo. O modelo do eletrodo contém as seguintes informações para ambos: tipo, material, usabilidade, impedância e marca. A localização para ambos os experimentos contém uma posição de referência e uma imagem de referência. A localização de eletrodos de EEG se difere por ter uma coordenada X e uma coordenada Y, o modelo da toca de eletrodos utilizada, o sistema de localização da toca e o padrão do indíce dos canais de eletrodos. Já a localização de eletrodos de EMG contém o nome, o lado e a subdivisão do músculo.

6.3.2 Módulos de Dados, DBFD e Anotações

Seguindo o passo-a-passo do DBFD proposto na abordagem de construção de DBFD do Capítulo 4, começamos identificando os módulos de dados a partir da descrição dos requisitos de dados apresentada na Seção 6.3.1. A partir da definição de quais seriam os módulos de dados existentes, foram criados os modelos conceituais EER que são mostrados nas Figuras 6.1 a 6.6.

Para a estrutura organizacional de pesquisa, três módulos de dados foram definidos – `Projeto`

`de Pesquisa`, `Laboratório de Pesquisa` e `Estrutura Departamento` – mostrados na Figura 6.1. Os três módulos serão opcionais pelo fato de que não é todo grupo de pesquisa que deseja armazenar nos bancos de dados as informações sobre os projetos de pesquisa, os laboratórios e as pessoas associadas ao projeto. Além disso, dividimos em três módulos de dados para possibilitar que sejam armazenadas somente informações sobre os projetos de pesquisa e a partir dessa informações somente dados sobre os laboratórios de pesquisa ou sobre a estrutura da universidade com os dados dos pesquisadores envolvidos no projeto. Essa divisão de módulos flexibiliza o modelo conceitual de forma a acomodar os dados de diferentes projetos de pesquisa.

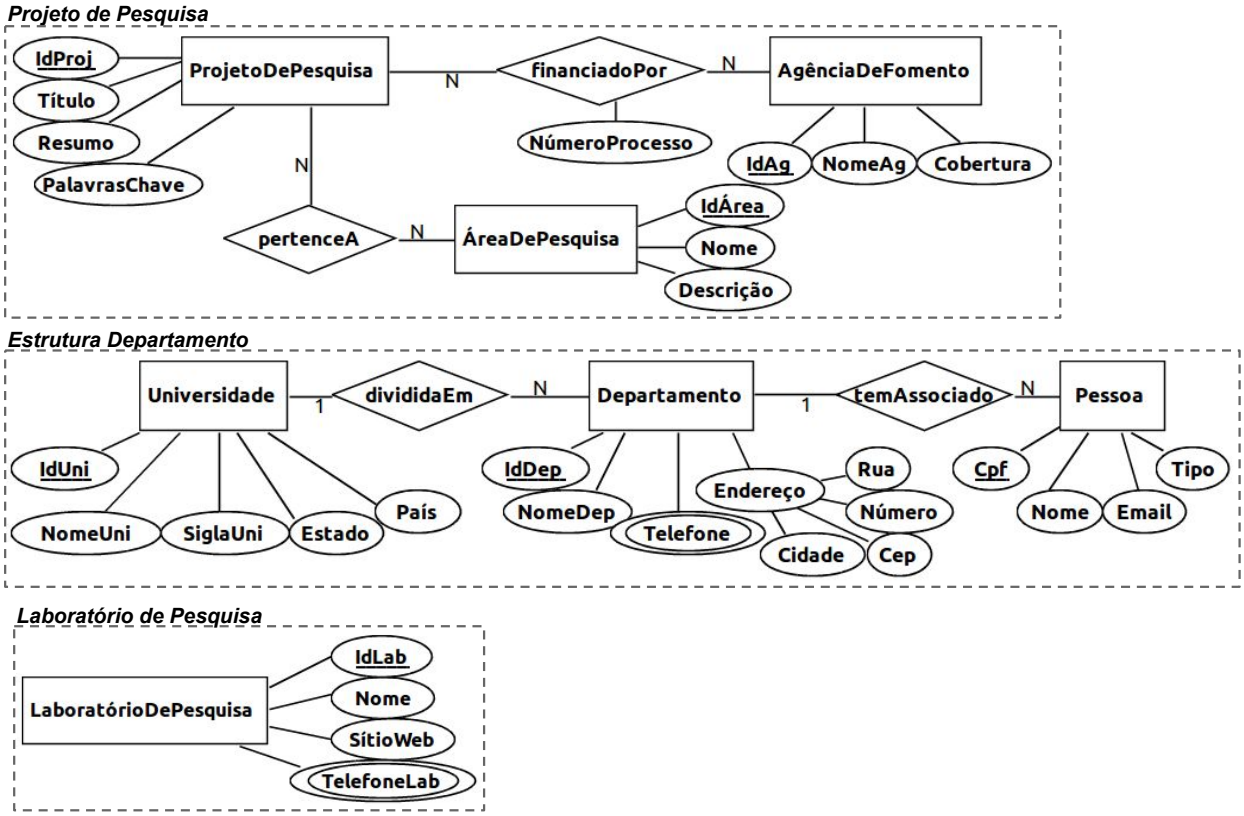

Figura 6.1: *Módulo de dados Projeto de Pesquisa, Laboratório de Pesquisa e Estrutura Departamento.*

A Figura 6.2 mostra os módulos de dados definidos para representar as informações básicas sobre experimentos, seus resultados e os fabricantes dos aparelhos usados na execução dos experimentos: `Experimento`, `Publicação` e `Fabricante`. No DBFD criado, `Experimento` será um módulo obrigatório, enquanto que `Publicação` e `Fabricante` serão módulos opcionais. Ou seja, é necessário armazenar os dados a respeito do experimento, mas não é obrigatório existir dados de publicações e os dados de referência dos fabricantes.

Grupos de sujeitos dos experimentos são representados nos três módulos mostrados na Figura 6.3: `Grupo`, `Humano` e `Não Humano`. No DBFD, todo experimento deverá ser realizado com grupos de sujeitos, podendo eles ser humanos ou não humanos. Os módulos `Humano` e `Não Humano` foram separados por dois motivos: primeiro, porque os dados de um sujeito humano e um sujeito não humano são distintos, e segundo para possibilitar que um grupo de pesquisa que realiza experimentos com somente um tipo de sujeito utilize somente o módulo referente para a construção do banco de dados.

Os módulos de dados projetados para representar informações do protocolo experimental são `Protocolo Experimental`, `Bloco`, `Tarefa`, `Pausa`, `Estímulo`, `TMS` e `Coleta de Dados` e estão retratados na Figura 6.4. Cada tipo de passo do protocolo experimental foi modelado como um módulo de dados diferente. Com isso, permite-se que um laboratório de pesquisa selecione apenas os passos usados nos experimentos que nele são conduzidos.

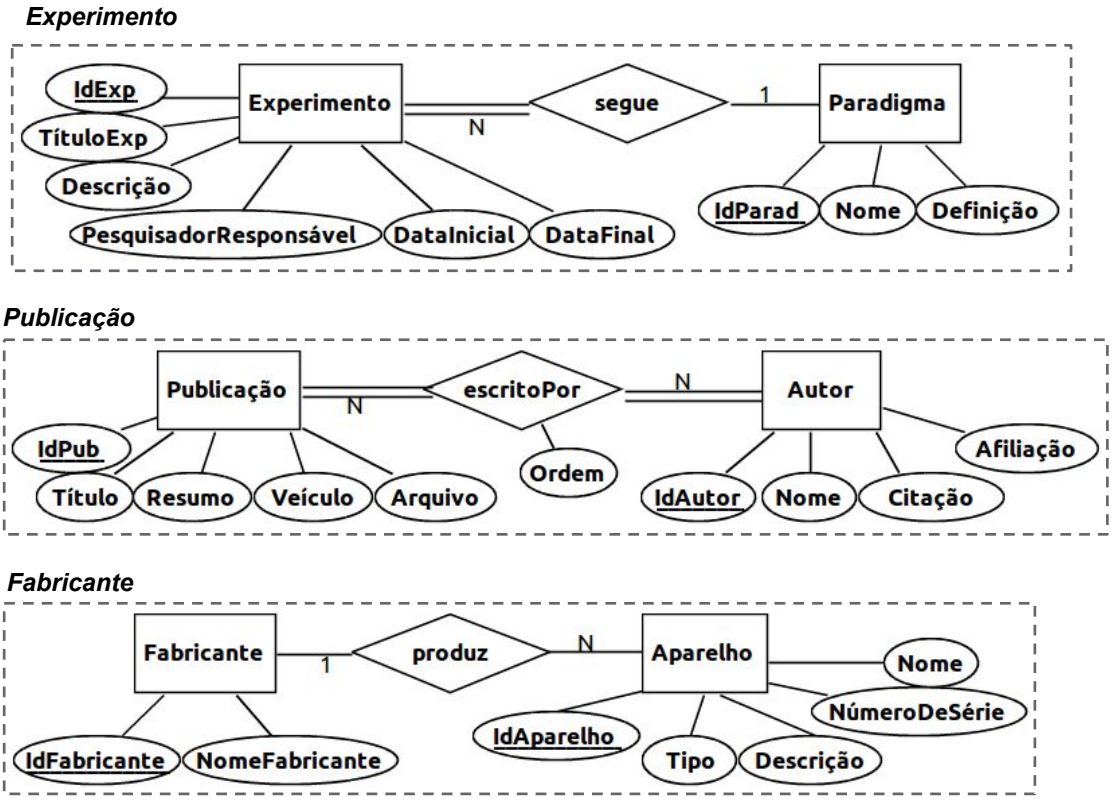

Figura 6.2: *Módulos de dados Experimento, Publicação e Fabricante.*

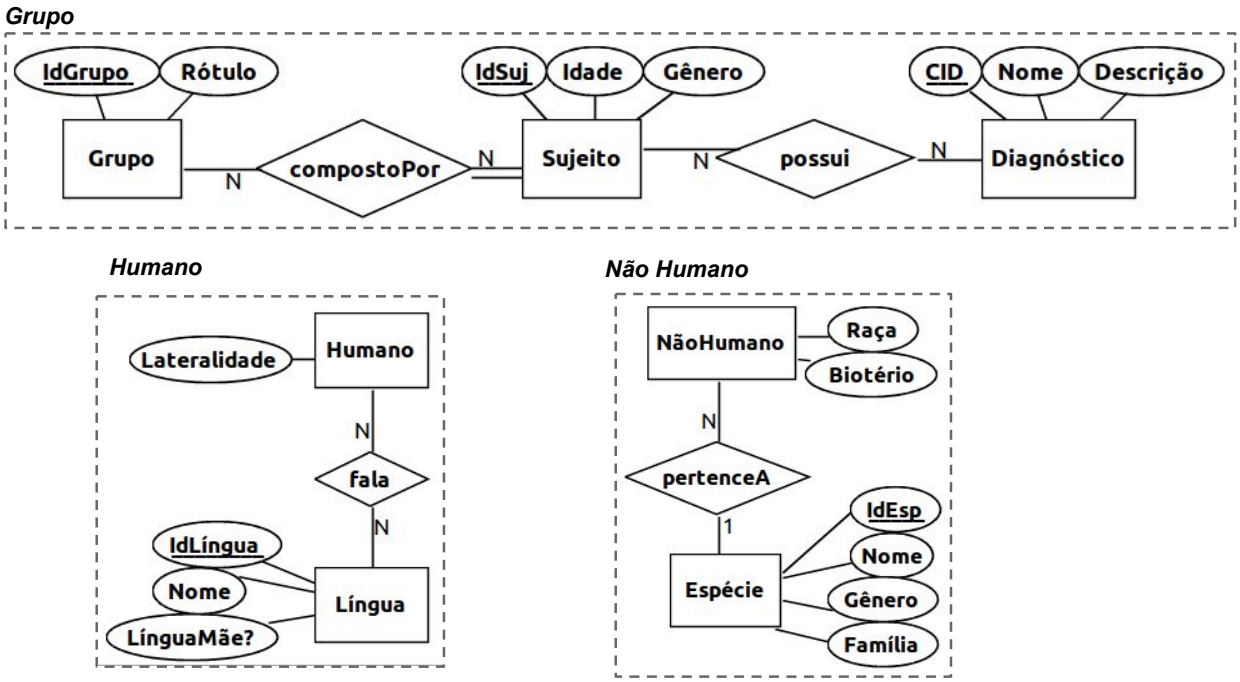

Figura 6.3: *Módulos de dados Grupo, Humano e Não Humano.*

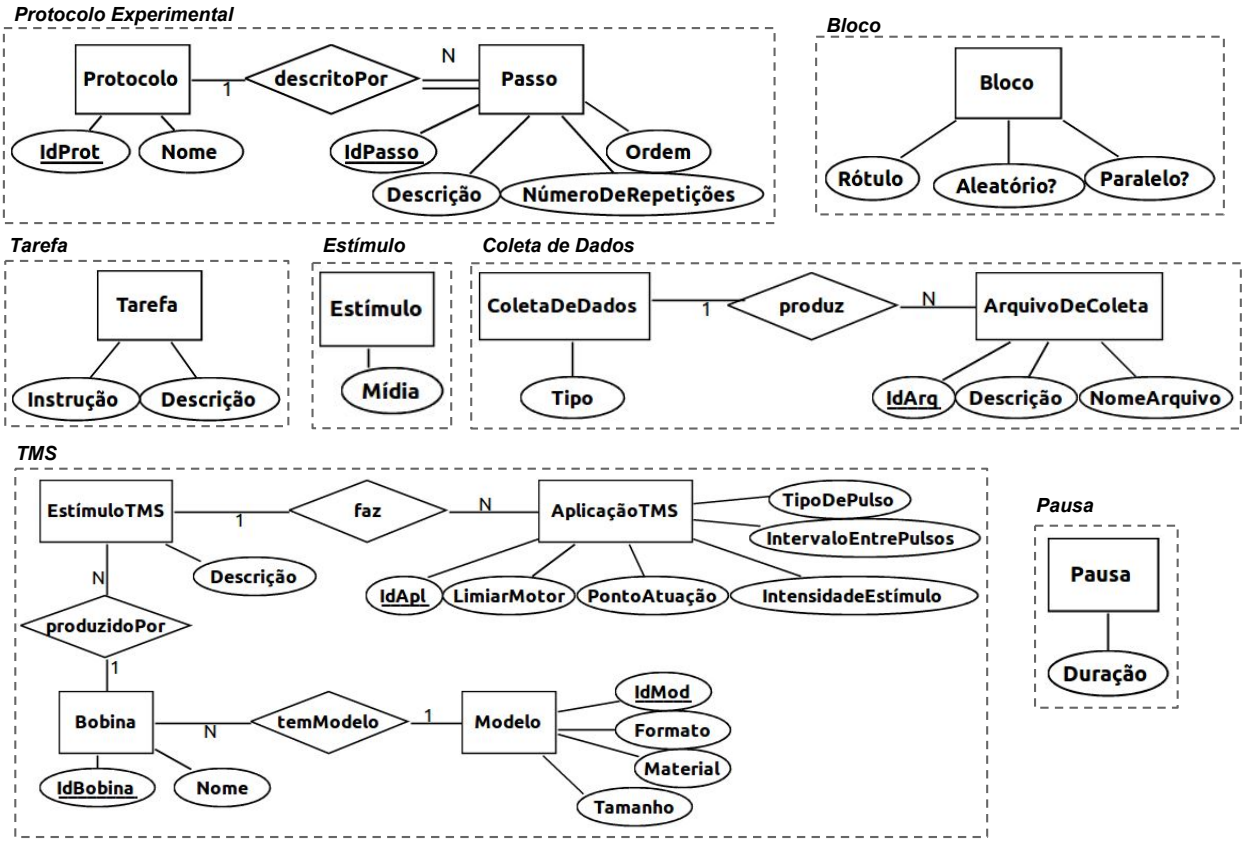

Figura 6.4: *Módulo de dados Protocolo Experimental, Estímulo, Pausa, Tarefa, Bloco, Coleta de Dados e TMS.*

Foram definidos os módulos `Questionário`, `Neuroimagem`, `Coleta com Eletrodo` para representar os diferentes tipos de coletas de dados. Para as coletas com eletrodo, existe a opção de ser uma coleta de `EMG` ou `EEG`. Os diagramas EER dos módulos anteriores estão representados nas Figuras 6.5 e 6.6.

O módulo de dados `Coleta com Eletrodo` é um módulo que agrupa os requisitos que são comuns entre a coleta com EEG e a coleta com EMG, e os módulos `EEG` e `EMG` contêm os atributos específicos. Foram divididos em diferentes módulos para possibilitar que cada projeto de pesquisa contenha em seu banco de dados os dados das coletas com eletrodos que utilizam nas pesquisas.

É importante notar que parte dos requisitos de dados descritos na Seção 6.3.1 não são diretamente representados nos módulos de dados, mas sim nas ligações entre os módulos definidas no DBFD da Figura 6.7, e melhor detalhados por meio de anotações. Como exemplo, considere os seguintes requisitos de dados: “um experimento é composto de grupos”, “sujeitos podem ser humanos ou não-humanos”. Todos esses requisitos se referem a “ligações” que devem ser estabelecidas entre tipos de entidades pertencentes a diferentes módulos.

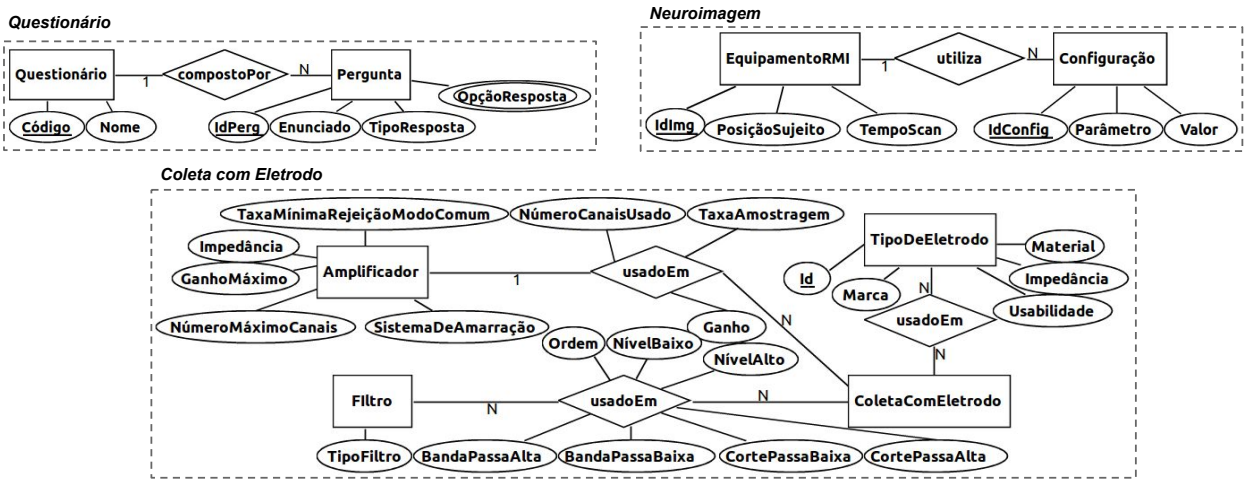

Figura 6.5: *Módulos de dados Questionário, Neuroimagem e Coleta com Eletrodo.*

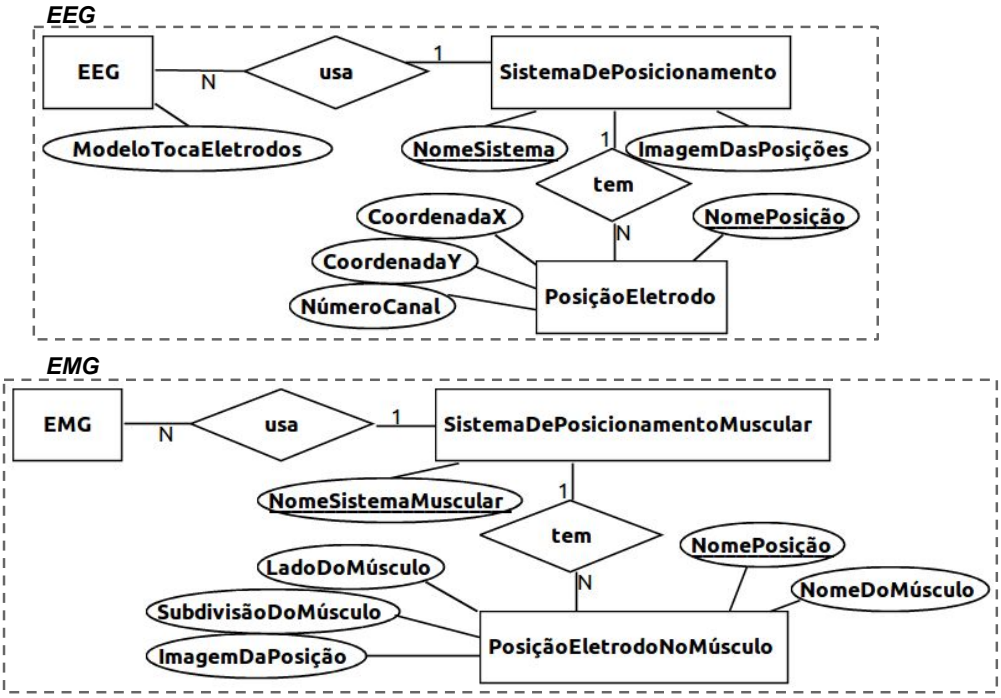

Figura 6.6: *Módulo de dados EEG e EMG.*

Entre os módulos de dados definidos anteriormente, foram definidas ligações que resultaram o DBFD apresentado na Figura 6.7. As anotações que precisam ser criadas sobre as ligações do DBFD da Figura 6.7 estão listadas a seguir. As anotações detalham em nível de modelo EER as ligações definidas no DBFD.

Observe que no DBFD do estudo de caso (Figura 6.7), os módulos `Experimento`, `Grupo` e `Protocolo Experimental` são denotados como módulos base. Além disso, para os módulos `Grupo` e `Protocolo Experimental`, será necessário definir pelo menos um módulo filho para complementar o núcleo do DBFD. Esses módulos de dados vão ser parte de todos os esquemas conceituais de banco de dados derivados a partir do DBFD e, por isso, compõem o núcleo do DBFD.

Neste exemplo é possível observar a variabilidade na possibilidade de escolher quais módulos de dados referentes aos protocolos experimentais estarão presentes no modelo conceitual derivado deste DBFD. Portanto, a variabilidade representada nos DBFDs possibilita selecionar módulos diferentes pra gerar modelos conceituais derivados que são específicos para um grupo de usuários, ao mesmo

tempo que seguem o padrão especificado pelo diagrama.

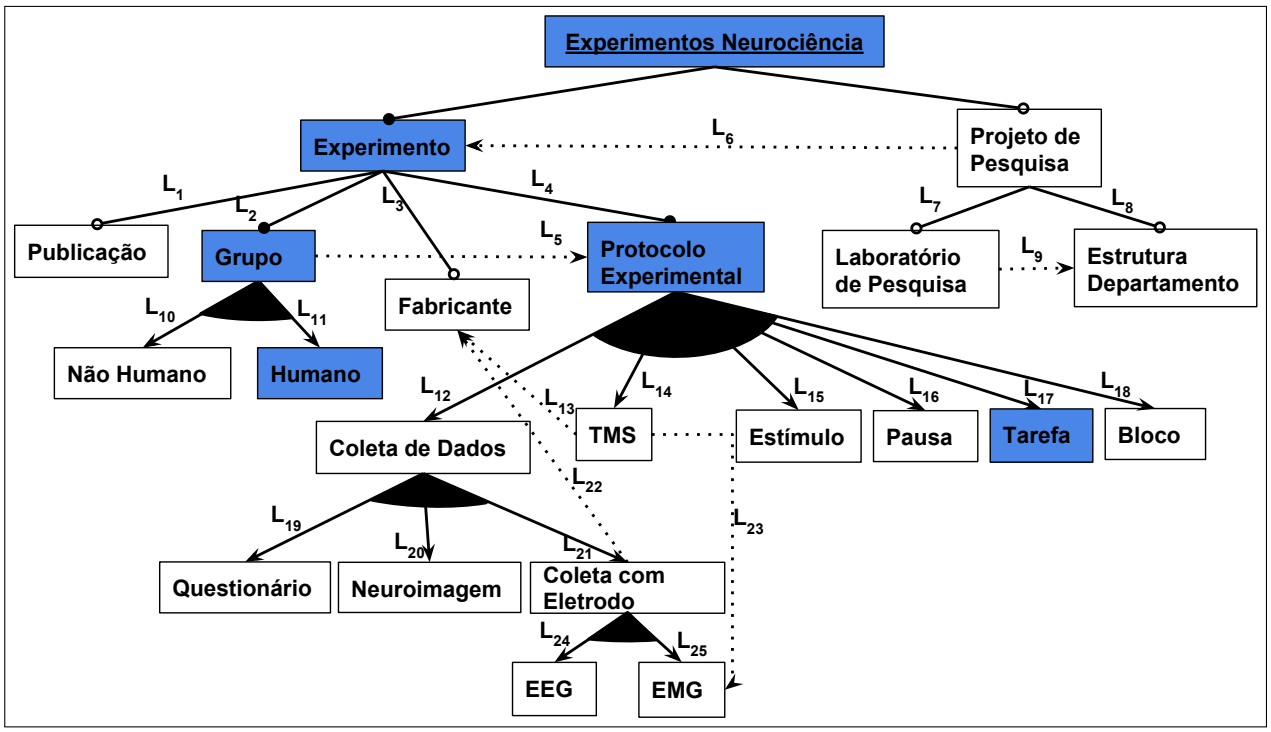

Figura 6.7: *Diagrama de características de banco de dados (DBFD) para o estudo de caso.*

Na Figura 6.7, os módulos destacados em outra cor representam o núcleo do diagrama, exceto pelos módulos `Humano` e `Tarefa`, que foram apenas selecionados como módulos filhos de `Grupo` e `Protocolo Experimental`, que fazem parte do núcleo.

L_1: ADD RELATIONSHIP gera BETWEEN (M Experimento - E Experimento)
AND (M Publicação - E Publicação) M:N PARTIAL:PARTIAL;

L_2: ADD RELATIONSHIP aplicadoA BETWEEN (M Experimento - E Experimento)
AND (M Grupo - E Grupo) 1:N PARTIAL:TOTAL;

L_4: ADD RELATIONSHIP temCadastrado BETWEEN (M Experimento - E Experimento)
AND (M Protocolo Experimental - E ProtocoloExperimental) 1:N;

L_5: ADD RELATIONSHIP temProtocolo BETWEEN (M Grupo - E Grupo) AND
(M Protocolo Experimental - E ProtocoloExperimental) 1:1 TOTAL:TOTAL;

L_6: ADD RELATIONSHIP contém BETWEEN (M Projeto de Pesquisa - E ProjetoDePesquisa)
AND (M Experimento - E Experimento) 1:N;

L_7: ADD RELATIONSHIP hospeda BETWEEN (M Laboratório de Pesquisa -
E LaboratórioDePesquisa) AND (M Projeto de Pesquisa - E ProjetoDePesquisa) 1:N;

L_8: ADD RELATIONSHIP participaDe BETWEEN (M Estrutura Departamento - E Pessoa)
AND (M Projeto de Pesquisa - E ProjetoDePesquisa) M:N attr=Coordenador?;

L_9: ADD RELATIONSHIP contém BETWEEN (M Estrutura Departamento - E Departamento)
AND (M Laboratório de Pesquisa - E Laboratório de Pesquisa) 1:N;

L_{10}: ADD SPECIALIZATION tipoSujeito FROM SUPERCLASS (M Grupo - E Sujeito)
TO SUBCLASS (M Não Humano - E NãoHumano) DISJOINT;

L_{11}: ADD SPECIALIZATION tipoSujeito FROM SUPERCLASS (M Grupo - E Sujeito)
TO SUBCLASS (M Humano - E Humano) DISJOINT;

L_{11}: ALTER (M Sujeito - E Sujeito) ADD ATTRIBUTE TermoDeConsentimento;

L_{12}: ADD SPECIALIZATION tipoPasso FROM SUPERCLASS (M Protocolo Experimental - E Passo) TO SUBCLASS (M Coleta de Dados - E ColetaDeDados) DISJOINT;

L_{13}: ADD RELATIONSHIP feitaPor BETWEEN (M TMS - E Bobina)
AND (M Fabricante - E Fabricante) N:1;

L_{14}: ADD SPECIALIZATION tipoPasso FROM SUPERCLASS (M Protocolo Experimental - E Passo) TO SUBCLASS (M TMS - E EstímuloTMS) DISJOINT;

L_{15}: ADD SPECIALIZATION tipoPasso FROM SUPERCLASS (M Protocolo Experimental - E Passo) TO SUBCLASS (M Estímulo - E Estímulo) DISJOINT;

L_{16}: ADD SPECIALIZATION tipoPasso FROM SUPERCLASS (M Protocolo Experimental - E Passo) TO SUBCLASS (M Pausa - E Pausa) DISJOINT;

L_{17}: ADD SPECIALIZATION tipoPasso FROM SUPERCLASS (M Protocolo Experimental - E Passo) TO SUBCLASS (M Tarefa - E Tarefa) DISJOINT;

L_{18}: ADD SPECIALIZATION tipoPasso FROM SUPERCLASS (M Protocolo Experimental - E Passo) TO SUBCLASS (M Bloco - E Bloco) DISJOINT;

L_{18}: ADD RELATIONSHIP compostoPor BETWEEN (M Bloco - E Bloco)
AND (M Protocolo Experimental - E Passo) 1:N TOTAL:PARTIAL;

L_{19}: ADD SPECIALIZATION tipoColeta FROM SUPERCLASS (M Coleta de Dados - E ColetaDeDados) TO SUBCLASS (M Questionário - E Questionário) DISJOINT;

L_{20}: ADD SPECIALIZATION tipoColeta FROM SUPERCLASS (M Coleta de Dados - E ColetaDeDados) TO SUBCLASS (M Neuroimagem - E EquipamentoRMI) DISJOINT;

L_{21}: ADD SPECIALIZATION tipoColeta FROM SUPERCLASS (M Coleta de Dados - E ColetaDeDados) TO SUBCLASS (M Coleta com Eletrodo - E ColetaComEletrodo) DISJOINT;

L_{22}: ADD SPECIALIZATION tipoAparelho FROM SUPERCLASS (M Fabricante - E Aparelho) TO SUBCLASS (M Coleta com Eletrodo - E Amplificador) OVERLAP;

L_{22}: ADD SPECIALIZATION tipoAparelho FROM SUPERCLASS (M Fabricante - E Aparelho) TO SUBCLASS (M Coleta com Eletrodo - E Filtro) OVERLAP;

L_{22}: ADD RELATIONSHIP tem BETWEEN (M Fabricante - E Fabricante)
AND (M Coleta com Eletrodo - E TipoDeEletrodo) 1:N;

L_{24}: ADD SPECIALIZATION tipoComEletrodo FROM SUPERCLASS (M Coleta com Eletrodo - E ColetaComEletrodo) TO SUBCLASS (M EEG - E EEG) DISJOINT;

L_{25}: ADD SPECIALIZATION tipoComEletrodo FROM SUPERCLASS (M Coleta com Eletrodo - E ColetaComEletrodo) TO SUBCLASS (M EMG - E EMG) DISJOINT;

6.3.3 Esquema Conceitual Personalizado

A partir do diagrama DBFD criado, é possível gerar esquemas conceituais personalizados que são parte da família de esquemas conceituais de banco de dados. Para exemplificar os esquemas

conceituais personalizados gerados a partir do DBFD anterior, considerando os módulos de dados, ligações entre eles e as anotações que complementam as ligações, serão apresentados dois exemplos nas Figuras 6.8 e 6.9.

Na Figura 6.8, o modelo conceitual contém os módulos base, que são `Experimento`, `Grupo` e `Protocolo Experimental`, e também os módulos `Humano` e `Tarefa`, que são escolhidos como filhos dos módulos `Grupo` e `Protocolo Experimental`, respectivamente. Na Figura 6.7, os módulos desse exemplo estão destacados em outra cor. E a Figura 6.9, contém a combinação dos módulos `Coleta de Dados`, `Coleta com Eletrodo` e `EEG`, incluindo o módulo `Fabricante`, que se torna obrigatório quando temos o módulo `Coleta com Eletrodo`, e omitindo a apresentação dos módulos base.

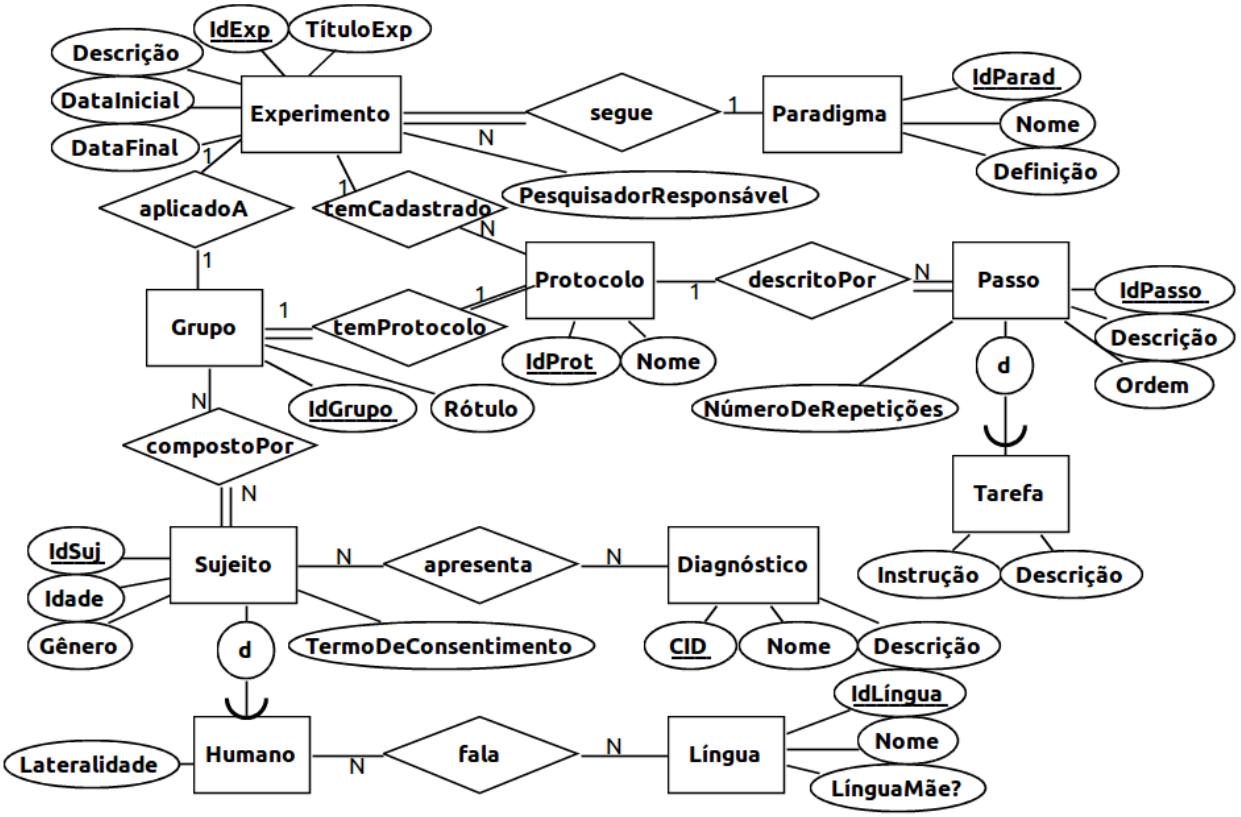

Figura 6.8: *Exemplo 1 de esquema conceitual derivado do DBFD da Figura 6.7.*

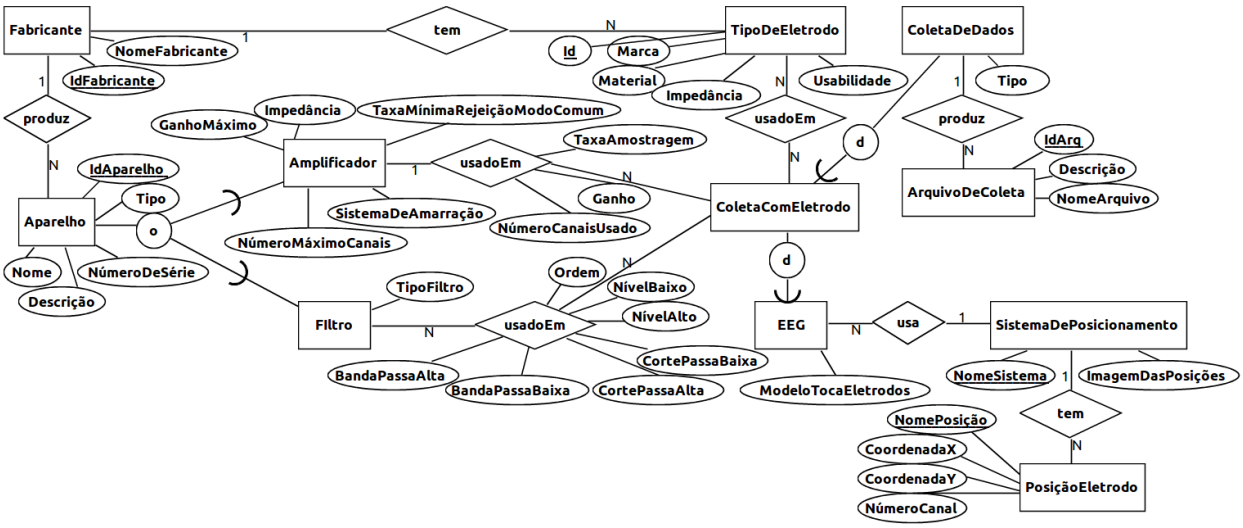

Figura 6.9: *Exemplo 2 de esquema conceitual derivado do DBFD da Figura 6.7.*

6.4 Considerações Finais do Capítulo

A partir da realização do estudo de caso em um domínio de aplicação real – o domínio de dados de experimentos de neurociência – foi possível verificar que o método proposto é viável de ser utilizado para criar famílias de esquemas conceituais de banco de dados em contextos práticos.

O DBFD apresentado neste capítulo pode ser considerado uma contribuição para o domínio de neurociência, já que ele consiste em um modelo conceitual para dados experimentais que, ao mesmo tempo que padroniza a representação dos dados, também acomoda as especificidades dos experimentos conduzidos em diferentes laboratórios participantes de um projeto de pesquisa.

No próximo capítulo, será apresentado o estudo exploratório que foi realizado a fim de avaliar o método proposto e a ferramenta criada para apoiá-lo.

Capítulo 7

Estudo Exploratório

A fim de avaliar a efetividade da utilização dos DBFDs (Diagramas de Características de Banco de Dados) para criar esquemas conceituais flexíveis capazes de lidar com a variabilidade estrutural existente em alguns domínios de aplicação, conduziu-se um estudo exploratório [KPP+02] com alunos e profissionais da área de Ciência da Computação do Instituto de Matemática e Estatística da Universidade de São Paulo.

O estudo tinha três objetivos: avaliar a dificuldade em se reproduzir o método proposto, mensurar o nível de usabilidade da ferramenta DBFD Creator e verificar quanto essa ferramenta é aderente ao passo-a-passo do método. Este capítulo contém primeiramente a descrição de como foi organizado o estudo e, posteriormente, a apresentação dos resultados obtidos para cada objetivo específico do estudo.

7.1 Caracterização dos Participantes

Do estudo, participaram 13 pessoas, sendo 9 delas do sexo masculino e 4 do sexo feminino. Em relação à escolaridade, 61.5% possuem graduação, sendo que 87.5% destes estão cursando o mestrado, 30.8% já possuem mestrado e 75% destes estão cursando o doutorado, e 7.7% possuem o doutorado. Para participar do estudo, era desejável que os participantes conhecessem modelagem conceitual de banco de dados. Entretanto, ter algum conhecimento prévio sobre linhas de produtos de software ou diagramas de características não foi considerado um requisito, embora fosse um ponto positivo.

A Figura 7.1 mostra o nível de conhecimento dos participantes sobre modelagem conceitual de banco de dados e diagramas EER. E a Figura 7.2 mostra o conhecimento sobre linhas de produtos de software e diagramas de características.

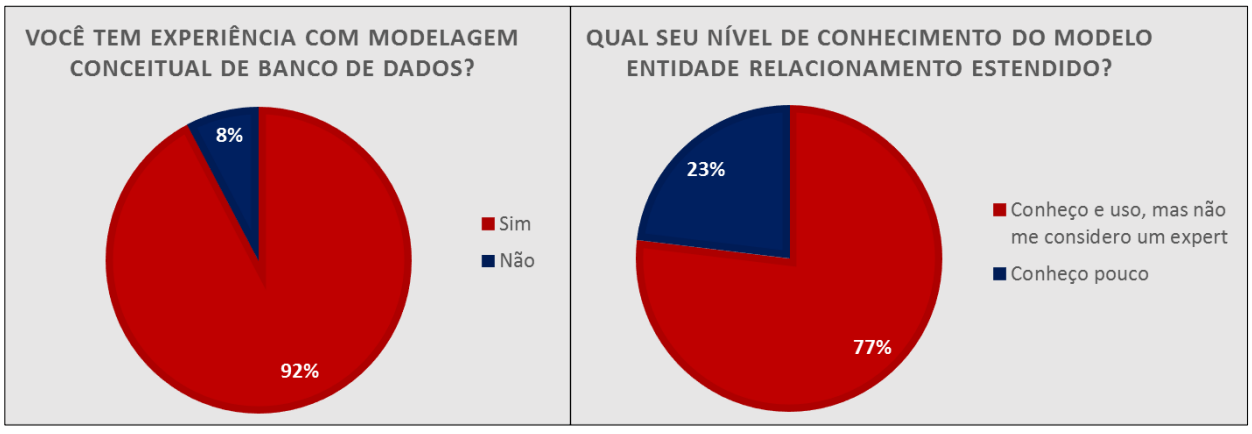

Figura 7.1: *Conhecimento dos participantes em Modelos Conceituais de BD.*

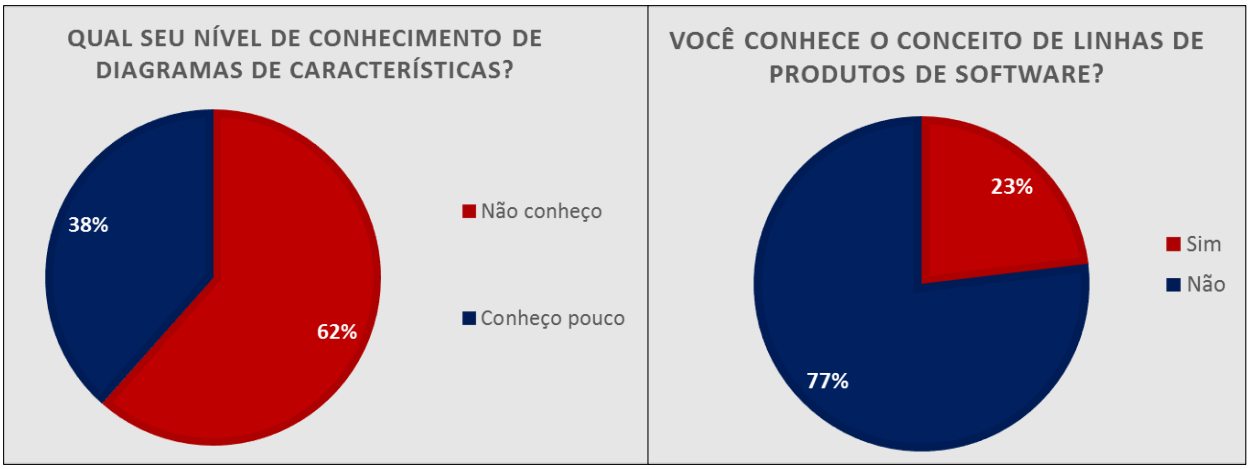

Figura 7.2: *Conhecimento dos participantes em Linhas de Produtos de Software.*

A maioria (92.3%) dos participantes do estudo declararam que tinham experiência em modelagem conceitual. Em relação ao Modelo Entidade Relacionamento Estendido (EER), 23.1% dos participantes declararam que conheciam pouco, enquanto os 76.9% restantes disseram que conheciam bem e usavam, mas não se consideravam especialistas no assunto. Entretanto, somente 23.1% dos participantes declararam que conheciam o conceito de Linhas de Produtos de Software, e a maioria (61.5%) também não conhecia os Diagramas de Características. Como era de extrema importância que todos os participantes conhecessem esses assuntos, foi dado um treinamento de meia hora no início do estudo de avaliação, que explicou os conceitos necessários com exemplos e aplicações práticas.

É necessário ressaltar as ameaças à validade deste estudo. A primeira delas é em relação ao número de participantes, que apesar de ser um número pequeno, consideramos suficiente, dado que este foi um estudo realizado de forma voluntária aplicado somente a alunos do IME-USP. A segunda delas é em relação à proximidade da aluna com alguns dos participantes, apesar disso ocorrer com a minoria dos participantes. Para amenizar essa ameaça, as respostas dos questionários foram analisadas sem ter o nome do participante, e também foi verificada a corretude da solução das tarefas realizadas pelos participantes.

7.2 Organização do Estudo

O estudo realizado pode ser considerado essencialmente qualitativo e foi embasado em experimentos da área de Engenharia de Software. Considerando que o método e a ferramenta criados introduzem novos conceitos para o domínio do problema estudado, buscou-se realizar um estudo onde se pudesse observar problemas ou qualidades que não haviam sido previstos inicialmente. Por isso, foi realizado um estudo exploratório ao invés de um estudo controlado.

Este estudo exploratório teve três objetivos principais, que são:

- avaliar a reprodutibilidade do método proposto;
- verificar a usabilidade da ferramenta de software desenvolvida;
- mensurar a aderência da ferramenta de software ao método proposto.

A Figura 7.3 descreve as etapas que foram realizadas durante o estudo exploratório. Inicialmente, por meio de e-mails, foram recrutados voluntários para participar do estudo. Depois, os participantes passaram por um treinamento, em que foi explicado o método de modelagem conceitual de dados criado neste projeto de mestrado, seguido da apresentação da ferramenta de software de apoio do método. Após o treinamento, foram realizados testes (tarefas) onde cada participante teve que reproduzir o método e utilizar a ferramenta. Por fim, cada participante preencheu alguns

questionários contendo perguntas com alternativas e também perguntas abertas, para avaliar os conceitos e a ferramenta introduzidos.

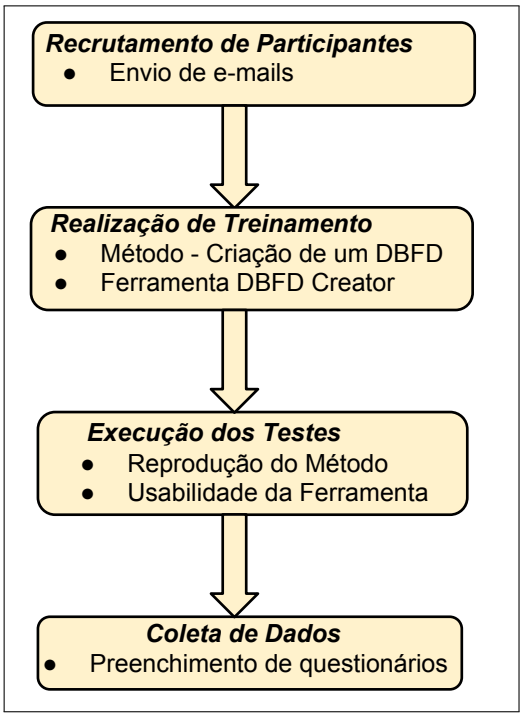

Figura 7.3: *Etapas do estudo.*

Para o recrutamento dos voluntários que realizariam os testes, foram enviados e-mails para as listas de alunos e ex-alunos de graduação e pós-graduação do Departamento de Ciência da Computação do Instituto de Matemática e Estatística da Universidade de São Paulo. Através desses e-mails, conseguimos agendar com 13 voluntários em duas datas para realizar a avaliação, dias 24 e 25 de agosto de 2016.

Para a realização das etapas de treinamento, execução dos testes e coleta de dados do estudo exploratório, foram criados materiais auxiliares, como roteiros, manuais, cenários de atividades e questionários, que guiam o estudo e visam facilitar a interpretação e avaliação dos resultados. Todos os materiais criados encontram-se no Apêndice A desta dissertação.

Todos os voluntários preencheram no início do estudo dois questionários: o de caracterização do participante, que identifica o perfil de cada pessoa que realiza o teste, e o termo de consentimento, onde o participante afirma estar ciente do teste do qual está participando e permite que suas respostas e sugestões sejam divulgadas de forma anônima nos resultados deste trabalho de mestrado.

O treinamento consistiu em uma apresentação de slides que inicialmente explicou os conceitos de modelagem conceitual de banco de dados e linhas de produtos de software. A apresentação também introduziu os diagramas EER e de características, usados como base para os DBFDs. Na sequência, foi explicado método proposto para a criação de um DBFD. E, por último, um exemplo de aplicação do método foi mostrado. Essa apresentação inicial durou cerca de 1 hora.

Após o treinamento, partiu-se para o teste de reprodutibilidade do método, que durou cerca de 2 horas. Os participantes receberam o cenário da tarefa de reprodutibilidade do método, que continha a descrição dos requisitos do domínio de aplicação chamado de "Casas Intelingentes" (*Smart Homes*) e as tarefas que eles deveriam executar para criar um DBFD para o domínio. Terminadas as tarefas do teste de reprodutibilidade, o participante preencheu o questionário de reprodutibilidade do método, que é parte dos resultados do estudo. O questionário de reprodutibilidade do método tinha como objetivo avaliar se a descrição do método está definida de forma que qualquer pessoa que saiba os conceitos básicos do método consiga reproduzi-lo para um domínio de aplicação específico.

Após o teste de reprodutibilidade do método, foi dado um treinamento rápido (de apenas 15 minutos) sobre a ferramenta de software DBFD Creator e o passo-a-passo que deve ser seguido nela

para se criar um DBFD e gerar esquemas personalizados a partir do DBFD criado.

O próximo teste aplicado foi o de usabilidade da ferramenta DBFD Creator, que durou cerca de 30 minutos. Da mesma forma que no teste anterior, os participantes receberam um cenário do teste de usabilidade com as tarefas que deveriam realizar na ferramenta. Nessa etapa, foi disponibilizado um arquivo contendo o diagrama de características criado no *plugin* FeatureIDE e os diagramas EER criados na ferramenta EERCase, a partir de uma solução dada para o domínio de aplicação "Casas Intelingentes". Enquanto o participante realizava as tarefas na ferramenta, a pessoa que estava conduzindo os testes (no caso, a autora deste trabalho) procurou anotar observações sobre a execução das tarefas, como dificuldades ou dúvidas percebidas. Finalizadas as tarefas, cada participante preencheu o questionário de usabilidade da ferramenta, que é proposto na escala *System Usability Scale (SUS)* [Bro96].

Para auxiliar os participantes do estudo durante os testes de reprodutibilidade, foram escritos um manual do método e um exemplo de aplicação completo. Esse exemplo, inclusive, foi usado na etapa de treinamento dos participantes. Outro material importante desenvolvido para o teste de usabilidade foi o manual da ferramenta DBFD Creator, que auxiliou os participantes nas dúvidas eventuais sobre como usar a ferramenta.

Por último, com o objetivo de avaliar se a ferramenta foi desenvolvida de forma a facilitar a utilização do novo método de modelagem, ou seja, avaliar se ela reproduz fielmente o passo-a-passo do método, pedimos que cada participante preenchesse um questionário de avaliação da aderência da ferramenta ao método.

Análises estatísticas foram realizadas sobre as respostas dadas pelos voluntários nos questionários aplicados durante os testes, a fim de se responder as seguintes questões:

1. O método proposto está claramente descrito e pode ser facilmente reproduzido?
2. A ferramenta DBFD Creator é intuitiva e simples de ser utilizada?
3. A ferramenta condiz com a descrição do método?

7.3 Reprodutibilidade do Método

O teste de reprodutibilidade do método foi aplicado para verificar se o método está definido de forma que uma pessoa que receba o material do método e um treinamento rápido sobre ele consiga reproduzi-lo sem dificuldades. Os principais pontos analisados foram: a clareza da definição dos conceitos de base, o nível de detalhamento dos passos e a eficiência de se seguir os passos do método para conseguir criar um DBFD.

A fim de avaliar a reprodutibilidade do método, foi criado um questionário com duas seções de perguntas. A primeira seção consiste em 12 questões, apresentadas na Tabela 7.1, com respostas que expressam o nível de concordância do participante, tendo como opções de "Concordo totalmente" até "Discordo totalmente". A segunda seção contém duas perguntas abertas, apresentadas na Tabela 7.2, para coletar as críticas e observações dos participantes sobre o método.

Tabela 7.1: *Seção 1 de questões do questionário de reprodutibilidade - etapa 1*

ID	Questão
Q1	Achei o nível de detalhamento das etapas do método bom.
Q2	Achei que os conceitos de base, como módulo, relação e anotação, estão suficientemente explicados.
Q3	Achei que as etapas do método poderiam ser subdivididas em outras etapas para facilitar a sua aplicação.
Q4	Achei que a gramática da linguagem utilizada para criar as anotações fácil de ser compreendida.
Q5	Os símbolos de notação de relações e restrições são fáceis de serem compreendidos e usados.
Q6	É fácil seguir os passos do método para se criar um DBFD.
Q7	É fácil definir os módulos de dados.
Q8	É fácil criar as relações e restrições entre os módulos de dados.
Q9	É fácil escrever as anotações para as relações e restrições.
Q10	Eu usaria o método para criar modelos de bancos de dados.
Q11	Eu acho que o método não me ajuda a criar esquemas conceituais customizáveis.
Q12	Foram dados os treinamentos adequados para que eu pudesse usar o método proposto.

Tabela 7.2: *Seção 2 de questões do questionário de reprodutibilidade - etapa 1*

ID	Questão
C1	Deixe seus comentários sobre a descrição do método.
C2	Deixe suas observações e críticas sobre o método.

As porcentagens das respostas para as perguntas de primeira seção estão apresentadas nas Figuras 7.4 e 7.5. As questões foram divididas em duas partes porque na primeira parte a afirmação é positiva, então espera-se respostas entre as opções “concordo” e “concordo totalmente”, enquanto que na segunda parte a afirmação é negativa, fazendo com que se espere respostas entre as opções “discordo” e “discordo totalmente”. A primeira parte está representada na Figura 7.4 e a segunda na Figura 7.5.

Para as questões afirmativas, houve concordância com a maioria delas. Para a questão Q7, houve 38,2% de discordância, o que indica um ponto problemático na definição do método, que diz respeito à definição dos módulos de dados. Acredita-se que essa dificuldade se deu pela ordem dos passos, considerando que o passo de definir os módulos de dados do DBFD é o primeiro na abordagem de construção de DBFD. Com isso, considerando que os diagramas EER ainda não estão desenhados, as pessoas sentiram mais dificuldade em definir quais seriam os módulos para aquele domínio de aplicação. A questão Q9 indicou também certa dificuldade em se escrever as anotações para as ligações do DBFD. Porém, quando se utiliza uma nova linguagem, é comum que uma pessoa só se sinta confortável com a sua sintaxe depois de algum tempo de uso.

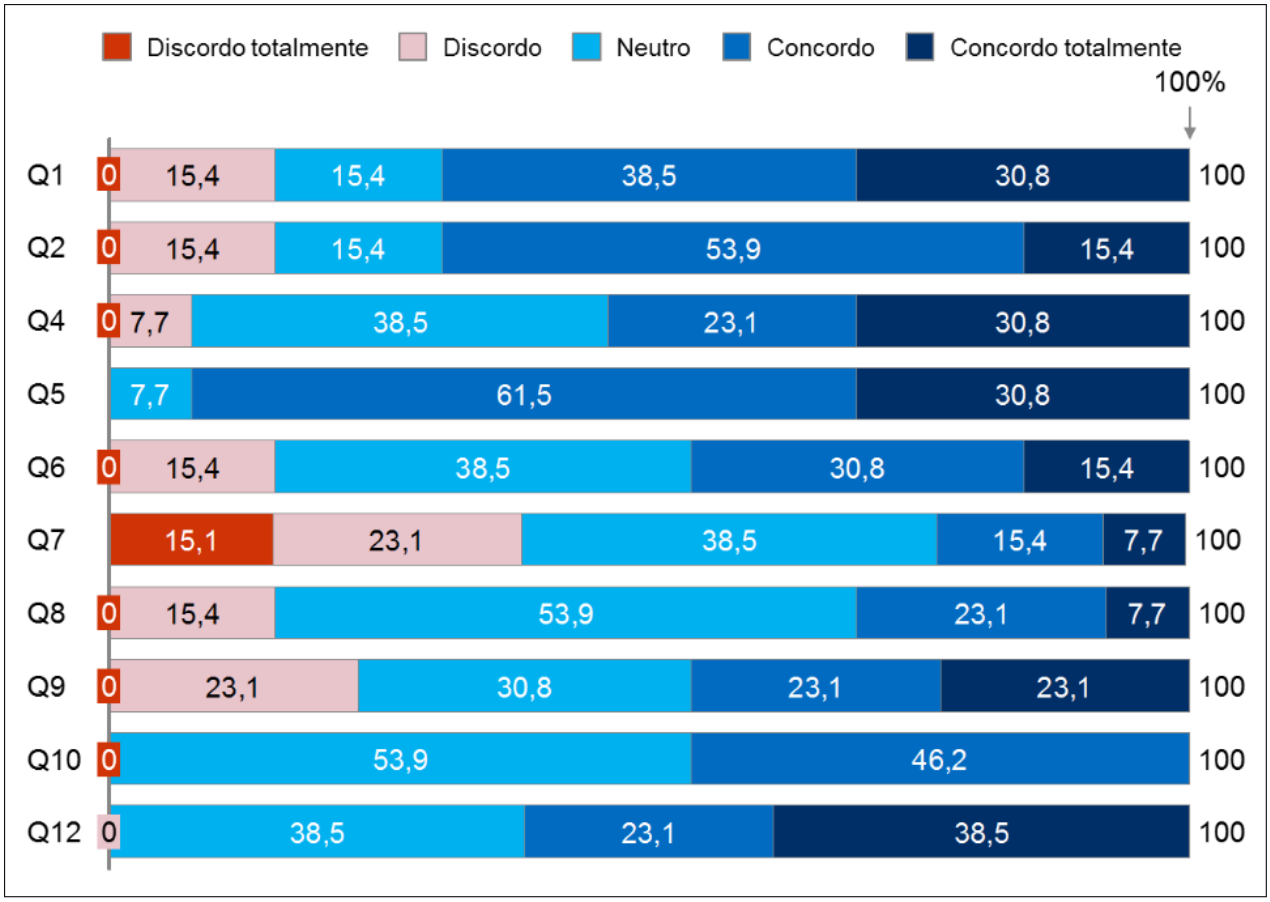

Figura 7.4: *Respostas para questões afirmativas - etapa 1.*

Para as questões negativas, a maioria das pessoas discordou da afirmação, o que significa que para elas as etapas do método não precisam ser subdivididas em outras etapas para facilitar a sua aplicação e também que o método as ajudaria a criar esquemas conceituais personalizáveis. Isso indica que elas consideraram o método uma boa abordagem para lidar com a variabilidade estrutural de dados e que as etapas do método estão bem escritas.

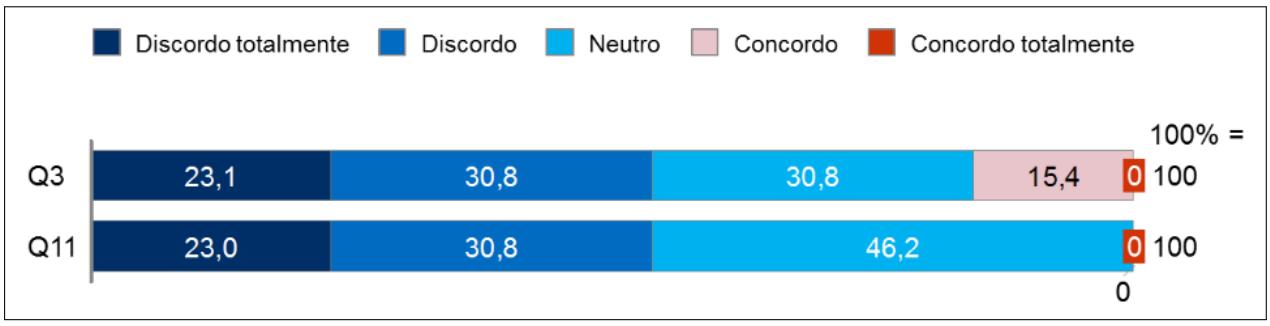

Figura 7.5: *Respostas para questões negativas - etapa 1.*

Para as questões abertas, os comentários positivos em relação ao método indicaram que o método está bem escrito e detalhado, de forma que o necessário para aplicar o método já está coberto pela sua descrição. Um participante destacou ainda que achou muito interessante a ideia de aplicar uma metodologia de software (LPS) em modelagem de dados.

Com relação aos comentários negativos, cerca de 40% dos participantes disseram que tiveram dificuldades em definir o que consistiria um módulo de dados, ou então, em decidir entre adicionar um novo módulo ou adicionar um tipo de entidade a um módulo já existente. Porém, para os participantes, esse é mais um problema relacionado à habilidade de abstração e modelagem do

que um problema inerente ao método. Uma sugestão interessante dada por um participante foi a de desenvolver trabalhos futuros propondo heurísticas ou diretrizes que auxiliem os projetistas na escolha da melhor forma de se criar os DBFDs, para que se obtenha mais benefícios com o método proposto.

Outro ponto que gerou dúvidas nos participantes é a definição de quando é necessário adicionar um módulo de dados vazio a um DBFD. Para um participante, foi necessário adicionar um módulo vazio quando ele queria que fosse adicionado um tipo de relacionamento entre dois tipos de entidades de outro módulo de dados, sem que fossem adicionados novos objetos de dados ao diagrama. Com o exemplo desse participante notou-se que o conceito de módulo de dados vazio foi compreendido e utilizado corretamente.

Por último, notou-se que os participantes não acharam intuitivo iniciar a modelagem pela divisão dos módulos de dados. Destacou-se a observação de que seria mais fácil criar um diagrama EER completo para, em seguida, identificar quais seriam os módulos de dados e então extrair desse diagrama as ligações entre os módulos. Considerando esse resultado obtido do teste, foi proposta uma nova abordagem para criação dos diagramas de características, que é a abordagem de extração de DBFD, descrita no Capítulo 4.

Após a definição da abordagem de extração de DBFD, foi feita uma nova etapa de testes. Dessa etapa, participaram 5 pessoas (que também estavam presentes na primeira etapa de testes) e o objetivo principal foi avaliar a diferença entre reproduzir o método pelas duas abordagens propostas. O questionário de reprodutibilidade do método para a etapa 2 consistiu em duas seções: a primeira com questões do tipo "Escolha o nível de concordância" e respostas de 1 a 5, sendo 1 "discordo totalmente" e 5 "concordo totalmente", e a segunda com perguntas abertas. As questões do questionário estão apresentadas nas Tabelas 7.3 e 7.4.

Tabela 7.3: *Seção 1 de questões do questionário de reprodutibilidade - etapa 2*

ID	Questão
Q1	Achei o nível de detalhamento das etapas do método bom.
Q2	Achei que as etapas do método poderiam ser subdivididas em outras etapas para facilitar a sua aplicação.
Q3	É fácil definir os módulos de dados.
Q4	É fácil criar ligações de consiste-em e restrição entre os módulos de dados.
Q5	É fácil escrever as anotações para as ligações.
Q6	É fácil seguir os passos do método para se criar um DBFD.
Q7	Eu usaria o método para criar modelos de bancos de dados customizados.
Q8	Eu acho que o método não me ajuda a criar esquemas conceituais customizáveis.

Tabela 7.4: *Seção 2 de questões do questionário de reprodutibilidade - etapa 2*

ID	Questão
C1	O que você achou da mudança do passo-a-passo do método para a nova abordagem?
C2	Deixe seus comentários sobre a nova descrição do método.

As respostas da primeira seção de perguntas foram divididas em duas partes: questões afirmativas e questões negativas, e estão apresentadas nas Figuras 7.6 e 7.7, respectivamente. Para as questões afirmativas, na maioria delas as pessoas apresentaram concordância, e somente na questão de definição dos módulos é que um participante ainda discordou ser uma tarefa trivial.

Para as questões negativas, a maioria dos participantes discordou das duas, mostrando que eles discordam que o método não ajude a criar esquemas conceituais personalizáveis e que o método poderia ser subdividido em outras etapas para facilitar a sua aplicação.

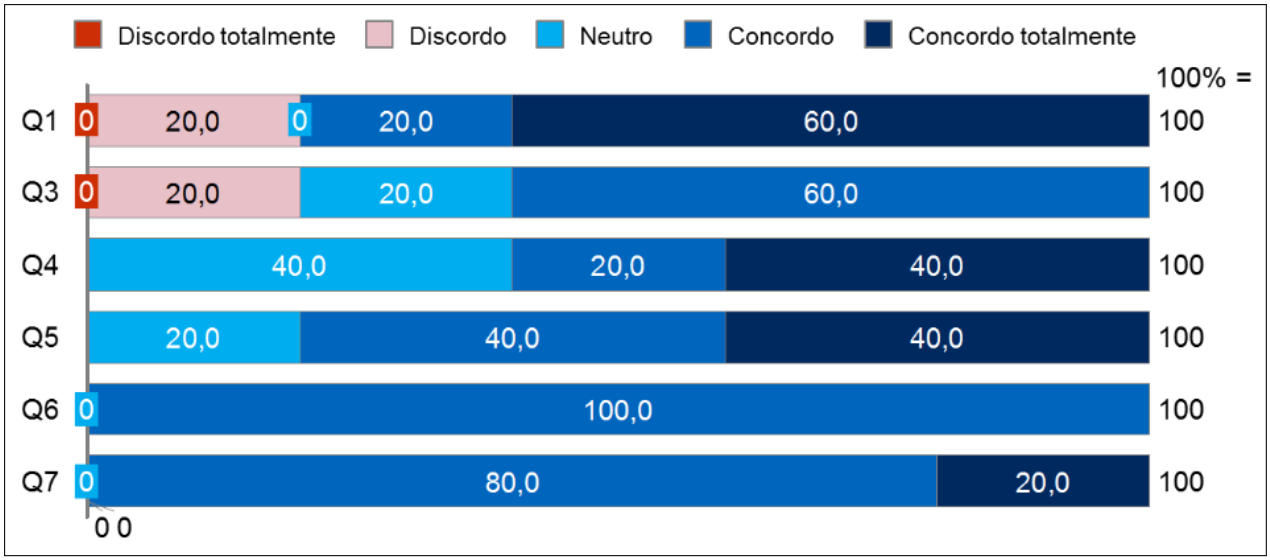

Figura 7.6: *Respostas para questões afirmativas - etapa 2.*

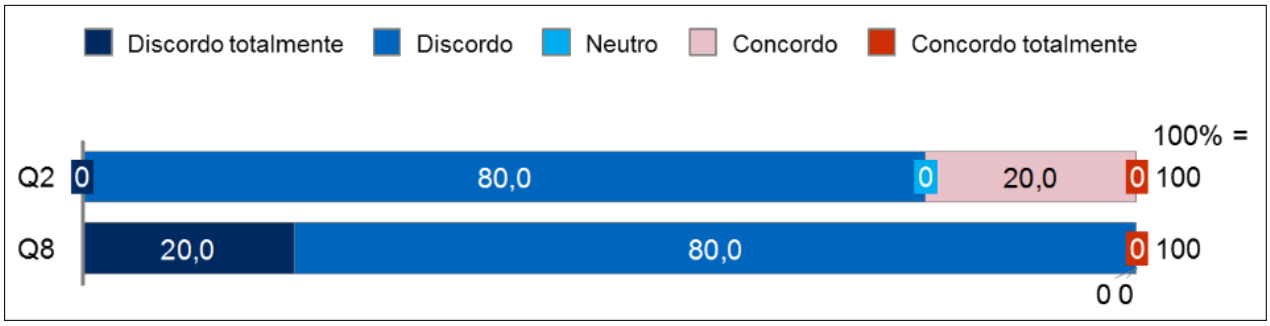

Figura 7.7: *Respostas para questões negativas - etapa 2.*

Segundo os comentários das questões abertas do novo questionário de reprodutibilidade do método, as pessoas avaliaram a nova abordagem como positiva, pois tornou o entendimento da aplicação do método mais fluído. Para os participantes, é mais fácil partir de um modelo conhecido por eles, que é o diagrama EER, e então modularizá-lo, utilizando as definições dos conceitos de base do DBFD para criar os módulos e as ligações.

Portanto, podemos concluir que a maioria dos participantes acredita que o método pode ajudar na criação de modelos de bancos de dados personalizáveis e que, com as mudanças implementadas no passo-a-passo do método para a abordagem de extração de DBFD, o método ficou mais fácil de ser utilizado.

7.4 Usabilidade da Ferramenta de Software

A usabilidade de um software não é algo fácil de se medir quantitativamente. Um teste de usabilidade com usuários reais do sistema é de extrema importância para indicar quais funcionalidades do software as pessoas têm mais dificuldade em utilizar. Um dos métodos mais empregados para medir a usabilidade em escala numérica é o SUS (*System Usability Scale*) [Bro96], criado em 1986 por John Brooke e usado para avaliar vários tipos de produtos e serviços, como hardware, software, dispositivos móveis, páginas web e aplicações. Foi escolhido o SUS devido a confiabilidade desse questionário, que é um dos mais utilizados na literatura para se mensurar usabilidade.

O SUS consiste em um questionário de 10 perguntas, apresentadas na Tabela 7.5, que podem ser respondidas com o nível de concordância do participante, com opções de 1 a 5, onde 1 significa "discordo totalmente" e 5 significa "concordo totalmente". Além das perguntados do SUS, foram

aplicadas outras 7 perguntas abertas, exibidas na Tabela 7.6, para saber das impressões gerais dos participantes ao usarem a ferramenta.

Tabela 7.5: *Questões do SUS*

ID	Questão
S1	Acho que gostaria de usar o sistema com frequência.
S2	Achei o sistema desnecessariamente complexo.
S3	Achei o sistema fácil de usar.
S4	Achei que seria necessário o apoio de um especialista para poder usar o sistema.
S5	As funções do sistema estavam bem integradas.
S6	Achei o sistema muito inconsistente.
S7	Imagino que a maioria das pessoas aprenderiam a usar o sistema rapidamente.
S8	Achei o sistema muito complicado de usar.
S9	Eu me senti muito confiante ao utilizar o sistema.
S10	Eu preciso aprender um monte de coisas antes de continuar usando o sistema.

Tabela 7.6: *Questões abertas de usabilidade*

ID	Questão
Q1	Qual é a sua impressão geral sobre o sistema?
Q2	O que você mais gostou no sistema?
Q3	O que você menos gostou no sistema?
Q4	Se você fosse o desenvolvedor do sistema, qual seria a primeira coisa que você faria para melhorá-lo?
Q5	Existe algo que você sinta que está faltando nas funcionalidades atuais do sistema?
Q6	Se você fosse descrever o sistema para um colega em uma ou duas frases, o que você diria?
Q7	Você tem outros comentários finais? Se sim, escreva-os.

Para chegar na pontuação final do SUS, que é um valor entre 0 e 100, Brooke [Bro96] determinou um cálculo que deve ser feito, com os seguintes passos:

1. Para as perguntas ímpares: subtrair 1 da resposta do usuário.
2. Para as perguntas pares: subtrair a resposta do usuário de 5.
3. Somar os novos valores para as respostas do usuário.
4. Multiplicar a soma dos resultados por 2,5.

Como resultado do questionário SUS, fizemos os cálculos propostos pelo método e tivemos 69,04 como média de pontuação da ferramenta e 67,5 como mediana, em uma escala de 0 a 100. Se desconsiderarmos a pontuação mínima obtida, que foi 25 pontos (e a única pontuação abaixo de 60 obtida), temos 72,71 de média. A maior pontuação obtida no teste foi 87,5. A Figura 7.8 mostra os valores das pontuações obtidas no questionário.

Tivemos apenas um usuário com pontuação abaixo de 60, mas pelas perguntas com respostas abertas, pudemos observar que a maior parte das dúvidas desse usuário eram na verdade sobre o método, e não sobre a ferramenta. Dessa forma, podemos dizer que o software não tem graves problemas de usabilidade.

Podemos dividir a pontuação na escala de A a E, onde A se refere a melhor faixa de pontuação, que vai de 81 à 100, e E se refere a pior faixa de pontuação que vai de 1 até 20. Considerando essa escala, observamos que o resultado de 77% dos participantes ficou entre as faixa A e B. E ainda, como a média de escore está acima de 60 para 92% dos participantes, podemos concluir que a ferramenta desenvolvida, DBFD Creator, atende aos requisitos de usabilidade, sendo fácil de ser utilizada.

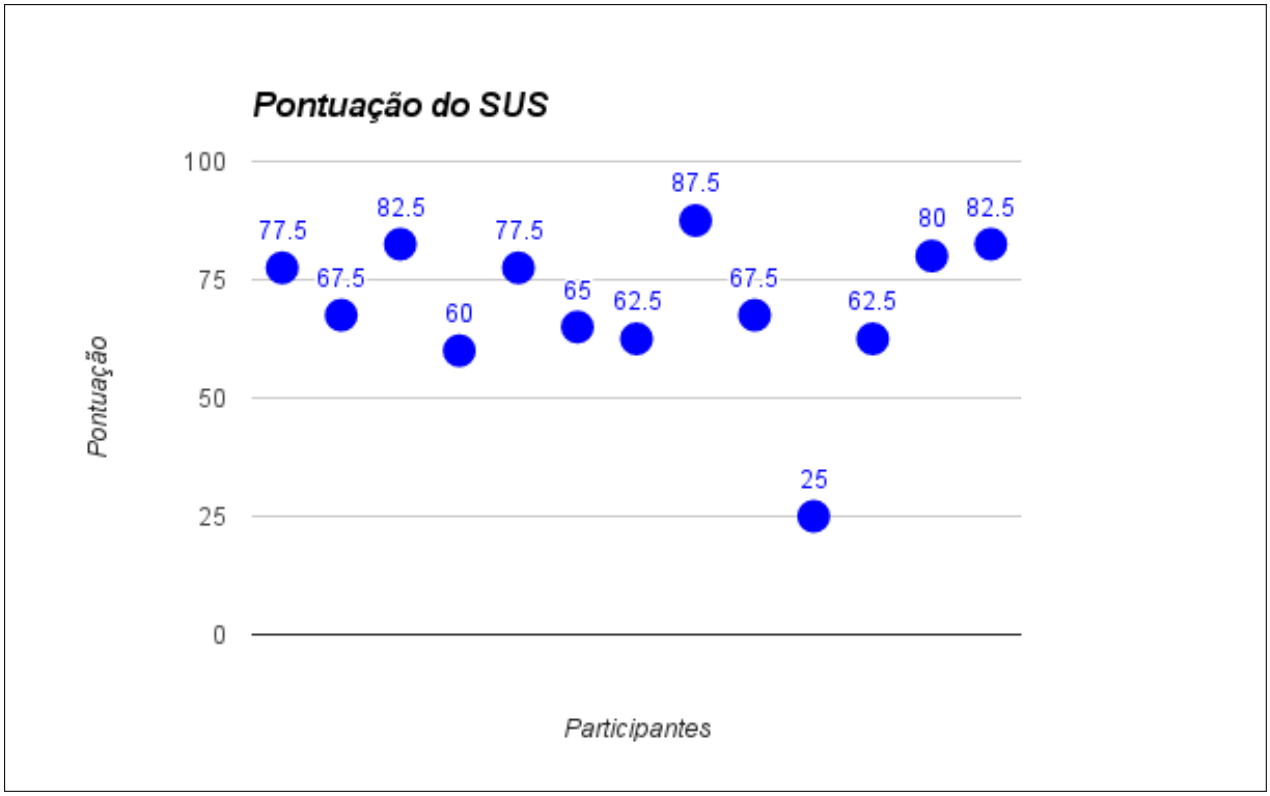

Figura 7.8: *Pontuação do SUS.*

Em relação às perguntas abertas, houve comentários positivos sobre a ferramenta e também algumas sugestões de melhoria. A maioria dos participantes comentou que a ferramenta é intuitiva e fácil de usar, tem uma interface limpa e amigável, tem uma boa responsividade e viabiliza a aplicação do método proposto. As funcionalidades da ferramenta das quais a maioria das pessoas mais gostaram são as de geração da imagem do diagrama EER completo e do *script* SQL para um esquema personalizado.

As características da ferramenta das quais as pessoas menos gostaram são a necessidade de se importar arquivos produzidos em outras ferramentas auxiliares e a falta de visualização do diagrama EER importado para um determinado módulo, o que impossibilita verificar se o diagrama importado está correto. E as funcionalidades que as pessoas apontaram como sugestões de melhorias são: incluir mais informações sobre erros, acrescentar exemplos de anotações para guiar o usuário na página de criação de uma anotação e possibilitar que o usuário importe uma lista de anotações de um arquivo texto, para que o trabalho não fique tedioso quando se tem um grande número de anotações.

Alguns erros identificados no teste de usabilidade foram corrigidos e as melhorias mais simples propostas pelos participantes foram implementados na versão do software que se encontra no Bitbucket.

7.5 Aderência da Ferramenta ao Método

Na avaliação da aderência da ferramenta ao método, os principais objetivos foram identificar as dificuldades que se pode encontrar ao reproduzir o passo-a-passo do método na ferramenta e verificar se ferramenta é adequada ao método proposto.

Para isso, foi criado um questionário com três seções de perguntas. A primeira seção contém 7 perguntas para avaliar a dificuldade do participante ao executar as tarefas propostas, que reproduziam o método na ferramenta. As perguntas têm como início “Selecione o nível de dificuldade para realizar cada uma das seguintes tarefas na ferramenta DBFD Creator:” e o restante delas está apresentado na Tabela 7.7, e com as opções de resposta de 1 a 5, onde 1 significa “muito fácil” e 5

significa “muito difícil”.

A segunda seção contém 2 perguntas diretas sobre a impressão do participante, que podem ser respondidas com as opções “totalmente”, “neutro” e “parcialmente”. E na terceira seção, fizemos perguntas abertas, para que o participante pudesse dar sua opinião e suas sugestões para melhoria da ferramenta. As perguntas das seções 2 e 3 estão descritas na Tabela 7.8.

Tabela 7.7: *Grupo 1 de questões do questionário de aderência.*

ID	**Questão**
Tarefa1	Criar módulos de dados.
Tarefa2	Associar um diagrama EER a um módulo de dados.
Tarefa3	Criar relações e restrições entre módulos de dados.
Tarefa4	Criar módulos de dados vazios.
Tarefa5	Escrever anotações para as relações e restrições.
Tarefa6	Criar um esquema de banco de dados específico a partir de um DBFD.
Tarefa7	Criar uma alteração para evoluir um banco de dados por meio de um DBFD.

Tabela 7.8: *Grupo 2 de questões do questionário de aderência.*

ID	**Questão**
Q1	Você considera a ferramenta adequada ao método proposto?
Q2	Você achou que a navegabilidade pela ferramenta é correspondente aos passos do método?
Q3	O que você mudaria na ferramenta para que essa fosse mais fiel ao método?
Q4	Deixe seus comentários sobre a aderência da ferramenta ao método.
Q5	Deixe suas sugestões e críticas para que possamos melhorar a ferramenta.

As respostas para as questões da seção 1 estão apresentadas na Figura 7.9. Utilizamos diagramas de caixa (do inglês *boxplot*), para representar e analisar para cada questão a variação de cada resposta no conjunto de respostas de todos participantes [IH87]. Com o diagrama de caixa é possível representar a mediana, o primeiro e terceiro quartil, o valor mínimo e máximo, e também os outliers para as respostas de cada questão.

De acordo com as respostas obtidas, as tarefas para reprodução do método na ferramenta foram consideradas muito fáceis ou fáceis. Apenas para as tarefas 5 e 7 algumas pessoas, apesar de indicarem que não tiveram dificuldade, apontaram a tarefa como algo não trivial. Além disso, obtivemos somente uma resposta mencionando dificuldade para associar um diagrama EER a um módulo de dados. Para todas as outras tarefas, nenhuma pessoa declarou que teve dificuldade em reproduzi-las.

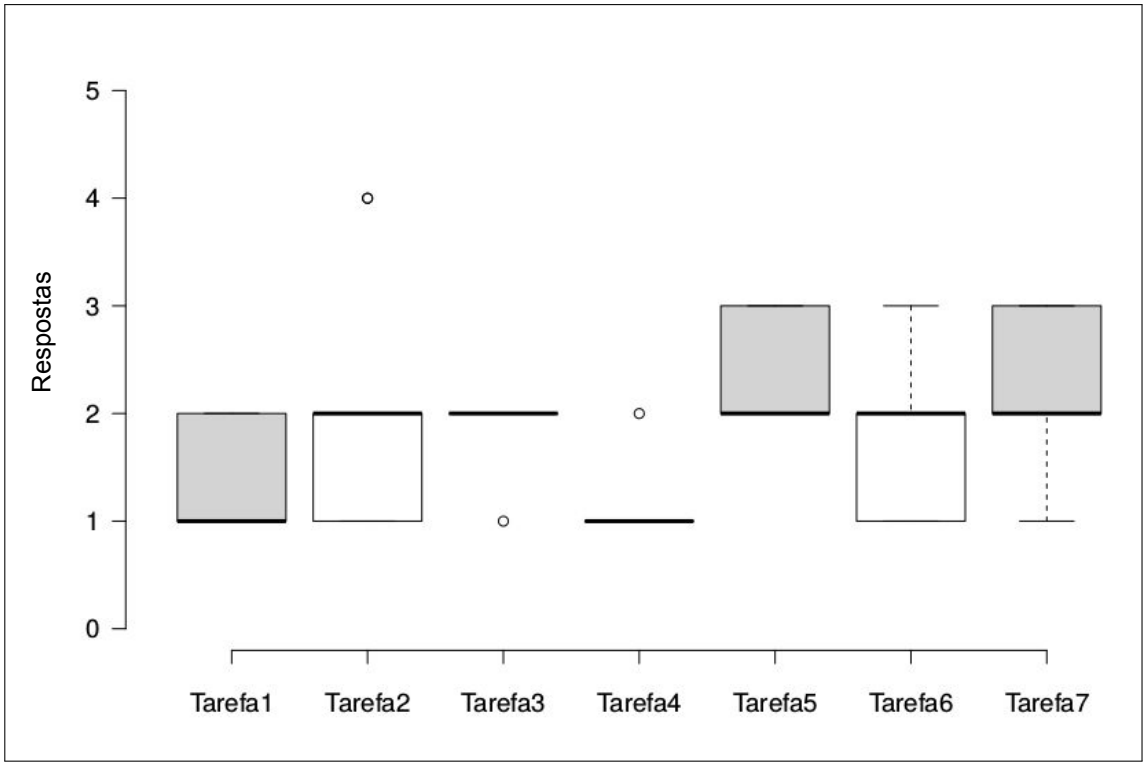

Figura 7.9: *Respostas do questionário de aderência.*

Segundo as respostas das questões Q1 e Q2, que estão apresentadas na Figuras 7.10, a maioria dos participantes, 76,9%, acharam que a ferramenta é adequada ao método proposto. Em relação à navegabilidade pela ferramenta ser correspondente aos passos do método, 53,8% concordou. Porém, uma porcentagem considerável dos participantes (38,5%) considerou que é parcialmente correspondente. Essa porcentagem condiz com os comentários feitos nas questões abertas e observou-se que essa percepção por parte dos participantes aconteceu principalmente porque em algumas telas o usuário tem que avançar e retornar várias vezes, como na importação do diagrama EER e na inserção de uma nova anotação. Para versões futuras do DBFD Creator, essa é uma reclamação que será levada em consideração.

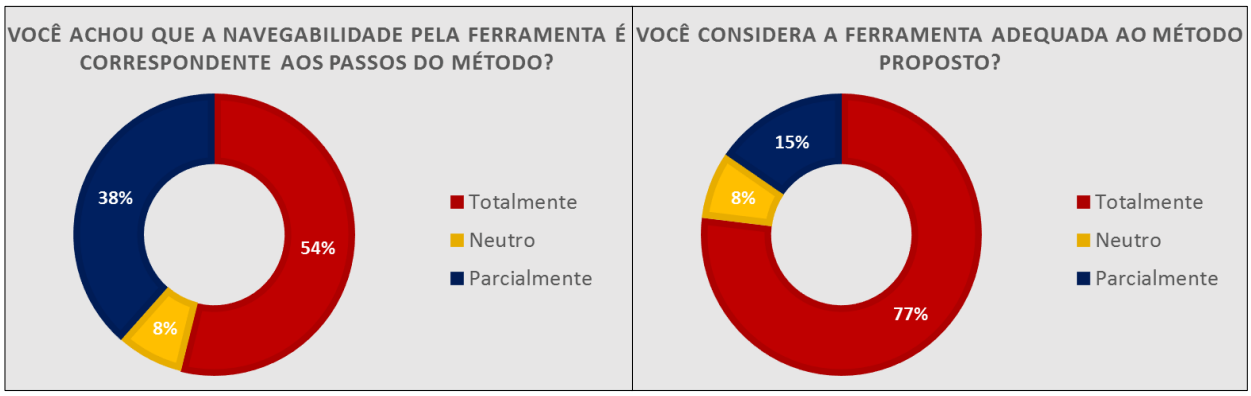

Figura 7.10: *Resposta das questões Q1 e Q2 do questionário de aderência.*

Nas perguntas abertas, de Q_3 a Q_5, recebemos como principais sugestões: modificar a navegação para que ela fique mais sequencial, como é o método (sem que seja necessário ir e voltar para uma mesma página várias vezes para acompanhar o método); e deixar a importação do diagrama EER na mesma página de criação dos módulos. Além disso, os participantes comentaram que a definição dos passos a serem seguidos para a reprodução do método proposto ficou mais clara com a utilização

da ferramenta.

7.6 Conclusões do Estudo Exploratório

A partir do estudo exploratório realizado, conseguimos avaliar e obter respostas para as questões iniciais que motivaram a validação, que foram:

1. O método proposto está claramente descrito e pode ser facilmente reproduzido?
2. A ferramenta DBFD Creator é intuitiva e simples de ser utilizada?
3. A ferramenta condiz com a descrição do método?

Com relação à primeira pergunta, foi possível levantar alguns pontos negativos em relação à descrição inicial do método. Os principais pontos negativos foram: a dificuldade em se determinar o que consiste um módulo de dados e em seguir uma abordagem onde é necessário começar definindo elementos que não são comuns na modelagem conceitual de banco de dados. Levando em consideração esses pontos, foi proposta uma nova abordagem, chamada de abordagem de extração de DBFD, para facilitar a reprodução do método. A principal alteração na abordagem foi em relação ao passo-a-passo do método, buscando-se iniciar de uma modelagem comum de banco de dados, para só então definir os elementos que constituem um DBFD.

Acredita-se que a nova abordagem de construção dos DBFDs facilitou a reprodução do método. E, além disso, foi obtido um retorno positivo em relação ao método proposto. A maioria dos participantes acredita que o método seja útil para a construção de esquemas conceituais personalizáveis para lidar com o problema de variabilidade estrutural de dados.

Para responder à segunda pergunta, foi usado um questionário bastante referenciado na literatura de Engenharia de Software Experimental – o SUS- -, por meio do qual foi possível calcular uma pontuação para a usabilidade da ferramenta DBFD Creator. Segundo as diretrizes da literatura, podemos considerar que a ferramenta é apropriada em termos de usabilidade. Também foram colocadas aos participantes do estudo questões abertas, por meio das quais foi possível coletar críticas e sugestões para melhorias na ferramenta. Algumas dessas melhorias já estão implementadas na versão atual do DBFD Creator.

Finalmente, foi avaliada também a aderência da ferramenta ao método. Os participantes concordaram que a ferramenta está desenvolvida de forma a permitir que o passo-a-passo do método seja seguido na própria ferramenta. Também consideraram que as tarefas propostas a eles para reproduzir o método na ferramenta foram fáceis de serem realizadas. Entretanto, por meio dos comentários recebidos nas questões abertas, identificou-se que é preciso melhorar a navegabilidade da ferramenta, para deixá-la ainda mais aderente ao método proposto.

7.7 Considerações Finais do Capítulo

Neste capítulo foram apresentadas as etapas do estudo exploratório aplicado em um grupo de pessoas para avaliar o método proposto, a ferramenta de software criada para apoiá-lo e também a aderência dessa ferramenta ao método, bem como os resultados desse estudo.

Por meio do estudo, podemos concluir que o método proposto pode ser reproduzido sem muitas dificuldades quando se utiliza a abordagem de extração do DBFD, proposta no Capítulo 4. Além disso, constatamos que a ferramenta de software não contém problemas de usabilidade. E por último, a ferramenta pode ser utilizada para apoiar a criação dos DBFDs.

No próximo capítulo serão apresentadas as conclusões para este trabalho de mestrado e alguns trabalhos futuros relacionados.

Capítulo 8

Conclusões

Este capítulo apresenta um resumo do trabalho de mestrado, destacando em seguida suas principais contribuições, e, por fim, traz os possíveis trabalhos futuros relacionados.

8.1 Resumo

A motivação deste trabalho de mestrado surgiu da observação de que, em alguns domínios de aplicação, diferentes organizações costumam ter requisitos de dados comuns, mas também apresentam necessidades específicas que devem ser consideradas no projeto de banco de dados. Em domínios desse tipo, é desejável que todos os bancos de dados locais tenham uma base comum em seus esquemas, e que as variações existentes entre seus esquemas sejam de alguma forma controladas por um modelo de dados padrão especificado para o projeto.

Apesar dos esquemas conceituais de banco de dados serem utilizados para se criar representações padronizadas dos dados de um domínio, com as técnicas de modelagem tradicionais não conseguimos destacar a variabilidade estrutural de dados por meio de esquemas conceituais flexíveis. Esse é um dos motivos que pode dificultar o compartilhamento de dados entre organizações de um mesmo domínio de aplicação que tenham algum grau de relacionamento.

Para representar variabilidade estrutural dos dados, este trabalho de mestrado introduziu um novo tipo de modelo conceitual de dados, os *Diagramas de Características de Banco de Dados (DBFDs)*, que são uma extensão dos *Diagrama de Características* usado nas Linhas de Produtos de Software, um paradigma da Engenharia de Software. Um DBFD possibilita a criação de uma família de esquemas conceituais de bancos de dados, que compreende todas as possíveis variações de esquema para um determinado domínio.

Além de definir o novo tipo de modelo conceitual, este trabalho de mestrado também propôs duas abordagens para guiar a criação desses modelos. A partir de um modelo DBFD criado com essas abordagens, é possível gerar esquemas conceituais personalizados mas que seguem um modelo padronizado para um determinado domínio, ou seja, esquemas que pertencem à uma mesma família de esquemas.

Para facilitar a criação e uso dos DBFDs, foi desenvolvida uma ferramenta computacional chamada DBFD Creator, que ampara todos os passos do método para se obter um DBFD. A ferramenta possibilita ainda, após a construção de um DBFD, a geração de esquemas conceituais personalizados e dos *scripts* SQL de criação dos bancos de dados físicos correspondentes.

Por fim, foram realizadas dois tipos de avaliação para o método de modelagem proposto e a ferramenta desenvolvida: um estudo de caso e um estudo exploratório. No estudo de caso, foi possível testar a aplicação do método proposto em um domínio de aplicação real – o de dados de experimentos de neurociência – e verificar que ele é capaz de modelar a variabilidade dos dados desse domínio de forma eficiente.

O segundo estudo de avaliação foi o estudo exploratório, que foi realizado com um grupo de 13 pessoas no IME-USP. Durante o estudo, os participantes receberam treinamentos sobre como reproduzir o método e também como utilizar a ferramenta desenvolvida, o DBFD Creator. Após

os treinamentos, oas participantes foram instruídos a realizar tarefas tais como a de reproduzir o método para um outro domínio de aplicação e utilizar a ferramenta para o construir o DBFD feito na tarefa anterior.

Após a realização das tarefas, os participantes tiveram que preencher três questionários para a avaliação do estudo. O primeiro questionário, de avaliação da reprodutibilidade do método, tinha como objetivo avaliar qual foi o nível de dificuldade que o participante teve pra reproduzir o método.O segundo questionário, de usabilidade, visava medir a usabilidade da ferramenta desenvolvida, verificando, por exemplo, se o fluxo da ferramenta é claro e se o usuário consegue executar as atividades que foram propostas de forma fácil na ferramenta.O terceiro questionário, de aderência da ferramenta ao método, tinha como objetivo verificar se os participantes acharam que o fluxo apresentado na ferramenta condiz com o passo-a-passo do método para a criação dos DBFDs.

A partir das respostas aos três questionários de avaliação, foram identificados pontos onde o método ou a ferramenta poderiam ser aperfeiçoados. O primeiro ponto é a descrição do passo-a-passo do método. Devido ao nível de dificuldade reportado pelos participantes, foi proposta uma segunda abordagem de criação dos DBFDs que, após uma nova etapa do estudo exploratório focada no método, mostrou ser mais simples e intuitiva para os participantes. Outro ponto levantado é em relação a navegabilidade da ferramenta, que os participantes classificaram como apenas satisfatória, por exigir idas e voltas em muitas telas para se executar alguma ação.

Os resultados do estudo também destacaram pontos positivos. Por exemplo, os participantes afirmaram que utilizariam o método proposto para criar esquemas conceituais flexíveis e acharam que a ferramenta desenvolvida condiz com o passo-a-passo do método.

Com os estudos realizados, pudemos observar que tanto o método quanto a ferramenta servem aos seus propósitos. Entretanto, existem alguns pontos a serem melhorados e que podem ser alvo de trabalhos futuros.

8.2 Contribuições

A principal contribuição deste trabalho de mestrado foi a proposta de um novo método de modelagem conceitual para representar variabilidade de dados, por meio do uso dos Diagramas de Características de Banco de Dados (DBFDs). Além da descrição dos DBFDs, foram propostas duas abordagens diferentes para se projetar famílias de esquemas conceituais por meio de DBFDs.

O conceito de famílias de esquemas conceituais foi definido neste trabalho de mestrado e é derivado do conceito de famílias de produtos de software. Uma família de esquemas conceituais de banco de dados compreende todas as possíveis variações de esquemas conceituais de banco de dados para um determinado domínio de aplicação.

As definições dos DBFDs e do método de criação de famílias de esquemas conceituais foi tema de um artigo publicado nos anais do XXX Simpósio Brasileiro de Banco de Dados (SBBD'15) [MB15].

Outra contribuição importante deste trabalho é a ferramenta DBFD Creator, que permite construir um DBFD e a partir dele gerar esquemas conceituais personalizados e os *scripts* SQL para criar os respectivos esquemas de banco de dados físico. Essa é uma ferramenta de código aberto, que possibilita que outros pesquisadores colaborem com este trabalho de pesquisa.

Por último, durante a etapa de estudo de caso foi possível criar uma família de esquemas conceituais para representar dados de experimentos de neurociência. Essa é uma contribuição específica para projetos de neurociência, que já está sendo utilizada no projeto NeuroMat e poderá ser utilizada em outros projetos de pesquisa. O domínio de neurociência é um bom exemplo de domínio com alto grau de variabilidade estrutural de dados.

8.3 Trabalhos Futuros

Finalizando este trabalho de mestrado, podemos citar alguns possíveis trabalhos futuros relacionados. O primeiro diz respeito à criação de heurísticas, ou diretrizes, para a definição dos módulos

de dados. Considerando que este foi um ponto de dificuldade na reprodução do método, seria importante trabalhar melhor essa definição para que o método fique mais consistente. Essas heurísticas podem auxiliar tanto na criação dos módulos de dados comuns quanto dos módulos de dados vazios.

Outro estudo interessante diz respeito à limitação existente no método em relação a modificação de um diagrama EER contido em um módulo de dados, quando esse módulo já está sendo utilizado em uma derivação do diagrama. Poderia ser estudada uma forma de evoluir o diagrama EER de um módulo sem interferir na compatibilidade de diagramas EER derivados de um mesmo DBFD.

Outro trabalho possível seria o aperfeiçoamento da ferramenta de software desenvolvida. Considerando que a ferramenta utiliza ferramentas complementares de suporte, seria interessante se todas as etapas do método fossem reproduzidas na mesma ferramenta para que o fluxo das atividades na ferramenta fosse mais fluído. Ou seja, seria possível desenhar o DBFD na própria ferramenta. Para isso, teria que ser desenvolvida uma parte gráfica para criação dos módulos e ligações dos DBFDs. Nessa nova versão da ferramenta também seria interessante ter a visualização das anotações na imagem do diagrama.

Outro ponto da ferramenta que pode ser melhor estudado é em relação a etapa de geração do *script* SQL de criação ou evolução do banco de dados. Considerando que o algoritmo de transformação de um modelo conceitual em um *script* SQL tem passos com várias possibilidades de decisão, principalmente na evolução do esquema, e o foco deste trabalho não era gerar modelos de dados para criar bancos de dados físicos, então este é um ponto que pode ser melhorado. Desta forma, poderiamos garantir que o *script* gerado poderia ser utilizado de forma automática para criação do banco de dados físico, sem ser necessária a interferência de um projetista de banco de dados.

Além do estudo para se gerar um *script* SQL do zero, outro estudo importante é para a geração de *scripts* SQL de evolução de banco de dados. Este é um estudo mais avançado, mas que traria benefícios não só para o método descrito neste trabalho de mestrado, mas também para outras áreas de banco de dados, como a área de bancos de dados evolutivos.

Considerando que foi possível aplicar o modelo e o método propostos em um domínio de aplicação real, conseguimos mostrar que é viável utilizá-los para modelagem conceitual. Para verificar a vantagem na sua utilização, um trabalho futuro importante seria comparar a modelagem conceitual usando os DBFDs com a modelagem usando os modelos tradicionais visando avaliar o desempenho do modelo em relação aos demais.

E, por fim, um estudo interessante seria utilizar o DBFD para modelagem de modelos de banco de dados em conjunto com a modelagem de artefatos de software em linhas de produtos de software. Ou seja, o objetivo seria fazer a modelagem de software utilizando LPSs, ao mesmo tempo em que o método proposto neste trabalho seria usado para a modelagem do banco de dados referente ao software. Com esse estudo, seria possível analisar se o método proposto pode ser utilizado para modelagem de sistemas de software completos.

Apêndice A

Material do Estudo Exploratório

Este apêndice contém todo o material que foi utilizado durante o estudo exploratório realizado neste trabalho de mestrado. O primeiro documento apresentado é o roteiro de testes, seguido do termo de consentimento que foi assinado por todos os participantes do estudo e do questionário de caracterização dos participantes.

Então, seguem os cenários de tarefas e os questionários de cada etapa, sendo eles: cenário de reprodutibilidade, questionário de reprodutibilidade, cenário de usabilidade, questionário de usabilidade, questionário de aderência da ferramenta ao método, cenário de reprodutibilidade (versão 2) e questionário de reprodutibilidade (versão 2).

Todos os materiais descritos anteriormente estão apresentados nas seções posteriores.

A.1 Roteiro de testes

1. Introdução

Este documento tem como finalidade descrever o protocolo a ser realizado nos testes de reprodutibilidade do método de construção de famílias de esquemas conceituais de banco de dados por meio dos Diagramas de Características de Banco de Dados, de usabilidade da ferramenta DBFD Creator e de aderência da ferramenta DBFD ao método de construção de famílias de esquemas conceituais. Dessa forma, os testes serão divididos em três etapas, e os procedimentos e recursos necessários para a execução de cada etapa do teste estão descritos posteriormente.

2. Objetivos do Teste

Esse teste terá objetivos específicos para cada etapa da validação, como descrito abaixo. Objetivos do teste de reprodutibilidade:

- Obter dados qualitativos sobre a descrição do método.
- Avaliar o nível de detalhamento das etapas do método.
- Verificar o entendimento dos conceitos existentes no método.
- Analisar a facilidade de entendimento e reprodução do método.
- Avaliar a sintaxe das notações do método.

Objetivos do teste de usabilidade:

- Obter dados qualitativos de usabilidade do sistema DBFD Creator.
- Obter dados quantitativos, por meio de:
  - Número de erros para a execução das tarefas.

  - Tempo gasto na execução de cada tarefa.

- Avaliar o nível de satisfação do usuário quanto aos diversos aspectos do sistema.
- Analisar os componentes do layout gráfico.
- Avaliar a navegabilidade do sistema.
- Avaliar a terminologia utilizada no sistema.
- Analisar se as telas da ferramenta facilitam o fluxo do usuário quanto a utiliza.

Objetivos do teste de aderência:

- Obter dados qualitativos quanto a reprodutibilidade do método na ferramenta.
- Analisar o quanto a ferramenta facilita na utilização do método.
- Avaliar se a ferramenta atende a todos os requisitos para executar o método.

3. Participantes do Teste

Para a realização dos testes serão aceitos alunos de graduação ou pós-graduação e graduados ou pós graduados. Esses participantes deverão apresentar o seguinte perfil:

- Faixa etária 20-50 anos de idade
- Conhecimentos básicos de modelagem conceitual de banco de dados
- Conhecimentos básicos de Engenharia de Software

O primeiro passo para participar do teste é preencher o questionário de caracterização do participante.

4. Protocolo e Procedimentos

O protocolo que será seguido durante os testes pode ser descrito pelo passo-a-passo abaixo:

1. O avaliador recebe o participante e lhe explica os objetivos do teste.
2. O avaliador dá início ao teste de reprodutibilidade do método.
3. O avaliador explica o método ao participante e lhe entrega um documento de definição do método para auxiliá-lo na tarefa.
4. O avaliador entrega o roteiro de tarefas do teste de reprodutibilidade ao participante.
5. O participante lê cada tarefa e a realiza.
6. Quando o participante finalizar as tarefas, o avaliador pede que ele responda ao questionário de avaliação da reprodutibilidade do método. O avaliador se afasta do participante para que ele se sinta mais à vontade.
7. O participante preenche o questionário com calma, perguntando ao avaliador caso exista alguma dúvida sobre o questionário e, ao final do preenchimento, entrega o questionário ao avaliador.
8. Finalizado o teste de reprodutibilidade do método, o avaliador dá início ao teste de usabilidade da ferramenta.
9. O avaliador introduz o participante às funcionalidades da ferramenta por meio de um treinamento e, ao final, fornece os manuais do sistema para guiar o participante.

10. O avaliador entrega o roteiro de tarefas e os materiais já prontos que poderão ser utilizados durante o teste de usabilidade ao participante.
11. O participante lê cada tarefa e a realiza.
12. Enquanto o participante realiza as tarefas, o avaliador preenche a sua ficha de avaliação.
13. Quando o participante finalizar as tarefas, o avaliador pede que ele responda ao questionário de avaliação da ferramenta.
14. O participante preenche o questionário e quanto terminar, entrega o questionário ao avaliador.
15. Finalizado o teste de usabilidade da ferramenta, o avaliador dá início ao teste de aderência da ferramenta ao método.
16. O avaliador pede para que o participante leia novamente a descrição do método de criação de família de esquemas conceituais de banco de dados utilizando Diagramas de Características de Banco de Dados (DBFDs).
17. Quando o participante finalizar a leitura, o avaliador pede que ele responda ao questionário.
18. O participante preenche o questionário com calma e entrega o questionário preenchido ao avaliador quando terminar.
19. O avaliador agradece a presença do participante na avaliação, entrega-lhe o brinde de participação e se despede.

5. Materiais do Teste

Os materiais utilizados durante os três testes serão:

1. Instruções ao participante: texto lido pelo avaliador ao participante no início de cada teste.
2. Cenários de tarefas: lista de tarefas que deverão ser executadas pelo participante em cada etapa do teste.
3. Questionário de caracterização dos participantes: questionário que tem como objetivo determinar o perfil do participante, bem como seu nível de conhecimento e tempo de experiência.
4. Termo de consentimento: termo que deverá ser assinado por todos os participantes do estudo, autorizando o uso de suas avaliações e observações no trabalho de mestrado.
5. Questionário de avaliação do método: questionário que tem como objetivo coletar as métricas qualitativas de avaliação da reprodutibilidade do método.
6. Lista de materiais: diagramas EER (Entidade Relacionamento Estendido) e o DBFD do domínio da avaliação para serem utilizados do teste de usabilidade da ferramenta.
7. Questionário de avaliação do sistema: questionário que tem como objetivo coletar as métricas qualitativas e quantitativas de avaliação da usabilidade do sistema.
8. Questionário de avaliação: questionário que tem como objetivo coletar as métricas qualitativas de avaliação da aderência do sistema ao método.

6. Instruções ao Participante

Agradecemos a sua disponibilidade em participar da avaliação do método de criação de famílias de esquemas conceituais de banco de dados e da ferramenta DBFD Creator. Essa avaliação é composta de 3 etapas: teste de reprodutibilidade do método, usabilidade da ferramenta e aderência da ferramenta ao metodo.

A primeira etapa de avaliação é composta de 8 tarefas a serem realizadas em papel e de maneira simples. Leia cada tarefa com muita atenção e só a execute quando estiver certo de que entendeu completamente. Saiba que o avaliador poderá fazer perguntas durante a execução das tarefas, sempre que achar necessário. Após realizar as tarefas, será solicitado que você responda a um questionário sobre diversos pontos do método. Essa etapa de avaliação deve durar no máximo uma hora.

A segunda etapa de avaliação é composta de 9 tarefas a serem realizadas na ferramenta DBFD Creator. Para a avaliação da usabilidade da ferramenta, será necessário coletar todas as informações sobre as ações e as impressões do usuário enquanto o mesmo utiliza o sistema. Portanto, diga sempre o que está pensando durante a execução de uma tarefa, seja fazendo críticas, apontando os pontos positivos e negativos ou indicando os acontecimentos que você esperava mas que não ocorreram.

Execute cada uma das tarefas com muita atenção. Você pode realizar as tarefas da maneira em que achar melhor, e não se preocupe, pois quem está sendo avaliado é o sistema, e não o seu processo de aprendizado. A única obrigatoriedade é que se utilize os materiais prontos disponibilizados para o participante. Após realizar as tarefas, será solicitado que você responda a um questionário sobre diversos aspectos de usabilidade do sistema. Essa etapa da avaliação deve durar no máximo 30 minutos.

Para iniciarmos a última etapa de avaliação, pedimos que você leia novamente à descrição do método com muita calma e atenção. Depois, você já pode começar a preencher o questionário, que contém uma pergunta para cada passo da metodologia, de forma a verificar a qualidade da funcionalidade correspondente a esse passo na ferramenta. Não se esqueça de responder às questões abertas, que nos ajudarão a avaliar pontos diversos que possam surgir pelos participantes. Essa etapa de avaliação deve durar entre 15 e 30 minutos.

Lembre-se de que todos os dados preenchidos pelos participantes desta avaliação são anônimos. Então, nenhum dado pessoal será exposto nos resultados finais do estudo. Antes de começarmos, você gostaria de fazer alguma pergunta?

A.2 Termo de Consentimento

Termo de Consentimento

Este documento visa afirmar minha participação no estudo de avaliação dos resultados obtidos no projeto de Mestrado de Larissa Cristina Moraes, do departamento de Ciências da Computação, do Instituto de Matemática e Estatística da Universidade de São Paulo. Este projeto tem como objetivo principal o desenvolvimento de uma metodologia para a criação de famílias de esquemas conceituais de banco de dados por meio do uso de "Diagramas de Características de Banco de Dados (DBFDs)", que estão sendo propostos no projeto de mestrado. Minha contribuição consiste em participar do estudo de reprodutibilidade do método DBFD e de usabilidade do sistema *web que apoia o uso dos DBFDs.* A sessão de teste poderá ser gravada em áudio e vídeo apenas para fins de uso acadêmico do projeto. Esses dados não serão repassados a terceiros nem usados para outros fins.

"Declaro estar ciente e de acordo com as informações constantes neste 'Termo de Consentimento', e também declaro entender que a análise das informações coletadas nesse estudo poderão ser publicadas em veículos de divulgação científica, sob a condição de que o meu nome ou outros dados de identificação não serão divulgados em momento algum. Poderei pedir, a qualquer momento, esclarecimentos sobre esta Pesquisa; me recusar a dar informações que julgue prejudiciais a minha pessoa; solicitar a não inclusão em documentos de quaisquer informações que já tenha fornecido e desistir de participar da Pesquisa, sem qualquer tipo de penalidade. Fico ciente também de que uma cópia deste termo permanecerá arquivada com os Pesquisadores responsáveis por esta Pesquisa."

São Paulo, ____ de _____ de 201__

Nome do Participante:_____
Endereço: _____
Email e/ou telefone de contato:_____
Assinatura do Participante:_____

Nome do Pesquisador: _____
E-mail: _____
Assinatura do Pesquisador _____

A.3 Questionário de Caracterização dos Participantes

Questionário de caracterização dos participantes

Este questionário tem como objetivo principal classificar os participantes do estudo. Outro objetivo secundário é a de encontrar um horário em que possa ser feito o treinamento e a avaliação com a maioria dos participantes.

* Required

1. **Nome** *

.....

2. **Sexo**
   Mark only one oval.

   ◯ Feminino
   ◯ Masculino

3. **Idade**

.....

4. **Escolaridade**
   Mark only one oval.

   ◯ Ensino Médio
   ◯ Graduação
   ◯ Mestrado
   ◯ Doutorado
   ◯ Especialização

5. **Profissão**

.....

6. **Área de atuação**

.....

7. **Tempo de Experiência**
   Mark only one oval.

   ◯ 1 a 3 anos
   ◯ 3 a 5 anos
   ◯ Mais de 5 anos

8. **Você tem experiência com modelagem conceitual de banco de dados?**
Mark only one oval.

- ◯ Sim
- ◯ Não

9. **Você tem experiência com diagramas de classe UML?**
Mark only one oval.

- ◯ Sim
- ◯ Não

10. **Você conhece o conceito de Linhas de Produtos de Software?**
Mark only one oval.

- ◯ Sim
- ◯ Não

11. **Qual seu nível de conhecimento do Modelo Entidade Relacionamento Estendido?**
Mark only one oval.

- ◯ Não conheço
- ◯ Conheço pouco
- ◯ Conheço e uso, mas não me considero um expert
- ◯ Uso frequentemente e me considero um expert

12. **Qual seu nível de conhecimento de Diagramas de Características?**
Mark only one oval.

- ◯ Não conheço
- ◯ Conheço pouco
- ◯ Conheço e uso, mas não me considero um expert
- ◯ Uso frequentemente e me considero um expert

13. **Em qual data você participou do estudo?**
Check all that apply.

- ☐ 24/08/2016 - 15:00 às 19:00
- ☐ 25/08/2016 - 16:00 às 20:00

A.4 Tarefas do Teste de Reprodutibilidade do Método

Este estudo tem como principal objetivo analisar a reprodutibilidade do método de criação de famílias de esquemas conceituais de banco de dados por meio dos diagramas de características de banco de dados (DBFDs). A seguir, são listadas as tarefas que você deve realizar. Tais atividades foram idealizadas considerando um cenário hipotético descrito posteriormente.

Descrição do Cenário Hipotético:

Considere um sistema de automatização residencial para uma "casa inteligente"que contém um painel de controle para acessar as funções automatizadas da casa. O painel de controle tem um link de acesso, para que as funções automatizadas possam ser acessadas onlines e uma pessoa responsável para acessá-lo. Quando acessado, o painel de controle exibe as imagens das câmeras em tempo real. As câmeras tem um modelo e código de série.

A casa contém sensores e atuadores. Os atuadores podem ser de diferentes tipos: climatizadores, interruptores de luz e sirenes. Cada atuador tem um tipo e um status, que pode ser ligado ou desligado. Um interruptor de luz tem um tipo, que pode ser dimmer ou normal, e uma voltagem. O climatizador tem um modelo e uma potência e pode ser do tipo ar condicionado ou aquecedor. O tipo ar condicionado tem funções, como ventilar ou refrigerar, enquanto que o aquecedor tem um timer. E cada sirene tem um tipo de som, um modelo e um nível de intensidade sonora.

Os sensores podem ser dos tipos: termostato, detector de presença ou medidor de luminosidade. Todo sensor possui localização, código, tipo e modelo. O termostato contém a informação de temperatura em graus Celsius. O detector de presença registra a informação binária de presença (sim ou não). E o medidor de luminosidade registra a informação do período do dia (manhã, tarde ou noite) e da luminosidade (baixa, média ou alta).

O sistema de automatização armazena as regras que podem existir para tornar a casa inteligente. As regras são armazenadas no painel de controle. Cada regra é baseada nos dados de um sensor e modifica o status de um atuador. Portanto, uma regra contém o código do atuador, o código do sensor e a lógica para a regra. A regra contém também uma data de criação e uma data de modificação.

Na casa inteligente, o climatizador é ativado pelo termostato, ou seja, quando a temperatura aumenta, o ar condicionado é ligado ou quando a temperatura diminui muito, é ligado o aquecedor. Portanto, quando temos um climatizador na casa, é necessário um termostato. Normalmente uma casa tem somente um aquecedor ou ar condicionado, de acordo com o clima da região onde a casa se encontra.

Já o interruptor é ligado ou desligado quando detecta a mudança de luminosidade da casa. Ou seja, quando a casa está escura, o interruptor é ligado e quando a casa está clara, o interruptor é desligado. Portanto, para ter um interruptor inteligente é necessário um sensor de luminosidade.

Além das câmeras no painel de controle que ajudam na segurança da casa, podem haver sirenes espalhadas pela casa que são ligadas quando as pessoas saem da casa, de forma que ela seja acionada quando detectar a presença de uma pessoa.

Desta forma, considerando este cenário hipotético, a atividade consiste em criar uma família de esquemas conceituais de banco de dados para o domínio de casas inteligentes.

Criação do DBFD

Tarefa 01: Definir quais serão os módulos de dados do seu DBFD.

Tarefa 02: Desenhar um diagrama EER para cada módulo de dados, exceto quando o módulo for um módulo vazio.

Tarefa 03: Criar uma árvore (hierarquia) dos módulos de dados. Lembrando que cada módulo será um nó da árvore, e as arestas serão as relações entre os módulos. Faça as arestas inicialmente

sem tipos, somente para conectar os nós.

Tarefa 04: Definir os tipos das relações entre os módulos de dados da árvore.

Tarefa 05: Criar restrições entre os módulos de dados da árvore definida.

Tarefa 06: Criar as anotações para incluir novos relacionamentos entre entidades de dois módulos de dados relacionados.

Tarefa 07: Criar as anotações para incluir novas especializações ou categorizações entre entidades de dois módulos de dados relacionados.

Tarefa 08: Criar as anotações para alterar uma entidade ou relacionamento de um módulo de dados, incluindo ou excluindo atributos.

Tarefa 09: Destaque os módulos de dados que constitui o núcleo do seu DBFD com um tracejado.

A.5 Questionário de Reprodutibilidade do Método

Questionário de Avaliação da Reprodutibilidade do Método

* Required

1. **Identificação do Participante (nome) ***

.....

Reprodutibilidade do Método

2. **Para cada uma das questões a seguir, escolha o seu nível de concordância:**
Mark only one oval per row.

	Discordo totalmente	Discordo	Neutro	Concordo	Concordo totalmente
1. Achei o nível de detalhamento das etapas do método bom.	⬭	⬭	⬭	⬭	⬭
2. Achei que os conceitos de base, como módulo, relação e anotação, estão suficientemente explicados.	⬭	⬭	⬭	⬭	⬭
3. Achei que as etapas do método poderiam ser subdivididas em outras etapas para facilitar a sua aplicação.	⬭	⬭	⬭	⬭	⬭
4. Achei que a gramática da linguagem utilizada para criar as anotações fácil de ser compreendida.	⬭	⬭	⬭	⬭	⬭
5. Os símbolos de notação de relações e restrições são fáceis de serem compreendidos e usados.	⬭	⬭	⬭	⬭	⬭
7. É fácil definir os módulos de dados.	⬭	⬭	⬭	⬭	⬭
8. É fácil criar as relações e restrições entre os módulos de dados.	⬭	⬭	⬭	⬭	⬭
9. É fácil escrever as anotações para as relações e restrições.	⬭	⬭	⬭	⬭	⬭
6. É fácil seguir os passos do método para se criar um DBFD.	⬭	⬭	⬭	⬭	⬭
10. Eu usaria o método para criar modelos de bancos de dados.	⬭	⬭	⬭	⬭	⬭
11. Eu acho que o método não me ajuda a criar esquemas conceituais customizáveis.	⬭	⬭	⬭	⬭	⬭
12. Foram dados os treinamentos adequados para que eu pudesse usar o método proposto.	⬭	⬭	⬭	⬭	⬭

Comentários

3. **Deixe seus comentários sobre a descrição do método.**

4. **Deixe suas observações e críticas sobre o método.**

A.6 Tarefas do Teste de Usabilidade da Ferramenta

Este estudo tem como principal objetivo analisar a usabilidade do DBFD Creator. A usabilidade é um termo usado para definir a facilidade com que as pessoas podem empregar uma ferramenta ou objeto a fim de realizar uma tarefa específica. Neste contexto, a seguir, são listadas as tarefas que você deve realizar a partir do uso do DBFD Creator. Considere o cenário hipotético utilizado na avaliação da reprodutibilidade do método de criação de famílias de esquemas conceituais de banco de dados para realizar este estudo.

Para realizar as tarefas na ferramenta DBFD Creator, considere os anexos do diagrama de característica criado no plugin do Eclipse FeatureIDE e dos diagramas Entidade Relacionamento Estendido criados na ferramenta EERCase. Esses anexos correspondem a uma das possíveis modelagens para o cenário proposto no teste de avaliação da reprodutibilidade do método e são usados como base para o teste de usabilidade da ferramenta.

Descrição das Tarefas:

Para as atividades listadas a seguir, considere os arquivos em anexo para o diagrama de características e também para os modelos conceituais (diagramas Entidade Relacionamento Estendido – EER).

Criação e Configuração Inicial do Diagrama

Tarefa 01: Criar um novo DBFD com o nome SmartHome-<seu_nome> na página inicial de DBFDs.

Tarefa 02: Importar o diagrama de características do arquivo [smart_home.velvet] e visualizar se os módulos de dados e relações presentes no diagrama foram carregados corretamente.

Tarefa 03: Importar o diagrama EER do arquivo com extensão .eer correspondente à cada módulo de dados.

Tarefa 04: Adicionar dois novos módulos de dados no diagrama: módulo aquecedor e módulo ar condicionado. A relação entre o módulo pai climatizador e esses dois módulos filhos será do tipo alternativa. Após concluir a tarefa de adicionar os módulos, importe os diagramas EER para esses dois novos módulos.

Tarefa 05: Adicionar uma nova restrição ao diagrama do tipo requer entre o módulo câmera e o módulo painel de controle.

Tarefa 06: Criar a lista de anotações do arquivo anotações.doc para as todas as relações existentes no seu diagrama.

Tarefa 07: Finalizar o DBFD, indo para a página de criação de uma configuração a partir do DBFD criado nessa primeira etapa.

Criação de uma Configuração do Diagrama

Tarefa 08: Suponha que você deseja que sua casa inteligente contenha os módulos do painel de controle, dos atuadores, da câmera, das regras e dos sensores. Crie uma configuração considerando esses módulos para gerar o modelo de banco de dados customizado. Lembre-se de corrigir os problemas de conflito da configuração se existirem.

Gerar Esquema de Banco de Dados Customizado

Tarefa 09: Gerar o diagrama EER completo para a configuração criada e verificar se ele condiz com os módulos e relações selecionados.

Tarefa 10: Gerar o script SQL para a configuração criada e verificar se ele realmente possibilita a criação de um banco de dados físico.

A.7 Questionário de Usabilidade da Ferramenta

Questionário de Usabilidade do DBFD Creator

* Required

1. **Identificação do Participante** *

.....

Usabilidade

Esse questionário é uma tradução da escala "System Usability Scale (SUS)" - Brooke, John. "SUS-a quick and dirty usability scale." Usability evaluation in industry 189. 194 (1996).4-7

2. **Para cada uma das questões a seguir, escolha seu nível de concordância:** *

Mark only one oval per row.

	Discordo totalmente	Discordo	Neutro	Concordo	Concordo totalmente
1. Acho que gostaria de usar o sistema com frequência.	◯	◯	◯	◯	◯
2. Achei o sistema desnecessariamente complexo.	◯	◯	◯	◯	◯
3. Achei o sistema fácil de usar.	◯	◯	◯	◯	◯
4. Achei que seria necessário o apoio de um especialista para poder usar o sistema.	◯	◯	◯	◯	◯
5. As funções do sistema estavam bem integradas.	◯	◯	◯	◯	◯
6. Achei o sistema muito inconsistente.	◯	◯	◯	◯	◯
7. Imagino que a maioria das pessoas aprenderiam a usar o sistema rapidamente.	◯	◯	◯	◯	◯
8. Achei o sistema muito complicado de usar.	◯	◯	◯	◯	◯
9. Eu me senti muito confiante ao utilizar o sistema.	◯	◯	◯	◯	◯
10. Eu preciso aprender um monte de coisas antes de continuar usando o sistema.	◯	◯	◯	◯	◯

Impressões Gerais

3. **Qual é a sua impressão geral sobre o sistema?**

4. **O que você mais gostou no sistema?**

5. **O que você menos gostou no sistema?**

6. **Se você fosse o desenvolvedor do sistema, qual seria a primeira coisa que você faria para melhorá-lo?**

7. **Existe algo que você sinta que está faltando nas funcionalidades atuais do sistema?**

8. **Se você fosse descrever o sistema para um colega em uma ou duas frases, o que você diria?**

9. **Você tem outros comentários finais? Se sim, escreva-os.**

A.8 Questionário de Aderência da Ferramenta ao Método

Questionário de Aderência do DBFD Creator ao Método

* Required

1. **Identificação do participante (nome) ***

   .....

2. **Selecione o nível de dificuldade para realizar cada uma das seguintes tarefas na ferramenta DBFD Creator: ***

   Mark only one oval per row.

	Muito fácil	Fácil	Regular	Difícil	Muito difícil
1. Criar módulos de dados.	◯	◯	◯	◯	◯
2. Associar um diagrama EER a um módulo de dados.	◯	◯	◯	◯	◯
3. Criar relações e restrições entre módulos de dados.	◯	◯	◯	◯	◯
4. Criar módulos de dados vazios.	◯	◯	◯	◯	◯
5. Escrever anotações para as relações e restrições.	◯	◯	◯	◯	◯
6. Criar um esquema de banco de dados específico a partir de um DBFD.	◯	◯	◯	◯	◯
7. Criar uma alteração para evoluir um banco de dados por meio de um DBFD.	◯	◯	◯	◯	◯

3. **Você considera a ferramenta adequada ao método proposto?**

   Mark only one oval.

   ◯ Totalmente
   ◯ Neutro
   ◯ Parcialmente

4. **Você achou que a navegabilidade pela ferramenta é correspondente aos passos do método?**

   Mark only one oval.

   ◯ Totalmente
   ◯ Neutro
   ◯ Parcialmente

5. **O que você mudaria na ferramenta para que essa fosse mais fiel ao método?**

6. **Deixe seus comentários sobre a aderência da ferramenta ao método.**

7. **Deixe suas sugestões e críticas para que possamos melhorar a ferramenta.**

A.9 Tarefas do Teste de Reprodutibilidade do Método - versão 2

Este estudo tem como principal objetivo analisar a reprodutibilidade do método de criação de famílias de esquemas conceituais de banco de dados por meio dos diagramas de características de banco de dados (DBFDs). A seguir, são listadas as tarefas que você deve realizar. Tais atividades foram idealizadas considerando um cenário hipotético descrito posteriormente.

Descrição do Cenário Hipotético:

Considere um sistema de automatização residencial para uma "casa inteligente"que contém um painel de controle para acessar as funções automatizadas da casa. O painel de controle tem um link de acesso, para que as funções automatizadas possam ser acessadas onlines e uma pessoa responsável para acessá-lo. Quando acessado, o painel de controle exibe as imagens das câmeras em tempo real. As câmeras tem um modelo e código de série.

A casa contém sensores e atuadores. Os atuadores podem ser de diferentes tipos: climatizadores, interruptores de luz e sirenes. Cada atuador tem um tipo e um status, que pode ser ligado ou desligado. Um interruptor de luz tem um tipo, que pode ser dimmer ou normal, e uma voltagem. O climatizador tem um modelo e uma potência e pode ser do tipo ar condicionado ou aquecedor. O tipo ar condicionado tem funções, como ventilar ou refrigerar, enquanto que o aquecedor tem um timer. E cada sirene tem um tipo de som, um modelo e um nível de intensidade sonora.

Os sensores podem ser dos tipos: termostato, detector de presença ou medidor de luminosidade. Todo sensor possui localização, código, tipo e modelo. O termostato contém a informação de temperatura em graus Celsius. O detector de presença registra a informação binária de presença (sim ou não). E o medidor de luminosidade registra a informação do período do dia (manhã, tarde ou noite) e da luminosidade (baixa, média ou alta).

O sistema de automatização armazena as regras que podem existir para tornar a casa inteligente. As regras são armazenadas no painel de controle. Cada regra é baseada nos dados de um sensor e modifica o status de um atuador. Portanto, uma regra contém o código do atuador, o código do sensor e a lógica para a regra. A regra contém também uma data de criação e uma data de modificação.

Na casa inteligente, o climatizador é ativado pelo termostato, ou seja, quando a temperatura aumenta, o ar condicionado é ligado ou quando a temperatura diminui muito, é ligado o aquecedor. Portanto, quando temos um climatizador na casa, é necessário um termostato. Normalmente uma casa tem somente um aquecedor ou ar condicionado, de acordo com o clima da região onde a casa se encontra.

Já o interruptor é ligado ou desligado quando detecta a mudança de luminosidade da casa. Ou seja, quando a casa está escura, o interruptor é ligado e quando a casa está clara, o interruptor é desligado. Portanto, para ter um interruptor inteligente é necessário um sensor de luminosidade.

Além das câmeras no painel de controle que ajudam na segurança da casa, podem haver sirenes espalhadas pela casa que são ligadas quando as pessoas saem da casa, de forma que ela seja acionada quando detectar a presença de uma pessoa.

Desta forma, considerando este cenário hipotético, a atividade consiste em criar uma família de esquemas conceituais de banco de dados para o domínio de casas inteligentes.

Criação do DBFD

Tarefa 01: Criar um diagrama EER completo para o domínio de aplicação.

Tarefa 02: Determinar os módulos de dados no diagrama EER completo. Dividir o diagrama EER, obtendo uma partição do diagrama completo para cada módulo de dados.

Tarefa 03: Criar as ligações entre os módulos de dados, com os tipos escolhidos para cada ligação.

Tarefa 04: Escrever as anotações para incluir novos relacionamentos entre entidades de dois módulos de dados relacionados (se existir em seu DBFD).

Tarefa 05: Escrever as anotações para incluir novas especializações ou categorizações entre entidades de dois módulos de dados relacionados (se existir em seu DBFD).

Tarefa 06: Escrever as anotações para alterar uma entidade ou relacionamento de um módulo de dados, incluindo ou excluindo atributos (se existir em seu DBFD).

Tarefa 07: Destaque os módulos de dados que constitui o núcleo do seu DBFD com um tracejado.

A.10 Questionário de Reprodutibilidade do Método - versão 2

Questionário de Avaliação da Reprodutibilidade do Método

* Required

1. **Identificação do Participante (nome) ***

.....

Reprodutibilidade do Método

2. **Para cada uma das questões a seguir, escolha o seu nível de concordância:**
   Mark only one oval per row.

	Discordo totalmente	Discordo	Neutro	Concordo	Concordo totalmente
1. Achei o nível de detalhamento das etapas do método bom.	◯	◯	◯	◯	◯
2. Achei que as etapas do método poderiam ser subdivididas em outras etapas para facilitar a sua aplicação.	◯	◯	◯	◯	◯
3. É fácil definir os módulos de dados.	◯	◯	◯	◯	◯
4. É fácil criar as ligações de consiste-em e restrição entre os módulos de dados.	◯	◯	◯	◯	◯
5. É fácil escrever as anotações para as ligações.	◯	◯	◯	◯	◯
6. É fácil seguir os passos do método para se criar um DBFD.	◯	◯	◯	◯	◯
7. Eu usaria o método para criar modelos de bancos de dados customizados.	◯	◯	◯	◯	◯
8. Eu acho que o método não me ajuda a criar esquemas conceituais customizáveis.	◯	◯	◯	◯	◯

Comentários

3. **O que você achou da mudança do passo a passo do método para a abordagem top-down?** *

.....

.....

.....

.....

.....

4. **Deixe seus comentários sobre a nova descrição do método.**

.....

.....

.....

.....

.....

Referências Bibliográficas

[ABS00] Serge Abiteboul, Peter Buneman e Dan Suciu. *Data on the Web: From Relations to Semistructured Data and XML*. Morgan Kaufmann Publishers Inc., 2000. 17

[AR07] Elena Alana e Ana Isabel Rodríguez. Domain engineering methodologies survey. Relatório técnico, 2007. 11

[AS06] S.W. Ambler e P.J. Sadalage. *Refactoring Databases: Evolutionary Database Design*. A Martin Fowler signature book. Addison Wesley, 2006. 16

[AZDT11] Lamia Abo Zaid e Olga De Troyer. Towards modeling data variability in software product lines. Em *Enterprise, Business-Process and Information Systems Modeling*, páginas 453–467. Springer Berlin Heidelberg, 2011. 16

[BFK+99] Joachim Bayer, Oliver Flege, Peter Knauber, Roland Laqua, Dirk Muthig, Klaus Schmid, Tanya Widen e Jean-Marc DeBaud. Pulse: A methodology to develop software product lines. Em *Proceedings of the 1999 Symposium on Software Reusability*, páginas 122–131. ACM, 1999. 12

[BHST04] Yves Bontemps, Patrick Heymans, Pierre-Yves Schobbens e Jean-Christophe Trigaux. Semantics of foda feature diagrams. Em *Proceedings SPLC 2004 Workshop on Software Variability Management for Product Derivation – Towards Tool Support*, páginas 48–58, 2004. 11

[BOR09] Joerg Bartholdt, Roy Oberhauser e Andreas Rytina. Addressing data model variability and data integration within software product lines. *International Journal On Advances in Software*, páginas 84–100, 2009. 16

[Bow14] Ann Bowling. *Research methods in health: investigating health and health services*. McGraw-Hill Education (UK), 2014. 13

[bpm] BPMN diagram homepage. http://www.bpmn.org/. Acesso em: 17/10/2016. 25

[BPSM+98] Tim Bray, Jean Paoli, C Michael Sperberg-McQueen, Eve Maler e François Yergeau. Extensible markup language (xml). *World Wide Web Consortium Recommendation*, 1998. 2

[Bro96] John Brooke. Sus: A quick and dirty usability scale, 1996. 13, 58, 62, 63

[BSRC] David Benavides, Sergio Segura e Antonio Ruiz-Cortes. Automated analysis of feature models 20 years later: A literature review. http://www.lsi.us.es/~dbc/en/?Research. Acesso em: 17/10/2016. xi, 12

[Car] Code Analysis, Repository & Modelling for e-Neuroscience (CARMEN) homepage. http://www.carmen.org.uk/. Acesso em: 17/10/2016. 42

[CB11] Lianping Chen e Muhammad Ali Babar. A systematic review of evaluation of variability management approaches in software product lines. *Information and Software Technology*, páginas 344–362, 2011. 10

[Che76] Peter Pin-Shan Chen. The entity-relationship model–toward a unified view of data. *ACM Trans. Database Syst.*, páginas 9–36, 1976. 6

[CN01] Paul Clements e Linda Northrop. *Software Product Lines: Practices and Patterns*. Addison-Wesley Longman Publishing Co., Inc., 2001. 10

[CSF00] Rainer Conrad, Dieter Scheffner e J. Christoph Freytag. Xml conceptual modeling using uml. Em *Proceedings of the 19th International Conference on Conceptual Modeling*, páginas 558–574. Springer-Verlag, 2000. 17

[Dem01] P. Demo. *Pesquisa E Informação Qualitativa*. Papirus, 2001. 12

[dja] Django framework homepage. https://www.djangoproject.com/. Acesso em: 12/10/2016. 36

[dM01] L.G. de Mello. *Antropologia cultural: iniciação, teoria e temas*. Vozes, 2001. 12

[dO01] S.L. de Oliveira. *Tratado de metodologia científica: projetos de pesquisas, TGI, TCC, monografias, dissertações e teses*. Pioneira Thomson Learning, 2001. 12

[dra] Draw io. https://www.draw.io/. Acesso em: 17/10/2016. 36

[DSFG13] J.C. Dos Santos Filho e S.S. Gamboa. *Pesquisa educacional: quantidade-qualidade*. Cortez Editora, 2013. 12

[EER] EERCASE homepage. https://sites.google.com/a/cin.ufpe.br/eercase/. Acesso em: 17/10/2016. 36

[EN10] Ramez Elmasri e Shamkant Navathe. *Fundamentals of Database Systems*. Addison-Wesley Publishing Company, 2010. 1, 5, 6, 7, 8, 36

[erd] ERD plus. https://erdplus.com. Acesso em: 17/10/2016. 36

[GB05] SK Grove e N Burns. The practice of nursing research: Conduct, critique, & utilization, 2005. 13

[gea] Gears software. http://www.biglever.com/. Acesso em: 17/10/2016. 35

[GFd98] Martin L Griss, John Favaro e Massimo d'Alessandro. Integrating feature modeling with the rseb. Em *Software Reuse Proceedings.*, páginas 76–85, 1998. 11, 35

[GS12] Rita Ganguly e Anirban Sarkar. Evaluations of conceptual models for semi-structured database system. *International Journal of Computer Applications*, 2012. 17

[Har02] M. Harsu. A survey on domain engineering. Relatório Técnico 31, 2002. 9, 10, 11

[HCMD92] Jean-luc Hainaut, Mario Cadelli, Olivier Marchand e Bernard Decuyper. Database case tool architecture : Principles for flexible design strategies. Em *Proceedings of the 4th International Conference on Advanced Information System Engineering*, páginas 187–207, 1992. 15

[HTO03] Patrick Heymans, Jean-Christophe Trigaux e First Europe Objectif. Software product lines: State of the art. 2003. 9

[IH87] Boris Iglewicz e David C Hoaglin. Use of boxplots for process evaluation. *Journal of Quality Technology*, páginas 180–190, 1987. 65

[JGJ97] Ivar Jacobson, Martin Griss e Patrik Jonsson. Software reuse architecture, process and organization for business success. Em *Computer Systems and Software Engineering, 1997., Proceedings of the Eighth Israeli Conference on*, páginas 86–89. IEEE, 1997. 11

[jsoa] JSON homepage. http://www.json.org/. Acesso em: 17/10/2016. 2

[jsob] JSON schema homepage. http://json-schema.org/. Acesso em: 17/10/2016. 2

[KC93] Jurek Kirakowski e Mary Corbett. Sumi: the software usability measurement inventory. *British Journal of Educational Technology*, páginas 210–212, 1993. 13

[KCH+90] Kyo C. Kang, Sholom G. Cohen, James A. Hess, William E. Novak e A. Spencer Peterson. A feature-oriented domain analysis (FODA) feasibility study. Relatório técnico, 1990. 9, 10

[KK13] Niloofar Khedri e Ramtin Khosravi. Handling database schema variability in software product lines. Em *Proceedings of the 20th Asia-Pacific Software Engineering Conference*, páginas 331–338, 2013. 17

[KKL+98] Kyo C. Kang, Sajoong Kim, Jaejoon Lee, Kijoo Kim, Euiseob Shin e Moonhang Huh. FORM: A feature-oriented reuse method with domain-specific reference architectures. *Annals of Software Engineering*, páginas 143–168, 1998. 9, 12

[KPP+02] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, K. El Emam e J. Rosenberg. Preliminary guidelines for empirical research in software engineering. *IEEE Transactions on Software Engineering*, páginas 721–734, 2002. 12, 55

[LCC94] Chien-Tsai Liu, Panos K. Chrysanthis e Shi-Kuo Chang. Database schema evolution through the specification and maintenance of changes on entities and relationships. Em *Entity-Relationship Approach, Business Modelling and Re-Engineering*, páginas 13–16, 1994. 15

[LKL02] Kwanwoo Lee, KyoC. Kang e Jaejoon Lee. Concepts and guidelines of feature modeling for product line software engineering. Em *Software Reuse: Methods, Techniques, and Tools*, páginas 62–77. 2002. 9

[LLY06] HongXing Liu, YanSheng Lu e Qing Yang. Xml conceptual modeling with xuml. Em *Proceedings of the 28th International Conference on Software Engineering*, páginas 973–976. ACM, 2006. 17

[LSR07] Frank J. van der Linden, Klaus Schmid e Eelco Rommes. *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*. 2007. 9, 10

[luc] ER diagram tool - Lucid Chart, howpublished = https://www.lucidchart.com/pages/tour/er_diagram_tool, note = Acesso em: 17/10/2016. 36

[Man04] Murali Mani. *EReX: A Conceptual Model for XML*. Springer Berlin Heidelberg, 2004. 17

[MB15] Larissa Cristina Moraes e Kelly Rosa Braghetto. Creating families of conceptual database schemas using database feature diagrams (DBFDs). *Anais do XXX Simpósio Brasileiro de Banco de Dados*, 2015. 3, 70

[MM07] Gerson Luís Russo Moysés e Roberto Giro Morri. Coleta de dados para a pesquisa acadêmica: um estudo sobre a elaboração, a validação e a aplicação eletrônica de questionário. 2007. 12

[MS10] Qaiser Munir e Muhammad Shahid. Software product line: Survey of tools. 2010. 35

[NCWD84] Shamkant Navathe, Stefano Ceri, Gio Wiederhold e Jinglie Dou. Vertical partitioning algorithms for database design. *ACM Trans. Database Syst.*, páginas 680–710, 1984. 16

[Nec07] Martin Necasky. Xsem: A conceptual model for xml. Em *Proceedings of the Fourth Asia-Pacific Conference on Comceptual Modelling - Volume 67*, páginas 37–48. Australian Computer Society, Inc., 2007. 17

[Nem] Neural ElectroMagnetic Ontologies (NEMO) homepage. http://nemo.nic.uoregon.edu/. Acesso em: 17/10/2016. 42

[neu] Centro de Pesquisa, Inovação e Difusão em Neuromatemática (NeuroMat) homepage. http://neuromat.numec.prp.usp.br/. Acesso em: 17/10/2016. 41, 43

[Nor02] Linda M. Northrop. SEI's software product line tenets. *IEEE Softw.*, páginas 32–40, 2002. 9

[Par91] A. Parasuraman. *Marketing Research*. Addison-Wesley Publishing Company, 1991. 12

[PFH+08] Russell A Poldrack, Paul C Fletcher, Richard N Henson, Keith J Worsley, Matthew Brett e Thomas E Nichols. Guidelines for reporting an fMRI study. *Neuroimage*, páginas 409–414, 2008. 41

[PK01] Shari Lawrence Pfleeger e Barbara A Kitchenham. Principles of survey research: part 1: turning lemons into lemonade. *ACM SIGSOFT Software Engineering Notes*, páginas 16–18, 2001. 12, 13

[Psa00] Giuseppe Psaila. Erx: A conceptual model for xml documents. Em *Proceedings of the 2000 ACM Symposium on Applied Computing - Volume 2*, páginas 898–903. ACM, 2000. 17

[pur] Pure::variant software. http://www.pure-systems.com/. Acesso em: 17/10/2016. 35

[pyt] Python language homepage. https://www.python.org/. Acesso em: 17/10/2016. 36

[RCR93] John F. Roddick, Noel G. Craske e Thomas J. Richards. A taxonomy for schema versioning based on the relational and entity relationship models. Em *Lecture Notes in Computer Science*, páginas 139–150. Springer-Verlag, 1993. 15

[RJB99] James Rumbaugh, Ivar Jacobson e Grady Booch, editors. *The Unified Modeling Language Reference Manual*. 1999. 11

[Sar12] Anirban Sarkar. Conceptual level design of semi-structured database system: Graph-semantic based approach. *CoRR*, 2012. 18

[SCK+96] Mark Simos, Dick Creps, Carol Klingler, Larry Levine e Dean Allemang. Organization domain modeling (odm) guidebook. Relatório técnico, 1996. 12

[SKR+09] Norbert Siegmund, Christian Kästner, Marko Rosenmüller, Florian Heidenreich, Sven Apel e Gunter Saake. Bridging the gap between variability in client application and database schema, 2009. 16

[SMD03] Arijit Sengupta, Sriram Mohan e Rahul Doshi. Xer – extensible entity relationship modeling. Em *in Proceedings of the XML 2003 Conference*, páginas 140–154, 2003. 17

[SP92] Ben Shneiderman e Catherine Plaisant. Designing the user interface: Strategies for effective human-computer interaction. *ACM SIGBIO Newsletter*, página 6, 1992. 13

[spl] S.P.L.O.T. software. http://www.splot-research.org/. Acesso em: 17/10/2016. 35

[TCG+93] Abdullah Uz Tansel, James Clifford, Shashi Gadia, Sushil Jajodia, Arie Segev e Richard Snodgrass, editors. *Temporal Databases: Theory, Design, and Implementation*. Benjamin-Cummings Publishing Co., Inc., 1993. 16

[TKB+14] Thomas Thüm, Christian Kästner, Fabian Benduhn, Jens Meinicke, Gunter Saake e Thomas Leich. FeatureIDE: An extensible framework for feature-oriented software development. *Sci. Comput. Program.*, 2014. 35, 36

[TLNJ11] Toby J. Teorey, Sam S. Lightstone, Tom Nadeau e H. V. Jagadish. *Database Modeling and Design: Logical Design*. 2011. 5, 8

[uml] Unified modeling language homepage. http://www.uml.org/. Acesso em: 17/10/2016. 1

[vdLM95] F.J. van der Linden e J.K. Muller. Creating architectures with building blocks. *Software, IEEE*, páginas 51–60, 1995. 9

[VDLP05] Frank Van Der Linden e Klaus Pohl. *Software Product Line Engineering: Foundations, Principles, and Techniques*. Springer-Verlag New York, Inc., 2005. xi, 9, 10

[wik] List of neuroscience databases. http://en.wikipedia.org/wiki/List_of_neuroscience_databases. Acesso em: 17/10/2016. 42

[WL99] David M. Weiss e Chi Tau Robert Lai. Software product-line engineering: a family-based software development process. 1999. 12

[WLLD01] Xiaoying Wu, Tok Wang Ling, Mong Li Lee e Gillian Dobbie. Designing semistructured databases using ora-ss model. Em *Proceedings of the Second International Conference on Web Information Systems Engineering (WISE'01) Volume 1 - Volume 1*, páginas 171–. IEEE Computer Society, 2001. 17

[xml] XML schema homepage. https://www.w3.org/XML/Schema. Acesso em: 17/10/2016. 2