UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

MILTON BITTENCOURT DE SOUZA NETO

brModeloWeb:
FERRAMENTA WEB PARA ENSINO E MODELAGEM DE BANCO DE DADOS

FLORIANÓPOLIS
2016

Milton Bittencourt de Souza Neto

brModeloWeb:
FERRAMENTA WEB PARA ENSINO E MODELAGEM DE BANCO DE DADOS

Monografia submetida ao Curso de Sistemas de Informação, do departamento de Informática e Estatística, da Universidade Federal de Santa Catarina para a obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Ronaldo dos Santos Mello.

Florianópolis
2016

Milton Bittencourt de Souza Neto

brModeloWeb:
FERRAMENTA WEB PARA ENSINO E MODELAGEM DE BANCO DE DADOS

Esta Monografia foi julgada e aprovada para a obtenção do Título de “Bacharel em Sistemas de Informação”, e aprovada em sua forma final pelo Curso de Sistemas de Informação.

Florianópolis, 10 de novembro de 2016.

Prof. Frank Siqueira
Coordenador do Curso

Banca Examinadora:

Prof. Renato Cislaghi
Presidente

Prof. Dr. Ronaldo dos Santos Mello
Orientador

Rodrigo Gonçalves

Angelo Augusto Frozza

RESUMO

Este trabalho descreve o desenvolvimento de uma ferramenta Web para auxiliar o processo de projeto de banco de dados (BD) relacional, com ênfase no ensino e aprendizado de técnicas de modelagem de BD baseado no modelo entidade-relacionamento. A principal motivação deste trabalho é a sua utilização em disciplinas da área de BD para atividades práticas de modelagem de dados. O projeto é inspirado na ferramenta *desktop* brModelo. A brModelo é uma ferramenta *desktop* voltada para o desenvolvimento de projeto de banco de dados relacionais, incluindo as etapas conceitual, lógico e físico, com ampla utilização na área de computação em todo o Brasil. A nova ferramenta desenvolvida, batizada inicialmente como brModeloWeb, destaca-se das demais ferramentas disponíveis no mercado de *software* por possuir interface *Web*, e, assim, acessível em qualquer dispositivo com acesso à um navegador, cobrir as três etapas de projeto de BD (conceitual, lógico e físico), realizar a conversão entre modelagens, armazenar modelagens, apresentar tutoriais, interagir com o usuário sempre que necessário durante a realização de um projeto, apresentar interface gráfica intuitiva, além de ser gratuita.

Palavras-Chave: projeto de banco de dados relacional, ferramenta brModeloWeb, brModelo.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ANSI	*American National Standards Institute*
BD	Banco de Dados
CEP	Código de Endereçamento Postal
CNPJ	Cadastro Nacional de Pessoa Jurídica
CPF	Cadastro de Pessoa Física
CSS	*Cascading Style Sheets*
DDL	*Data Description Language*
DER	Diagrama Entidade-Relacionamento
DOM	*Document Object Model*
ER	Entidade-Relacionamento
HTML	*HyperText Markup Language*
JSON	*JavaScript Object Notation*
MVC	*Model-View-Controller*
PNG	*Portable Network Graphics*
REST	*Representational State Transfer*
RF	Requisitos Funcionais
RNF	Requisitos Não-Funcionais
SGBD	Sistemas Gerenciadores de Bancos de Dados
SQL	*Structured Query Language*
SVG	*Scalable Vector Graphics*
UF	Unidade da Federação
URL	*Uniform Resource Locator*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Programas de computadores e sistemas podem ser úteis em diversas situações do cotidiano, tais como: automatizar atividades repetitivas; simplificar tarefas; e auxiliar no ensino e no aprendizado. Dentre as ferramentas computacionais mais úteis e mais utilizadas estão os Sistemas Gerenciadores de Bancos de Dados (SGBD).

Aplicações nos mais diversos domínios geralmente necessitam gerenciar um grande volume de dados, sendo assim, é essencial a utilização de um SGBD, bem como, a realização de um projeto de Banco de Dados (BD).

O projeto de um BD é uma atividade complexa. Para Sillberschatz *et al*. (2006), fatores como o escopo do problema e a experiência do projetista interferem no grau de dificuldade. Sendo assim, esse é um campo pleno de oportunidades para o desenvolvimento de ferramentas que auxiliem o processo de modelagem de um BD, propriamente, e no ensino dos conceitos envolvidos na atividade de projeto.

Ainda segundo Sillberschatz *et al*. (2006), o objetivo do projeto de BD é a construção de um conjunto de esquemas que permita a representação de um dado de forma não redundante e que esse dado possa ser recuperado de forma simples. Este processo é uma atividade contínua que se passa em diferentes níveis de abstração. Em geral, inicia-se com o entendimento do problema, para depois seguir por diferentes fases de modelagem, até chegar à implementação.

Nesse processo, um modelo considerado padrão para a etapa inicial de modelagem é o entidade-relacionamento (ER) (HEUSER, 2009). Proposta por Peter Chen, a técnica permite a representação de uma dada realidade por meio de um modelo ER. A representação gráfica se dá pelo diagrama entidade-relacionamento (DER).

Inúmeras aplicações auxiliam no desenvolvimento e no aprendizado de projetos de BDs relacionais. A ferramenta *brModelo CÂNDIDO (2005)*, desenvolvida pelo Grupo de Banco de Dados da UFSC (GBD/UFSC), tornou-se uma das aplicações *desktop* mais populares no meio acadêmico brasileiro[1]. Apresentada em 2005, a

[1] Conforme o portal *Baixaki* (disponível em: <http://www.baixaki.com.br/>), já houve 303.021 downloads da ferramenta (contagem aferida em 13.09.2016). Há dezenas de vídeos tutoriais no site *Youtube* (disponível em: <https://www.youtube.com/>), feitos por usuários e não pelo criador; e diversas citações no Twitter (disponível em: <https://twitter.com/>), acessíveis pela hasthag #brModelo.

ferramenta nasceu com o intuito de prover uma solução amigável e didática, considerando todos os níveis da modelagem de um BD, desde a definição do esquema conceitual, passando pelo esquema lógico, até a proposta de implementação em um SGBD relacional.

Apesar da atual versão da *brModelo* atender bem ao seu propósito[2], é preciso considerar como a Internet vem modificando a forma de utilização de *softwares*. Aplicações *web* têm facilitado o acesso e a manipulação de informações armazenadas em servidores remotos – a partir de qualquer plataforma com acesso à *Internet*, e a qualquer momento.

Já se faz uso desse tipo de arquitetura quando, por exemplo, usamos aplicações como o *Google Mail*, *Moodle*, *Dropbox* e *Google Docs*. Muitas outras seguem em movimento de migração do *desktop* para a *web*. Mas a lógica de um programa pensado para ser executado em ambiente *desktop* é totalmente diferente da lógica de aplicações *web*. Outro desafio envolvido é que os recursos computacionais muitas vezes são mais limitados em ambiente *web*.

Por outro lado, uma vez vencidos os obstáculos e implementada a migração, o público beneficiado pela aplicação tende a ser bastante ampliado, além de serem proporcionadas as vantagens do acesso e monitoramento remoto aos dados do sistema. Por essas razões, este trabalho propõe o desenvolvimento de uma nova versão da ferramenta *brModelo*, voltada para o aprendizado e para a modelagem de BD relacional em ambiente *web*.

1.1 MOTIVAÇÃO

Diversas aplicações voltadas ao projeto de BDs relacionais estão disponíveis no mercado de *software*. Entre as ferramentas comerciais, é possível citar *DBDesigner*[3], *Erwin*[4] e *Visual Paradigm*[5]. Existem, também, as ferramentas voltadas especificamente para o ensino de projeto de BDs, como é o caso da *LEaRning* (FERREIRA; ARANTES, 2005), *AprendER* (MAGALHÃES et al, 2007) e da própria *brModelo*, entre outras.

---

[2] Segundo avaliação feita no artigo *Uma Análise dos Ambientes de Ensino de Banco de Dados,* disponível em http://homepages.dcc.ufmg.br/~juliana.pereira/papers/2012_SBSI/SBSI.pdf

[3] Disponível em: <https://dbdesigner.net/>. Acesso em: 19 ago. 2016.

[4] Disponível em: <http://erwin.com/>. Acesso em: 19 ago. 2016.

[5] Disponível em: <https://www.visual-paradigm.com/>. Acesso em: 19 ago. 2016.

Entretanto, esse tema é pouco explorado em termos de aplicações *web*. Os poucos sistemas que oferecem o serviço de modelagem em ambiente *web*, além de serem comerciais, em geral, não cobrem todas as etapas de um projeto de BD.

Para o usuário, a facilidade de acesso e a independência do sistema operacional são excelentes vantagens de uma aplicação *web* em relação à sua versão para *desktop*. Essa foi a principal motivação para este trabalho: o desenvolvimento de uma ferramenta *web* inspirada na ferramenta *brModelo*, que facilite ainda mais o aprendizado de projetos de BDs, e de forma totalmente gratuita. Como motivação paralela, este trabalho também visa estimular a cultura do uso de *softwares* de código aberto e novas tecnologias *web*.

1.2 OBJETIVOS

1.2.1 Objetivo geral

O objetivo geral deste trabalho é desenvolver uma ferramenta *web* visando, principalmente, o aprendizado de modelagem de BDs relacionais, baseado em modelagem entidade-relacionamento, por meio de qualquer dispositivo conectado à Internet com a acesso a um browser. Esta ferramenta é denominada *brModelo*Web e cobre todas as etapas da modelagem de um BD: a modelagem conceitual, a lógica e a física.

1.2.2 Objetivos específicos

Os objetivos específicos necessários para se alcançar o objetivo geral são os seguintes:

- Desenvolver estrutura de BD para armazenar modelagens e dados dos usuários;
- Desenvolver controle de acesso;
- Desenvolver módulo de projeto conceitual;
- Desenvolver módulo de projeto lógico;
- Desenvolver módulo de projeto físico;
- Implementar conversões entre módulos;

- Desenvolver um *website* para apresentação do *software*, tutoriais e tópicos de ensino;

1.3 METODOLOGIA

A primeira etapa do trabalho consiste em um levantamento bibliográfico a respeito de desenvolvimento de aplicações *web* e projetos de BDs relacionais. A partir desse levantamento, é feito um estudo teórico sobre as tecnologias envolvidas no processo para se criar o *software* proposto.

A segunda etapa prevê a análise dos trabalhos relacionados ao tema de projeto de BD relacional, para fazer o levantamento de requisitos e características de aplicações do gênero, com o objetivo de se propor uma nova ferramenta que ofereça um diferencial em relação às já existentes.

Por fim, é descrito o ciclo de desenvolvimento da aplicação, da realização um estudo de caso, de um teste de usabilidade e a apresentação dos resultados obtidos.

1.4 ORGANIZAÇÃO DO TRABALHO

Uma vez conhecidos os objetivos, a metodologia e o contexto de surgimento deste projeto de conclusão de curso, os capítulos seguintes se dedicam às etapas de trabalho propostas.

No próximo capítulo, apresenta-se a fundamentação teórica, com breves conceitos acerca da teoria de modelagem de BD e a apresentação da evolução da ferramenta *brModelo*.

O terceiro capítulo reúne uma análise das ferramentas *web* gratuitas e comerciais similares à ferramenta *brModelo*. O processo de desenvolvimento da nova aplicação *web* é descrito no quarto capítulo, quando também se explicam as decisões tecnológicas, as bibliotecas utilizadas e a arquitetura escolhida.

O quinto capítulo, por sua vez, traz um estudo de caso e os resultados. O sexto apresenta uma analise de usabilidade realizada na ferramenta. Por fim, a conclusão reforça a contribuição deste estudo e aponta alguns caminhos para a continuidade do projeto em trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

A ferramenta *brModelo* tem como finalidade facilitar o ensino e o aprendizado de projetos de banco de dados relacionais. Por isso, este capítulo apresenta uma breve revisão dos conceitos de projeto de banco de dados e as três etapas de modelagem. Também são apresentadas as ferramentas que inspiraram o desenvolvimento da versão *web* da *brModelo*, detalhando suas características, pontos fortes e pontos fracos.

2.1 PROJETO DE BANCO DE DADOS

O projeto de um Banco de Dados especifica, basicamente, a estrutura deste último, podendo, em alguns casos, indicar também o seu comportamento. Esse tipo de atividade tem como ponto de partida os requisitos de informação e as regras de negócio inerentes ao domínio do problema, com a utilização de ferramentas de projeto ou modelagem, procurando atender a uma série de critérios de qualidade. A partir da compreensão desses requisitos e dessas regras, Heuser (2009)[6] divide o processo de projeto de BD em três etapas fundamentais: projeto conceitual, projeto lógico e projeto físico.

2.1.1 Projeto conceitual

Nesta fase, é construída a modelagem conceitual, em geral representada de forma gráfica, com base em requisitos extraídos do domínio do problema. O modelo conceitual descreve os dados como os usuários os percebem, ou seja, como uma abstração do mundo real (ELMASRI; NAVATHE, 2011). Neste modelo, são definidos quais dados estão presentes no BD, sem que haja a descrição de como essa representação é arquitetada no SGBD. Dessa forma, há uma abstração em nível de modelo de dados do SGBD.

Ainda, segundo Elmasiri e Navathe (2011), a modelagem conceitual é muito importante no projeto de uma aplicação de BD, podendo determinar o sucesso ou o

[6] Todas as referências a Heuser se reportam à seguinte obra: HEUSER, C. A. **Projeto de banco de dados.** Porto Alegre: Bookman, 2009.

insucesso do projeto final. Dentre os modelos de dados conceituais, o mais difundido e utilizado é o modelo entidade-relacionamento, proposto por Chen[7] (1976).

2.1.1.1 Modelo entidade-relacionamento

O modelo entidade-relacionamento (modelo ER) tornou-se um padrão para a modelagem conceitual, segundo Heuser (2009). Uma das principais vantagens dessa proposta é a técnica de diagramação, que permite representar de forma simples os principais aspectos do projeto de um BD. Basicamente, o modelo baseia-se em entidades e relações. Uma entidade representa um fato do mundo real e seus dados são valores de propriedades associadas a esse fato. Já, as relações, são representações da forma como uma entidade se associa a outras entidades.

Heuser (2009) aponta que os principais conceitos da abordagem ER são: entidade, relacionamento, atributo, generalização/especialização e entidade associativa.

2.1.1.1.1 *Entidade*

Uma entidade é uma abstração de um fato do mundo real para o qual se deseja manter seus dados no BD. Ela é simbolizada no diagrama ER por um retângulo. Cada retângulo representa um conjunto de objetos sobre os quais se deseja guardar informações. A Figura 1 apresenta um exemplo de uma entidade denominada ‘Pessoa’.

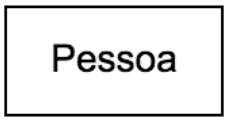


Figura 1 - Entidade ‘Pessoa’
Fonte: elaborado pelo autor.

[7] CHEN, Peter Pin-Shan. **The entity of relationship model:** toward a unified view of data. Massachusetts Institut of Technology. Association for Computing Machinery: Massachusetts, 1976. Disponível em: <http://www.csc.lsu.edu/~chen/pdf/erd-5- pages.pdf>. Acesso em: 20 set. 2016.

2.1.1.1.2 *Relacionamento*

Heuser (2009) define relacionamento como um conjunto de associações entre entidades. O relacionamento é representado por um losango ligado por linhas às entidades que fazem parte do relacionamento. Ao se criar um relacionamento, é necessário definir o número de ocorrências de instâncias das entidades que participam dele, por meio das cardinalidades mínima e máxima. A Figura 2 apresenta o relacionamento ‘Lotação’ entre as entidades ‘Pessoa’ e ‘Departamento’.

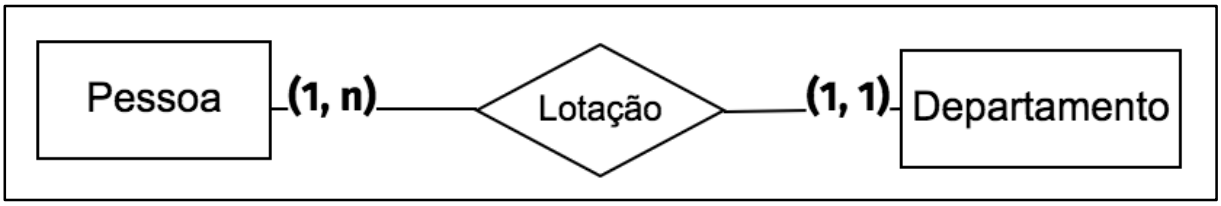


Figura 2 - Relacionamento lotação
Fonte: elaborado pelo autor.

2.1.1.1.3 *Atributo*

Atributos são características que descrevem uma entidade ou um relacionamento. Por exemplo, a Figura 3 demonstra a representação das características de uma entidade ‘Pessoa’ por seus atributos ‘CPF’, ‘Nome’ e ‘Endereço’. Sua representação é dada por um círculo conectado à entidade ou relacionamento.

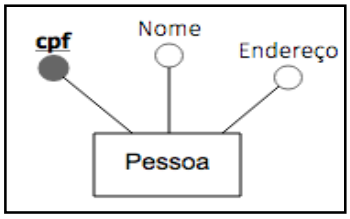


Figura 3 - Atributos da entidade ‘Pessoa’
Fonte: elaborado pelo autor.

Atributos podem ser de diferentes tipos, conforme segue:

- **Obrigatório/opcional:** um atributo **obrigatório** deve obrigatoriamente estar associado a uma ocorrência de uma entidade/relacionamento a qual ele pertence, ou seja, deve possuir valor diferente de *NULL* (cardinalidade

mínima igual a 1). Já um atributo **opcional** pode ou não possuir valor (cardinalidade mínima igual a 0).

- **Monovalorados/multivalorados:** atributos **monovalorados** possuem apenas um valor para uma entidade em particular (cardinalidade máxima igual a 1). Atributos **multivalorados** possuem 1 ou mais valores (cardinalidade máxima igual a ‘n’).

- **Simples/composto:** atributos **simples** são formado por apenas um valor, ou seja, não podem ser quebrados em partes menores. Já os atributos **compostos** podem ser formados (ou compostos) por partes menores.

2.1.1.1.4 *Especialização/generalização*

Especialização/generalização é uma técnica de herança em que as entidades mais especializadas herdam características ou relacionamentos de uma ou mais entidades genéricas. Dessa forma, evita-se a repetição dessas características ou desses relacionamentos na representação de várias entidades. O seu símbolo de representação é um triângulo isósceles.

Uma especialização pode ser classificada em total (t) ou parcial (p). Na especialização total, para cada ocorrência da entidade genérica existe sempre uma ocorrência em uma ou mais entidades especializadas. Já na especialização parcial, nem toda ocorrência da entidade genérica corresponde a alguma entidade especializada. As Figuras 4 e 5 representam especializações totais e parciais, respectivamente.

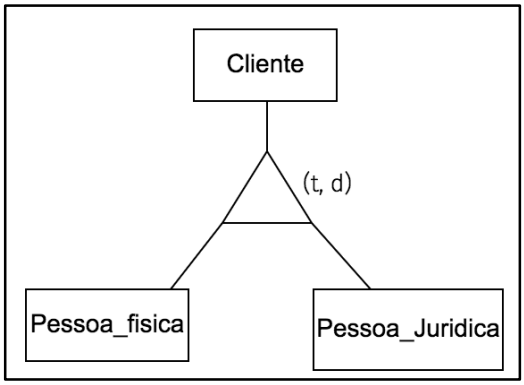


Figura 4 - Especialização total
Fonte: elaborado pelo autor.

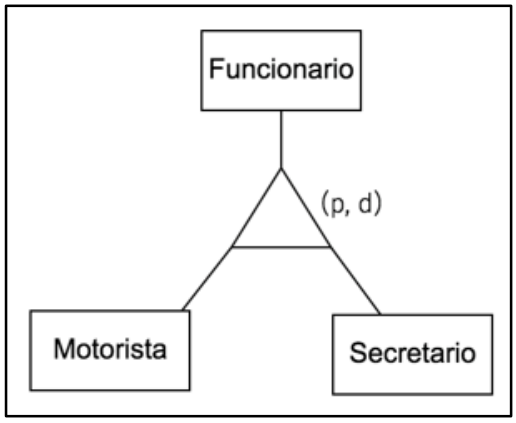


Figura 5 - Especialização parcial
Fonte: elaborado pelo autor.

Uma especialização ainda pode ser classificada em disjunta (d) ou compartilhada (c). A especialização disjunta determina que uma ocorrência de uma entidade genérica pode aparecer apenas em uma especialização. Já a especialização compartilhada permite que uma entidade genérica tenha ocorrências em mais de uma especialização. A Figura 6 mostra um exemplo de especialização compartilhada. Neste caso, a especialização também é parcial, indicando que uma ocorrência de 'Pessoa' não necessita obrigatoriamente de especialização como Professor ou Aluno.

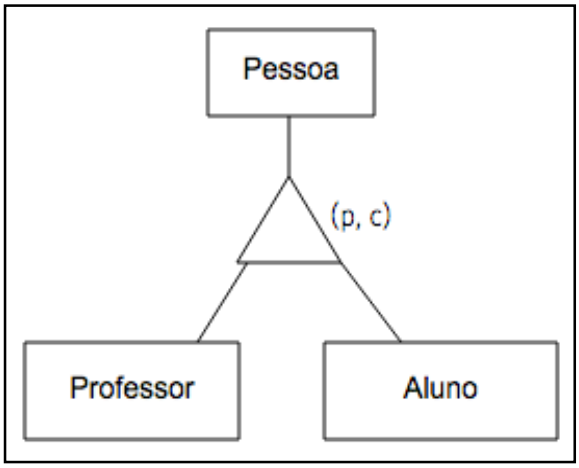


Figura 6 - Especialização parcial compartilhada
Fonte: elaborado pelo autor.

2.1.1.1.5 *Entidade associativa*

Uma entidade associativa abstrai uma associação de relacionamento com relacionamento, a qual não é permitida no modelo ER. Nesse caso, um relacionamento é promovido (se 'transforma') em uma entidade associativa para

resolver esse problema. Sua representação é feita por meio de um retângulo circundando o losango do relacionamento e as entidades conectadas a ele. A Figura 7 apresenta uma entidade associativa.

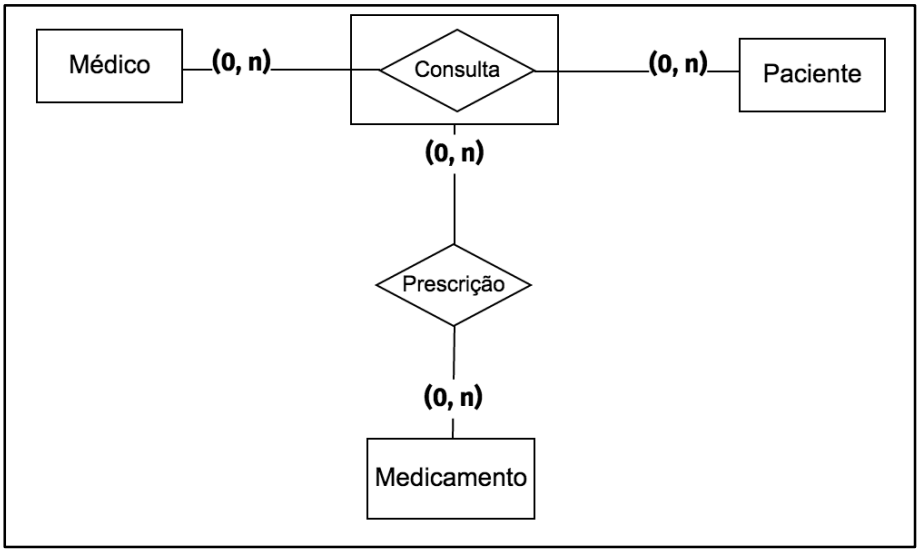


Figura 7 - Entidade associativa
Fonte: elaborado pelo autor.

Cabe salientar que esta seção apresentou sucintamente os principais conceitos do modelo ER. Existem várias outras características e peculiaridades que não foram abordadas aqui. Um aprofundamento do tema é apresentado no Capítulo 4, no qual cada um dos conceitos do modelo ER é descrito na ferramenta proposta neste trabalho.

2.1.2 Projeto lógico

Nesta etapa é gerada uma modelagem lógica, ou seja, uma descrição do BD no modelo de dados do SGBD alvo. Sendo assim, ela depende do tipo particular de SGBD (objeto, relacional ou hierárquico, por exemplo). Esta etapa é produto da conversão de uma modelagem conceitual, apesar de essa premissa nem sempre ser seguida, especialmente quando o projetista já tem domínio conceitual do projeto e decide iniciar o projeto do BD diretamente pelo projeto lógico. A Figura 8 mostra um exemplo de modelagem lógica, onde as tabelas Obra e Autor se conectam através da chave estrangeira (FK) idAutor, presente na entidade Obra.

Figura 8 - Modelagem lógica
Fonte: elaborado pelo autor.

A transformação ER para projeto lógico relacional segue um conjunto de regras que visam a obtenção de uma boa performance do banco de dados, a simplificação do desenvolvimento e a manutenção das aplicações. Essa transformação ocorre de acordo com os seguintes passos:

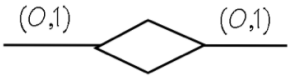
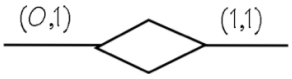
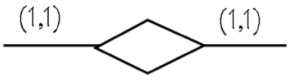
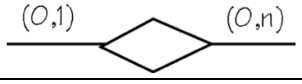
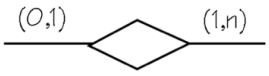
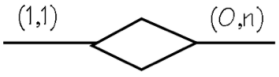
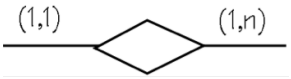
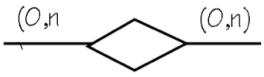
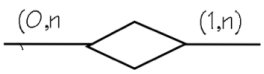
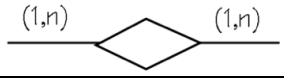
- Passo 1: Mapeamento de entidades e seus atributos: entidades identificadas na modelagem conceitual são convertidas em tabelas e seus atributos simples são convertidos em colunas de tabelas. Atributos multivalorados podem gerar uma nova tabela ou não. No caso de entidades fracas (cuja existência depende da existência de outra entidade), o atributo identificador da entidade forte torna-se parte da chave primária na tabela correspondente à entidade fraca.

- Passo 2: Mapeamento de especializações: no mapeamento de especializações, três alternativas podem ser adotadas. A primeira alternativa é a criação de uma tabela única para a entidade genérica e suas especializações. Também, se pode gerar uma tabela para a entidade genérica e uma tabela para cada entidade especializada, ou ainda, criar tabelas apenas para as entidades especializadas.

- Passo 3: Mapeamento de relacionamentos e seus atributos: nesta etapa, dependendo da cardinalidade atribuída ao relacionamento, algumas opções

de conversão podem ser adotadas. O Quadro 1 apresenta as opções de conversão de acordo com as cardinalidades.

Quadro 1 - Regras de implementação de relacionamentos

Tipo de Relacionamento	Regra de Conversão		
	Tabela Própria	**Adição de Coluna**	**Fusão de Tabelas**
Relacionamentos (1, 1)			
	TALVEZ	SIM	NÃO
	NÃO	TALVEZ	SIM
	NÃO	TALVEZ	SIM
Relacionamentos (1, n)			
	TALVEZ	SIM	NÃO
	TALVEZ	SIM	NÃO
	NÃO	SIM	NÃO
	NÃO	SIM	NÃO
Relacionamentos (n, n)			
	SIM	NÃO	NÃO
	SIM	NÃO	NÃO
	SIM	NÃO	NÃO

Legenda: SIM: alternativa preferida; TALVEZ: pode ser usada; NÃO: não usar.
Fonte: Houser (2009).

2.1.3 Projeto físico

O projeto físico descreve a forma como os dados são armazenados no SGBD. Nesta fase, as regras descritas nas etapas anteriores são convertidas em regras de integridade.

Como o projeto físico está diretamente ligado ao SGBD escolhido, detalhes de armazenamento interno, os índices para otimização de consultas e os caminhos de acesso às informações devem ser especificados aqui, uma vez que são particulares de cada SGBD e do dialeto SQL seguido por ele. Essas regras, bem como as tabelas geradas na etapa anterior, são convertidas em uma especificação SQL/DDL (*Structured Query Language/Data Description Language*) específica de um SGBD. O trecho de código apresentado na Figura 9 apresenta um exemplo de *script* de criação das tabelas referente à modelagem lógica.

```
CREATE TABLE Autor (
        nome VARCHAR(255),
        sexo VARCHAR(255),
        codigo INT NOT NULL,
        PRIMARY KEY (codigo),
)
CREATE TABLE Obra (
        ano VARCHAR (255),
        titulo VARCHAR (255),
        data_criacao INT,
        codigo INT NOT NULL,
        idAutor INT NOT NULL,
        PRIMARY KEY (codigo),
        FOREIGN KEY (idAutor) REFERENCES Autor (codigo)
)
```

Figura 9 - *Script* de criação de tabela
Fonte: elaborado pelo autor.

2.2 A FERRAMENTA brModelo E SUAS VERSÕES

2.2.1 *brModelo* – a primeira versão

A primeira versão da ferramenta *brModelo* foi desenvolvida em 2005, como resultado da monografia de especialização de Carlos Henrique Cândido, intitulada

Aprendizagem em banco de dados: implementação de ferramenta de modelagem E.R.[8].

A proposta era construir uma ferramenta voltada para o ensino e para a aprendizagem de técnicas de modelagem de BDs relacionais. Para essa finalidade, foram adotados os conceitos apresentados por Heuser (2009), inclusive a representação gráfica proposta pelo autor.

Seguindo essas premissas, a primeira versão da *brModelo* tinha como principais funcionalidades: a criação de modelos conceituais baseados no modelo ER; o mapeamento semiautomático de uma modelagem ER para uma modelagem lógica relacional; e a posterior geração de um esquema físico descrito na linguagem SQL e independente de dialeto SQL de SGBDs relacionais.

A aplicação destacou-se principalmente no meio acadêmico, por ser gratuita, considerar todas as etapas de um projeto de BD e suportar uma etapa de modelagem conceitual que considera a grande maioria dos construtores do modelo ER, o que lhe conferiu um diferencial, quando comparada às aplicações disponíveis na época. A Figura 10 apresenta a tela de modelagem conceitual da ferramenta *brModelo*.

[8] CÂNDIDO, Carlos Henrique**. Aprendizagem em banco de dados:** implementação de ferramenta de modelagem E.R. 2005. Monografia (Especialização em Banco de Dados)- Pós-Graduação em Banco de Dados, Universidade Federal de Santa Catarina - UFSC, Universidade de Várzea Grande - UNIVAG, Várzea Grande-MT, 2005.

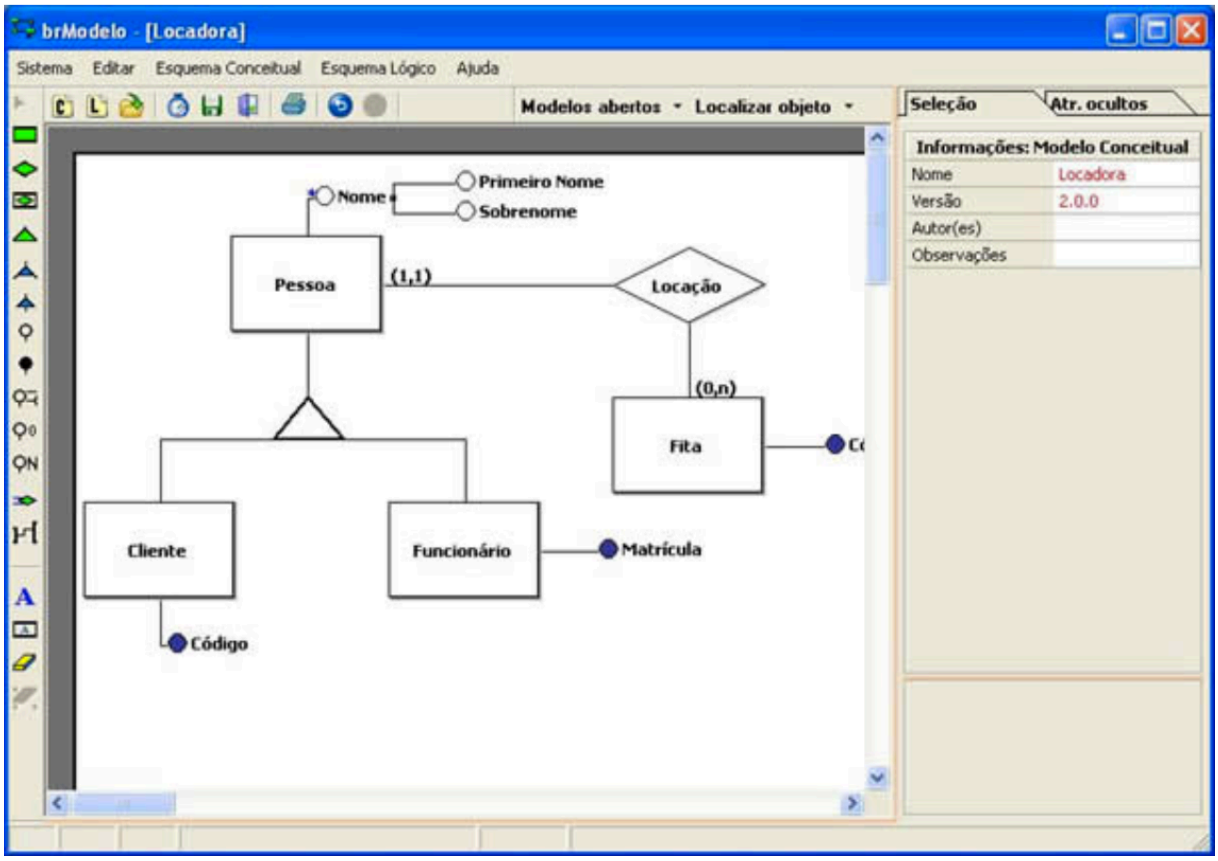


Figura 10 - *brModelo*: tela modelagem conceitual
Fonte: elaborado pelo autor.

Apesar de ser bastante difundida no meio acadêmico, a ferramenta *brModelo* apresenta alguns problemas, principalmente relacionados à portabilidade. Uma vez que a implementação foi realizada por meio da linguagem *Object Pascal/Delphi*, a execução da ferramenta é possível somente em sistema operacional *Windows*.

A exigência de que professores e alunos utilizem um sistema operacional proprietário vai contra um dos principais objetivos do projeto, que era criar um programa gratuito para ensino no meio acadêmico.

2.2.2 brModeloNext

Com o objetivo principal de solucionar o problema da portabilidade encontrado na primeira versão da *brModelo*, ou seja, torná-la multiplataforma (independente de sistema operacional) está em desenvolvimento, desde 2011, a ferramenta *brModeloNext (MENNA; RAMOS; MELLO, 2011)*, tendo sido tema de trabalhos de conclusão de curso no INE/UFSC . Além disso, a aplicação traz melhorias na interface gráfica com base em avaliações heurísticas e testes de usabilidade.

Para a execução independentemente de sistema operacional, foi proposta a utilização da linguagem de programação *Java*. Para os componentes gráficos de representação dos modelos, foi utilizada a biblioteca de código aberto *JGraph* e o desenvolvimento seguiu o paradigma de programação orientado a objetos com padrão de projeto *Model-View-Controller*, que permite uma melhor separação de código. Entre as melhorias gráficas adotadas no projeto da ferramenta *brModeloNext*, destacam-se:

- Modelagens exibidas em formato de janelas: possibilita a visualização de diversas modelagens simultaneamente, facilitando comparações entre as elas (Figura 11);

- Edição *in-place*: permite a edição dos componentes gráficos por meio de uma janela aberta em cima do componente. Essa mudança foi justificada pelo ganho de tempo para o usuário, pois a criação e edição dos atributos de objetos como entidades e tabelas é feita mais rapidamente, bastando um duplo clique sobre o objeto;
- *Tooltip*: facilita a visualização das propriedades de um objeto e dos ícones da ferramenta apenas fixando o ponteiro do *mouse* sobre esses elementos por um curto espaço de tempo.

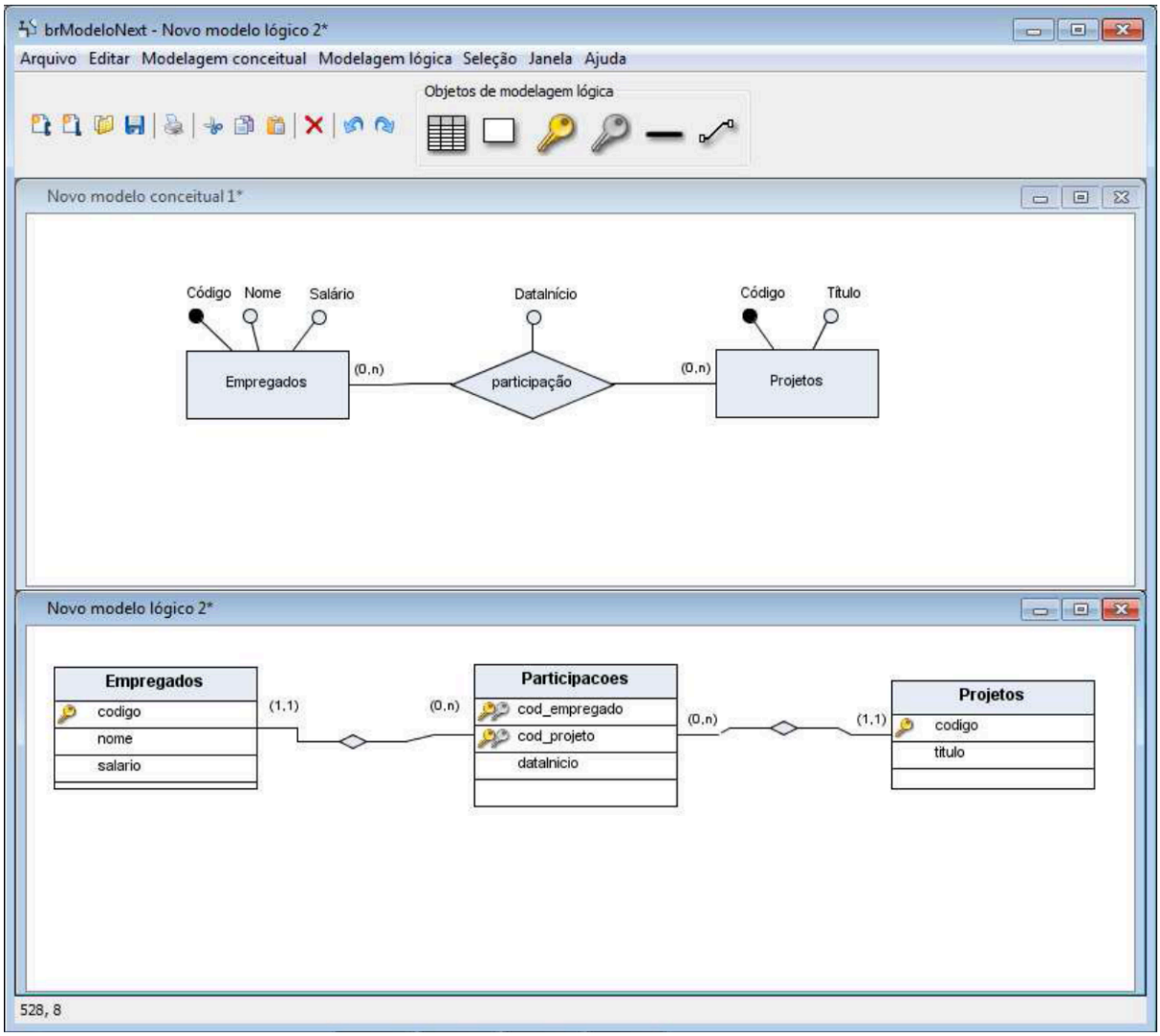


Figura 11 - Ferramenta *brModeloNext*: apresentação de janelas
Fonte: elaborado pelo autor.

Apesar de a ferramenta *brModeloNext* ter resolvido o problema da portabilidade e apresentar uma grande evolução na questão de usabilidade em relação à *brModelo*, ela não fez tanto sucesso quanto sua antecessora. Acredita-se que isso se deve à demora em se publicar uma versão estável para download. Neste momento (2016), a ferramenta *brModeloNext* ainda não está disponível para uso e não pode ser encontrada para download em nenhuma fonte da internet.

Este capítulo apresentou uma revisão dos conceitos de projeto de banco de dados e as versões da brModelo. O capítulo seguinte apresenta e compara ferramentas *web* destinadas a projeto de BDs relacionais.

3 TRABALHOS RELACIONADOS

Este capítulo apresenta e compara ferramentas *web* destinadas a projeto de BDs relacionais. Apenas esses tipos de ferramentas são analisadas devido a diferente plataforma e recursos disponíveis entre aplicativos *desktop* e *web*.

Além disso, comparar ferramentas com propostas tecnológicas opostas também não traz nenhuma contribuição para a análise. Assim sendo, as quatro ferramentas *web* apresentadas nessa sessão são a *WWW SQL DESIGNER*, *ERDPLUS*, *SmartDraw Cloud* e *Lucidchart*. Estas ferramentas foram escolhidas pois são aquelas que estão a mais tempo disponíveis para uso, apresentam um maior número de funcionalidades, maior quantidade de usuários, e facilidade de uso.

3.1 WWW SQL *DESIGNER*

A *WWW SQL Designer* é uma ferramenta de modelagem com foco no projeto lógico. Apesar de apresentar muitas limitações, ela tem a vantagem de ser de código aberto. Isso possibilita que a comunidade de desenvolvedores ajude em sua evolução.

Além dos recursos básicos, como criação de tabelas, relacionamentos e tipos de dados para colunas, seus pontos fortes são a edição simplificada de tabelas (como pode ser visto na Figura 12) e a forma visual de relacionamento entre as tabelas e as chaves estrangeiras, além da colaboração em tempo real entre projetistas.

A ferramenta destaca-se ainda pela geração automática de *script* SQL e por permitir a extração e a importação de diagramas, além da funcionalidade de engenharia reversa. Apesar disso, ela ainda tem muito a evoluir, principalmente nos quesitos usabilidade e documentação. É bastante difícil entender como iniciar um projeto e como usar as suas funcionalidades de modo geral. Outro ponto negativo é o fato de não permitir o projeto conceitual, o que limita o seu uso para fins acadêmicos, por exemplo.

Para usar a ferramenta é preciso baixar o pacote no repositório *GitHub* do desenvolvedor[9] e descompactar em um servidor (*web* ou local), ou então usar diretamente a demonstração *on-line*[10].

[9] Disponível em: <https://github.com/ondras/wwwsqldesigner>. Acesso em: 15 set. 2016.
[10] Disponível em: <http://ondras.zarovi.cz/sql/demo/?keyword=default>. Acesso em: 15 set. 2016.

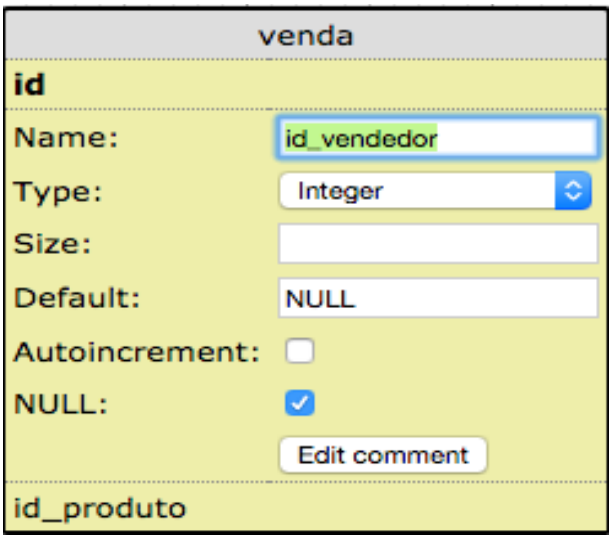


Figura 12 - *WWW SQL Designer*: janela de edição rápida
Fonte: elaborado pelo autor.

A Figura 13 ilustra uma modelagem lógica realizada na ferramenta. É possível observar que ela não é intuitiva, pois não permite, por exemplo, que o usuário continue o processo de conversão para modelo físico de forma fácil, sem que o usuário tenha que dar muitos cliques em menus de opção apresentados à direita.

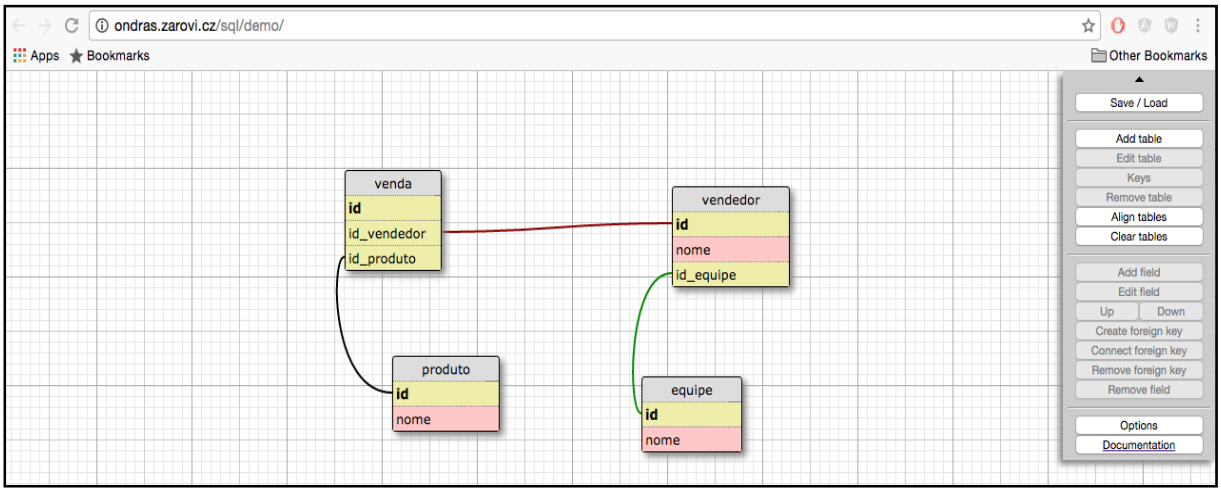


Figura 13 - Ferramenta *WWW SQL Designer*
Fonte: elaborado pelo autor.

3.2 ERDPLUS

A *ERDPLUS* é a ferramenta gratuita mais completa entre as analisadas. É a única que cobre as três etapas de modelagem do projeto de BD. Para o projeto conceitual, ela utiliza o modelo ER com a notação clássica apresentada por Peter Chen (CHEN, 1976), suportando o desenho de entidade, entidade fraca, entidade

associativa, os vários tipos de atributo (regular, único, composto, derivado e opcional) e relacionamentos.

Os pontos negativos do módulo para projeto conceitual são o fato de não permitir generalização/especialização e pela notação não intuitiva para definição das cardinalidades. Para a modelagem lógica, ela suporta a criação de tabelas, a definição de tipos de atributos e restrições de integridade. Esse módulo ainda se encontra em sua fase *beta* e apresenta alguns *bugs*. Os problemas se encontram principalmente na geração das chaves estrangeiras e na apresentação gráfica. O módulo também não é tão simples de ser usado quanto o que trata da modelagem conceitual, o que dificulta o uso.

Além desses recursos, a ferramenta ainda possui as funcionalidades de exportação para arquivo em formato PNG, conversão automática do projeto conceitual para projeto lógico, armazenamento das modelagens criadas no servidor remoto, criação de conta na ferramenta com credenciais do *Facebook* e *Google*, bem como geração de *script* SQL a partir da modelagem lógica.

Em geral, a ferramenta é bem completa e intuitiva. No entanto, ainda apresenta muitos *bugs* no processo de modelagem, forçando o usuário a ter que recomeçar o processo inúmeras vezes. Além disso, pontos que podem ser melhorados são a apresentação de *feedbacks*, tratamento de erros e disponibilização de mais tutoriais. A conversão totalmente automática também é um outro ponto que pode ser revisto, pois impede o projetista de decidir, dentre as diversas opções de conversão a partir da modelagem conceitual, qual é a conversão mais adequada para um projeto de BD em questão.

As Figuras 14 e 15 ilustram um exemplo de modelagem conceitual e sua respectiva conversão em esquema lógico.

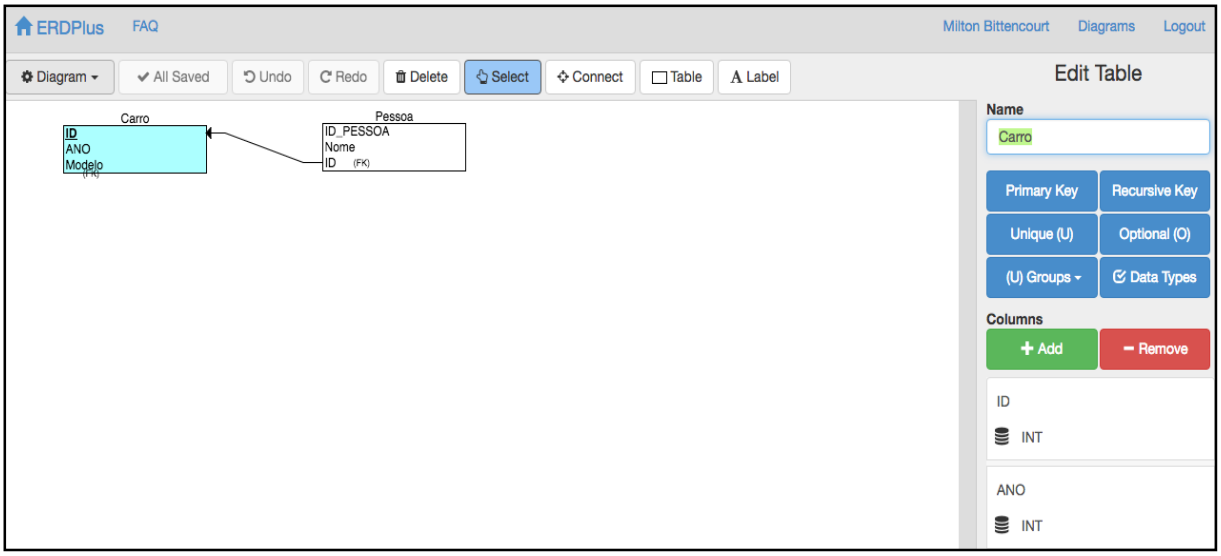


Figura 14 - Ferramenta *ERDPLUS*: modelagem lógica
Fonte: elaborado pelo autor.

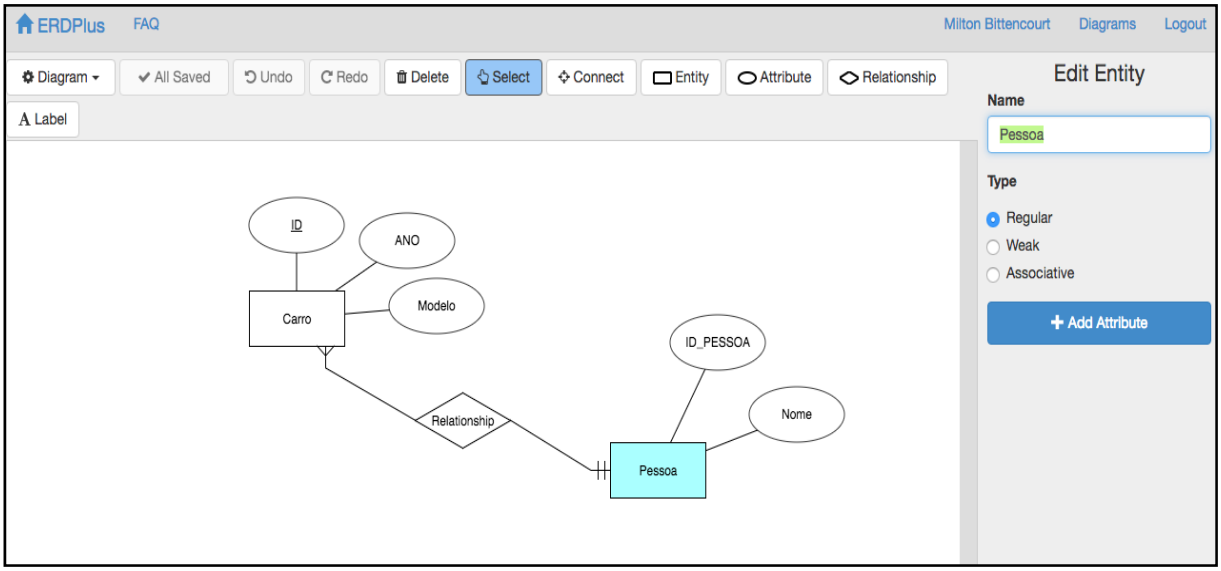


Figura 15 - Ferramenta *ERDPLUS*: modelagem conceitual
Fonte: elaborado pelo autor.

3.3 SMARTDRAW CLOUD

A *SmartDraw Cloud* é uma ferramenta proprietária voltada ao mercado corporativo, contemplando a criação e a edição de diversos tipos de diagramas, além dos diagramas ER.

A versão *desktop* desta ferramenta é mais conhecida, por existir há mais tempo. Ela foi concebida apenas para a plataforma *Windows*, mas possui completa integração com a sua versão *Web*.

A ferramenta possui, segundo seu *site* oficial, mais de um milhão de clientes. Ela cobre apenas o projeto conceitual. No entanto, destaca-se por ser muito completa nesse aspecto. Tem como base o modelo ER, com a notação proposta por Peter Chen. Todavia, é bastante flexível, permitindo que o usuário decida qual notação para relacionamento utilizar e até que ele faça customizações.

A ferramenta destaca-se bastante pela ótima usabilidade. Há uma facilidade em adicionar formas e textos e navegar utilizando apenas o teclado. Os textos sempre obedecem aos limites das formas, que são redimensionados automaticamente. Textos podem ser adicionados ao longo de uma linha, nas formas ou em qualquer área da modelagem em formato de observação. Ao selecionar qualquer componente, as dimensões são apresentadas por meio de réguas com a escala dos gráficos. Ainda, a ferramenta redimensiona diretamente as formas e linhas com exatidão, por meio da edição dos seus atributos. Além disso, ela permite o compartilhamento de documentos por meio de um *link*, que pode ser distribuído a qualquer pessoa, pode ser acessado de qualquer navegador em qualquer dispositivo e oferece suporte ao *Google Drive*™, *Dropbox*® e *OneDrive*™.

Apesar de cobrir apenas a etapa de modelagem conceitual, a *SmartDraw Cloud* é uma boa ferramenta, muito simples de usar e com uma infinidade de tutoriais, inclusive materiais educativos relacionados a fundamentos e modelagem de BD. No entanto, para a utilização é necessário comprar uma licença anual, ao custo de 155 dólares americanos (valor consultado em setembro/2016).

Um ponto negativo é que, por permitir inúmeras customizações, a ferramenta não possui um sistema de verificação, permitindo que o usuário crie modelagens ilegais no modelo ER.

Outro ponto negativo é que a ferramenta e toda a documentação estão disponíveis apenas no idioma Inglês. A Figura 16 apresenta a área de edição de modelagem e uma modelagem construída para demonstração.

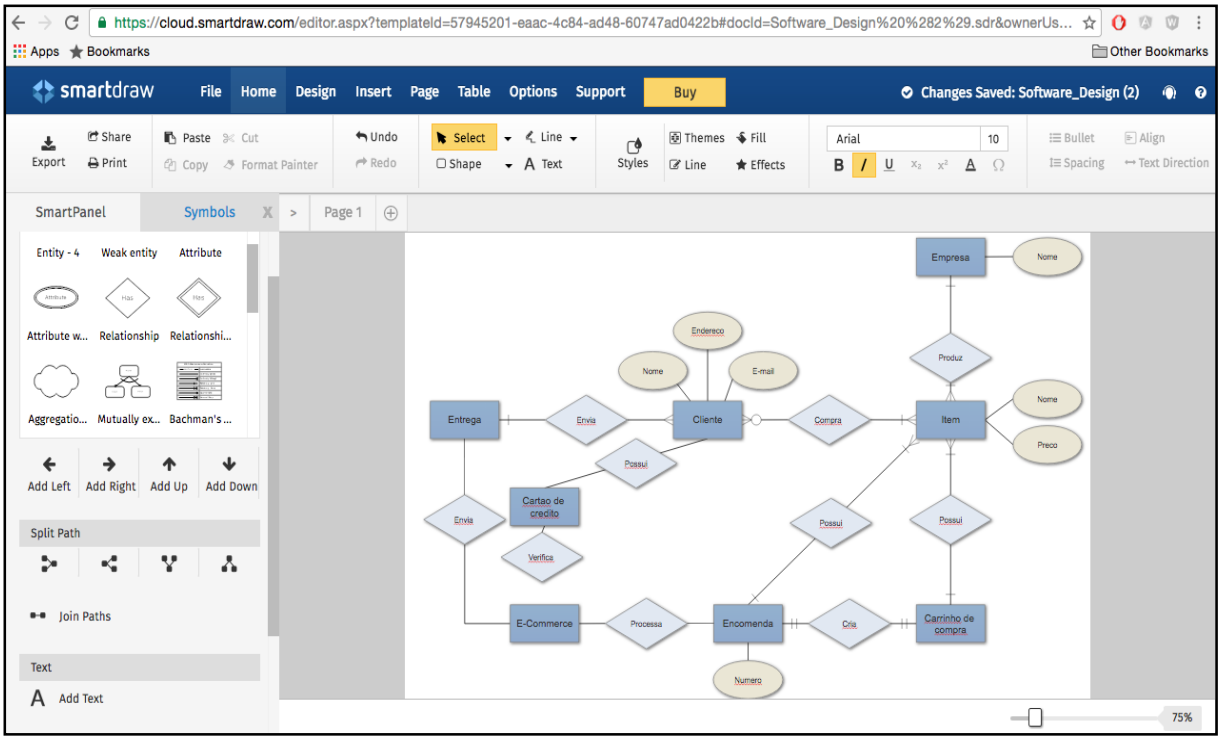


Figura 16 - Ferramenta *SmartDraw Cloud*: modelagem conceitual
Fonte: elaborado pelo autor.

3.4 LUCIDCHART

A *Lucidchart* é uma das ferramentas web de diagramação mais utilizadas no mundo. Segundo o *site* expositivo do produto, são mais de seis milhões de usuários ativos. É importante ressaltar que a ferramenta não foca apenas na modelagem de BD, mas em praticamente todos os tipos de diagramação.

Quanto à modelagem de BD, ela possibilita a criação do projeto conceitual e do projeto lógico. A ferramenta destaca-se por ser muito simples e intuitiva, e possuir muitos tutoriais, o que facilita a criação de modelagens. Destaca-se, também, o módulo colaborativo, que permite que dois ou mais usuários editem o mesmo arquivo simultaneamente, além de um vasto acervo de modelagens de exemplo.

Outro ponto positivo da ferramenta é a compatibilidade com outros programas, incluindo o *Google Apps* e o *Microsoft Visio*.

A *Lucidchart* utiliza a notação de Chen na modelagem conceitual e uma notação própria na modelagem lógica. Os pontos negativos são a falta de possibilidade de conversão entre os modelos e a possibilidade de se criar modelagens inválidas.

A ferramenta é inicialmente gratuita, embora com restrições, como, por exemplo, o limite de no máximo cinco modelagens, limitação de número de componentes e não-exportação de modelagens, nem mesmo no formato de imagem.

Para se ter acesso a todas as funcionalidades de edição e à criação ilimitada de modelagens e componentes é necessário contratar uma assinatura mensal, que varia entre 5 e 20 dólares americanos. A Figura 17 ilustra uma modelagem conceitual gerada na ferramenta.

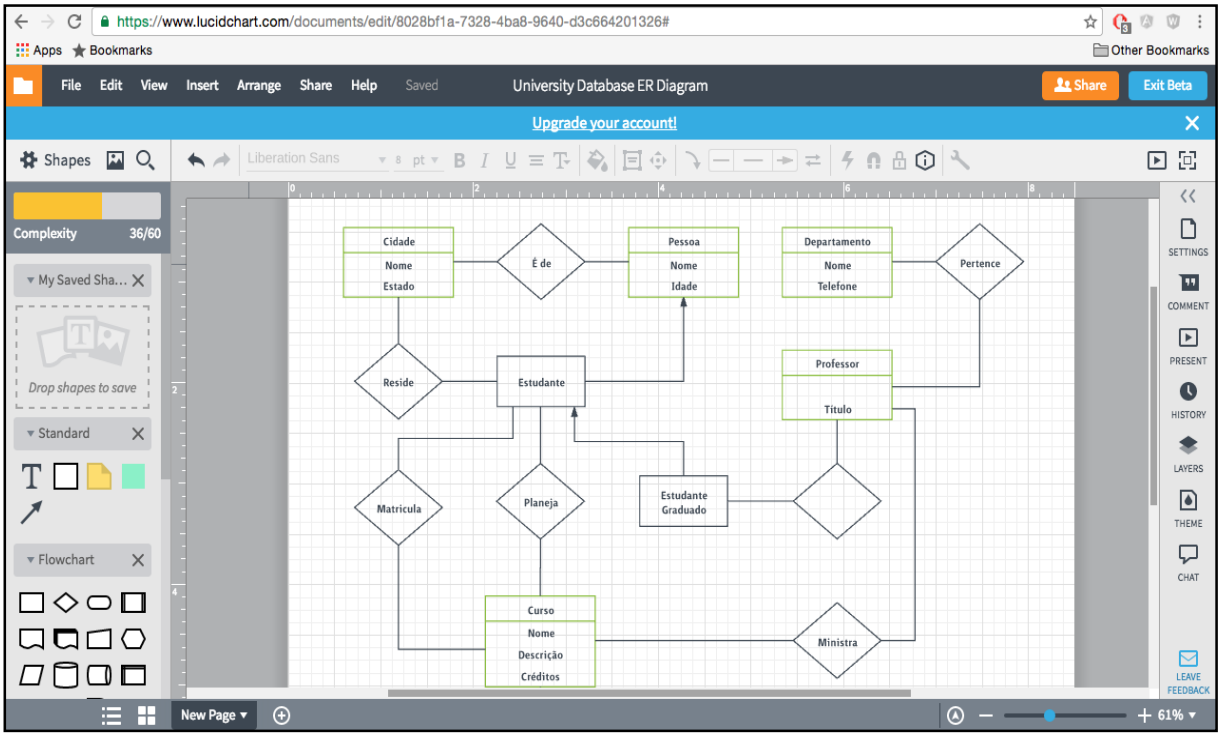


Figura 17 - Ferramenta *Lucidchart*: modelagem conceitual
Fonte: elaborado pelo autor.

No Quadro 2, uma comparação entre a ferramenta proposta e as ferramentas existentes é apresentada, levando em consideração algumas funcionalidades relacionadas ao projeto de BD e outros itens relevantes no contexto de ensino, como materiais tutoriais e gratuidade. Conforme mostrado no Quadro 2, a ferramenta *brModeloWeb*, proposta neste trabalho de conclusão de curso, é a única que contempla todos os itens comparados. A ERDPLUS é outra ferramenta que se destaca, no entanto, a análise realizada na seção 3.2 mostra que ela cumpre apenas funcionalidades básicas, precisando ainda evoluir nas modelagens propostas.

Quadro 2 - Comparação entre ferramentas de projeto de BD para a web

Funcionalidade/ Ferramenta	***SQL DESIGNER***	***ERD PLUS***	***SmartDraw Cloud***	***Lucidchart***	***brModeloWeb***
1. Modelagem conceitual	NÃO	SIM	SIM	SIM	SIM
2. Modelagem lógica	SIM	SIM	NÃO	SIM	SIM
3. Conversão conceitual-lógico	NÃO	SIM	NÃO	NÃO	SIM
4. Conversão lógico-físico	SIM	SIM	NÃO	NÃO	SIM
5. Persistência de modelagens	NÃO	SIM	SIM	SIM	SIM
6. Gratuito	SIM	SIM	NÃO	NÃO	SIM
7. Tutoriais	NÃO	NÃO	SIM	SIM	SIM

Fonte: elaborado pelo autor.

Este capítulo apresentou uma analise das ferramentas web similares à ferramenta proposta neste trabalho. Após o estudo individual das ferramentas, foi realizada uma comparação entre as mesmas e a *brModeloWeb,* que justifica sua implementação. O próximo capítulo apresenta as tecnologias utilizadas para o desenvolvimento da *brModeloWeb* e suas funcionalidades.

4 PROJETO DA FERRAMENTA PROPOSTA

Esta seção apresenta tecnologias e ferramentas utilizadas no desenvolvimento da *brModeloWeb*, assim como, a motivação e a justificativa das escolhas tecnológicas necessárias para o entendimento da proposta de arquitetura da ferramenta.

4.1 ANÁLISE DE REQUISITOS

Os requisitos representam a descrição das necessidades dos usuários perante um *software*. O objetivo desta fase é identificar e documentar os requisitos. É necessário que a documentação dos requisitos seja feita de forma clara e que comunique efetivamente a informação que o usuário busca apresentar (LARMAN, 2007).

Os requisitos de um *software* podem ser classificados em Requisitos Funcionais (RF) ou Requisitos Não-Funcionais (RNF). Os Requisitos Funcionais são os responsáveis por definir funções e comportamentos do *software*; e os Requisitos Não-Funcionais são os responsáveis por especificar as qualidades do *software*, isto é, a confiabilidade, a segurança e o desempenho.

Para o levantamento de RF foram levadas em consideração as funcionalidades das versões anteriores da ferramenta *brModelo*. Os RNF foram definidos a partir de estudos de tecnologias *web* modernas. Os RF levantados são os seguintes:

- RF 01: Fazer *login*.
- RF 02: Criar conta.
- RF 03: Criar, editar, salvar e excluir modelagens.
- RF 04: Converter projeto conceitual em lógico.
- RF 05: Converter projeto lógico em físico.
- RF 06: Compartilhar modelagens.
- RF 07: Dimensionar área de trabalho.
- RF 08: Desfazer ações.
- RF 09: Imprimir modelagem.

Os RNF levantados são os seguintes:

- RNF 01: Utilização apenas da linguagem *JavaScript.*
- RNF 02: Arquitetura cliente-servidor.
- RNF 03: Utilização do *framework AngularJS*.
- RNF 04: Suporte aos navegadores *Web Chrome 4*+, *Firefox 3*, *Opera*, *Edge* e *Safari*.
- RNF 05: Utilização do *framework Node.JS*.

4.2 TECNOLOGIAS E FERRAMENTAS

As tecnologias e ferramentas utilizadas no desenvolvimento da ferramenta *brModeloWeb* são descritas a seguir.

4.2.1 JavaScript

JavaScript é uma linguagem de programação originalmente desenvolvida para a execução de código em páginas *web*. Desde então, ela foi adotada por praticamente todos os grandes navegadores do mercado, como *Google Chrome*, *Firefox*, *Safari* e *Opera*, tornando-a umas das linguagens mais populares do mundo (CROCKFORD, 2008). É uma linguagem interpretada, com tipagem fraca e dinâmica, com suporte à programação funcional e à programação orientada a objetos.

Inicialmente utilizada apenas como *script* para programar o comportamento de páginas e manipular os seus elementos, a linguagem *JavaScript* evoluiu a tal ponto que é utilizada como linguagem de consulta em BD, como o *MongoDB* e o *CouchDB*; em *frameworks* de manipulação do *DOM* como o *AngularJs*, e em plataformas de desenvolvimento *server-side* como o *Node.js.*

Devido à flexibilidade, eficiência e robustez, a *JavaScript* foi escolhida como linguagem de implementação deste trabalho, tanto na parte no servidor, onde há o processamento de dados e interação com o BD, quanto na parte cliente, onde existe a interação do usuário com páginas HTML, gerando o comportamento da aplicação. Como toda a ferramenta foi desenvolvida com *JavaScript*, diversas bibliotecas foram necessárias para facilitar o processo.

4.2.2 AngularJS

AngularJS é um *framework JavaScript* criado para aumentar a produtividade de desenvolvimento e organização de código no lado do cliente. O *framework* propõe o controle do DOM (*Document Object Model*) de forma declarativa, a extensão do vocabulário HTML, a criação de componentes para o reuso de código e uma forma mais organizada de manter o código *JavaScript*.

As principais virtudes da biblioteca são leveza e capacidade de propiciar organização e estrutura para aplicações de grande porte. Por isso, ela foi usada para organizar a parte do cliente no padrão MVC (*Model-View-Controller*), aumentando a velocidade de desenvolvimento, organização e manutenção.

4.2.3 JointJS

JointJS é uma biblioteca *JavaScript* de diagramação, de código aberto, que possibilita a criação de diversos elementos gráficos estáticos ou interativos renderizados em SVG (*Scalable Vector Graphics*). É possível utilizar *JointJS* para criar qualquer diagrama, ou ainda editores de diagramas interativos. A *JointJS* facilita a criação de aplicações visuais de diversos tipos, e a natureza orientada a eventos, somada à arquitetura MVC, facilita a ligação com qualquer ambiente de servidor. Segundo *Client.IO*[11], entre as principais funcionalidades e recursos estão:

- Renderização de elementos gráficos (retângulo, círculo, elipse, texto, imagem e *links*);
- Serialização e desserialização de elementos em formato JSON (*JavaScript Object Notation*);
- Conexão entre elementos por *links* via portas pré-fixadas;
- Eventos de interação com elementos (*click*, adição, remoção, mudança de posição).

Apesar de a biblioteca *JointJS* ter como base o *BackboneJS*, um *framework JavaScript* concorrente ao *AngularJS* utilizado como base da aplicação, o ótimo

[11] Disponível em: <https://github.com/clientIO/joint>. Acesso em: 15 set. 2016.

desempenho na manipulação de gráficos vetoriais foi decisivo para a escolha dessa biblioteca.

4.2.4 Node.js

Node.js é um *framework* orientado a eventos que possibilita a execução de *JavaScript* do lado do servidor. Ele foi desenvolvido utilizando o compilador de interpretação do navegador *Google Chrome*, cujo propósito é acelerar o desempenho de aplicações compilando o código *JavaScript* para o formato nativo de máquina antes de executá-lo.

A principal diferença entre *Node.js* e outros *frameworks web* é forma como as requisições são tratadas. Enquanto *frameworks* tradicionais utilizam *threads* em paralelo, fazendo com que a alocação de recursos possa ficar bloqueada esperando um evento concorrente finalizar, o *Node.js* propõe um conceito de fila de eventos, onde requisições são tratadas de maneira assíncrona, portanto, não bloqueante. O *framework* conta com milhares[12] de extensões disponíveis no repositório NPM[13], desenvolvidas pela comunidade de *software*, que aceleram o processo de desenvolvimento, uma vez que é possível reutilizar esses componentes.

4.2.5 MongoDB

MongoDB é um BD orientado a documentos, projetado para aumentar a produtividade do desenvolvedor e facilitar a escalabilidade de aplicações. Sua arquitetura consiste em um conjunto de bases de dados, sendo que cada base possui múltiplas coleções. Essas coleções podem possuir esquemas dinâmicos, ou seja, cada coleção pode conter diferentes tipos de objetos. Os objetos (também chamados de documentos) são representados em estruturas JSON chave-valor. Apesar da ferramenta *brModeloWeb* tratar especificamente de BDs relacionais, o *MongoDB* foi escolhido como BD de armazenamento por possibilitar sua manipulação direta por meio da linguagem *JavaScript*.

[12] Em 04.10.2016 eram 331.479, com uma taxa de 398 novos plugins por dias. Informação disponível em: <http://www.modulecounts.com/>. Acesso em: 25 set. 2016.
[13] Disponível em: <https://www.npmjs.com>. Acesso em: 25 set. 2016.

4.2.6 REST

REST (*Representational State Transfer*) é um estilo arquitetural de *software* voltado para sistemas de hipermídia distribuídos, tais como a *World Wide Web*. Esse estilo arquitetural, proposto por Roy Fielding (2000), tem sido largamente usado para guiar o *design* e o desenvolvimento de diversas aplicações da *web* moderna. O estilo arquitetural REST enfatiza a escalabilidade das interações entre componentes, a generalidade das interfaces, o desenvolvimento independente de componentes e a utilização de componentes intermediários para reduzir a latência das interações, reforçar a segurança e encapsular sistemas legados (FIELDING, 2000).

4.2.7 HTML

HTML (*HyperText Markup Language*) é uma linguagem de marcação utilizada para produzir páginas na *web* independentemente de plataforma, ou seja, o mesmo código HTML é interpretado da mesma forma por diferentes *browsers*. Atualmente, a HTML se encontra em sua 5ª versão, com facilidades para manipulação de dados via *JavaScript* e CSS.

4.2.8 CSS

CSS (*Cascading Style Sheets*) é uma linguagem de folha de estilo, utilizada para descrever a aparência de linguagens de marcação como HTML e XML (*eXtensible Markup Language*), incluindo cores, tamanhos, fontes e posicionamento dos elementos na tela.

4.3 ARQUITETURA

A arquitetura utilizada para estruturar a ferramenta *brModeloNext* se baseia na mescla entre o modelo Cliente-Servidor e o padrão de projetos MVC. Segundo Tanenbaum (2003), no modelo Cliente-Servidor a comunicação toma a forma do processo cliente, enviando uma mensagem pela rede ao processo servidor. Quando o servidor recebe a solicitação, ele executa o trabalho solicitado ou procura pelos

dados solicitados e envia de volta uma resposta. Ainda, segundo Tanenbaum (2003), esse modelo é amplamente usado no contexto de aplicações web.

O MVC, padrão de projetos mais utilizado em arquitetura em desenvolvimento de *software*, determina que a aplicação esteja dividida em três camadas com especialidades distintas. O *Model* é responsável por representar os dados da aplicação, realizando o acesso ao sistema de armazenamento e lógica sobre eles. O *View* trata de como o modelo é apresentado ao usuário. Por fim, o *Controller* recebe todas as solicitações da aplicação e é responsável por executar sobre o modelo e a visão (FOWLER, 2003). O MVC foi utilizado tanto na parte cliente, quanto servidor.

A Figura 18 apresenta uma visão geral da arquitetura da aplicação. Toda a comunicação entre os componentes se dá por meio do protocolo REST. As seções 3.3.1 e 3.3.2 descrevem as responsabilidades dos componentes apresentados nesta visão geral de arquitetura.

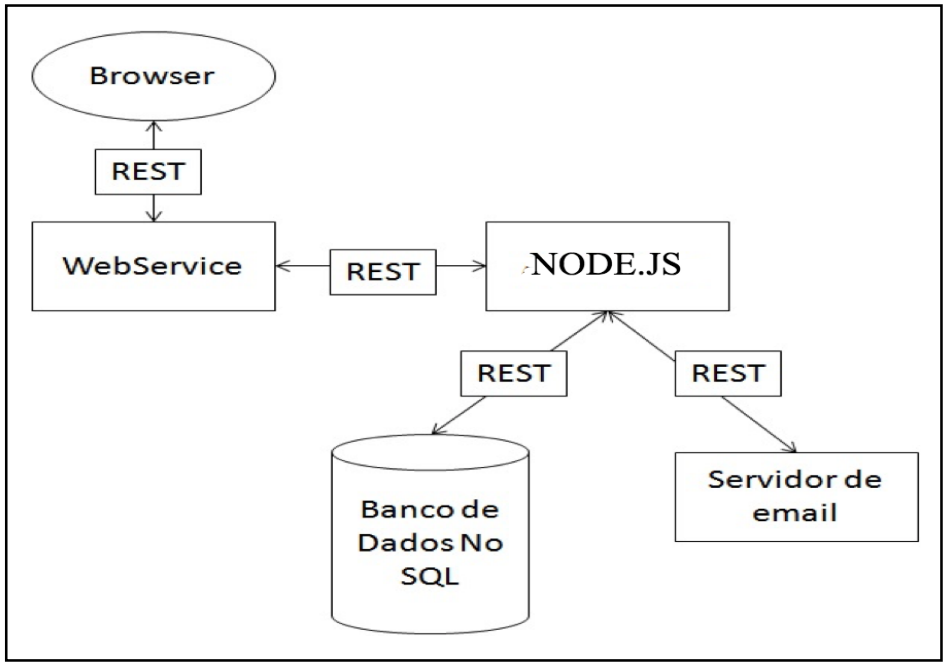


Figura 18 - Arquitetura da ferramenta *brModeloWeb*
Fonte: elaborado pelo autor.

4.3.1 Servidor

O servidor foi desenvolvido em *NODE.js* no esquema de *Web Service REST*. Ele pode ser facilmente instalado localmente em qualquer sistema operacional ou em

serviços de hospedagem como *Heroku*[14], *Google Appengine*[15], *Red Hat OpenShift*[16], entre outros. A parte de servidor da ferramenta *brModeloWeb* é responsável pela persistência de dados, envio de *e-mail* e todo o controle da lógica da aplicação.

4.3.2 Cliente

A parte cliente foi inteiramente desenvolvida em HTML e *JavaScript*. Para renderização do conteúdo foram utilizados o *framework AngularJS* e a biblioteca de gráficos SVG *JointJS*. Em *JointJS*, diagramas consistem em um conjunto de elementos conectados por *links* e são representados pelo modelo (*Model*) *joint.dia.Graph*. Este modelo agrupa células, as quais podem ser elementos ou *links*. Nesse esquema, as visões são reativas, ou seja, reagem por meio da captura de eventos das modificações no modelo. Assim, garante-se que o modelo sempre reflete o estado atual do dado. A Figura 19 exibe o fluxo de informação entre os componentes da biblioteca *JointJs*.

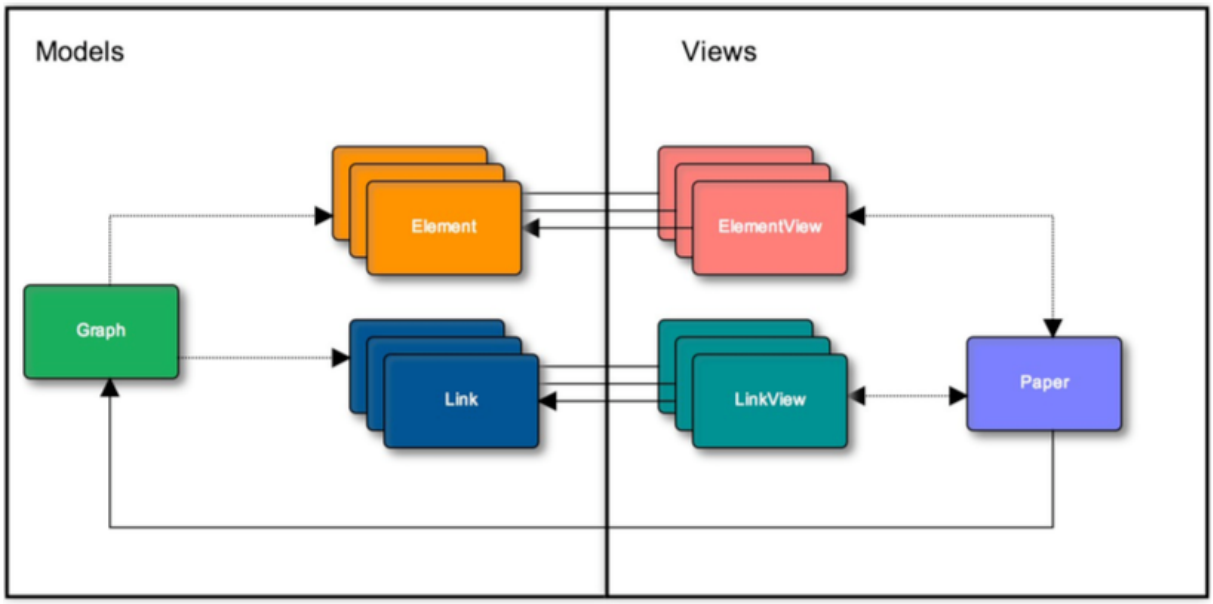


Figura 19 - Fluxo de informação na parte cliente
Fonte: elaborado pelo autor.

[14] Disponível em: <https://devcenter.heroku.com/articles/getting-started-with-nodejs#introduction>. Acesso em: 15 set. 2016.
[15] Disponível em: <https://cloud.google.com/nodejs/>. Acesso em: 15 set. 2016.
[16] Disponível em: <https://developers.openshift.com/languages/nodejs/getting-started.html>. Acesso em: 15 set. 2016.

4.4 FUNCIONALIDADES

4.4.1 Criação de conta

Para acessar a *brModeloWeb* é necessário criar uma conta. A Figura 20 exibe o formulário de criação de conta. Os únicos requisitos para esta etapa são que o *e-mail* seja válido e que esse *e-mail* não esteja previamente cadastrado no sistema. Caso algum desses requisitos não sejam cumpridos, uma mensagem de erro é apresentada. Após o cadastro, o usuário recebe um *e-mail* com um *link* de ativação para garantir que o endereço eletrônico fornecido seja de sua propriedade.

Figura 20 - Formulário de criação de conta
Fonte: elaborado pelo autor.

4.4.2 Login

Após a criação de conta, outro requisito que deve ser cumprido para que o usuário tenha acesso à ferramenta é o *login*. Assim como na etapa de cadastro, caso o usuário não forneça suas credenciais, o *e-mail* seja inválido ou inexistente, ou a senha não esteja relacionada ao e-mail, uma mensagem de erro é apresentada. A Figura 21 demonstra a apresentação de mensagem de erro quando o usuário tenta entrar no sistema sem inserir suas credenciais.

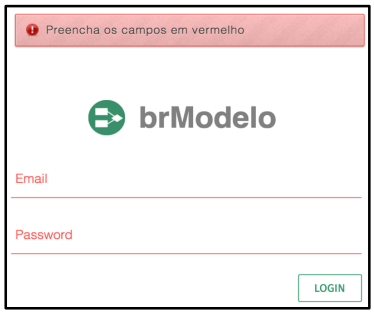


Figura 21 - Formulário de *login* com mensagem de erro
Fonte: elaborado pelo autor.

4.4.3 Lista de modelagens

Após o *login*, a primeira tela apresentada pela ferramenta é uma lista de modelagens. Nessa tela, todas as modelagens já criadas são visualizadas. Cada item dessa lista exibe as seguintes informações: tipo (conceitual, lógica ou física), nome, autor e data de criação. A Figura 22 apresenta uma captura de tela já com algumas modelagens salvas. Nela estão marcados ações que o usuário pode realizar:

1) Acessar uma área de configuração, onde é possível sair do sistema, ir para a área de ajuda ou editar as credenciais *login* e senha.
2) Criar uma nova modelagem.
3) Excluir uma modelagem já existente.
4) Editar uma modelagem.

Figura 22 - Tela área de trabalho
Fonte: elaborado pelo autor.

Essas funcionalidades são detalhadas a seguir.

4.4.4 Criar nova modelagem

Ao clicar em ‘NOVO MODELO’, uma janela modal é apresentada ao usuário para que o mesmo preencha o nome da modelagem e o tipo (CONCEITUAL ou LÓGICO) que deseja criar. Esses campos são obrigatórios e devem ser preenchidos.

Para concluir a ação, basta clicar em salvar. A Figura 23 apresenta a janela modal de criação de modelagem.

Figura 23 - Modal criação de modelagem
Fonte: elaborado pelo autor.

4.4.5 Módulo modelagem conceitual

A Figura 24 exibe uma captura de tela do módulo de modelagem conceitual. As principais áreas de interação estão identificadas por marcadores numéricos conforme segue:

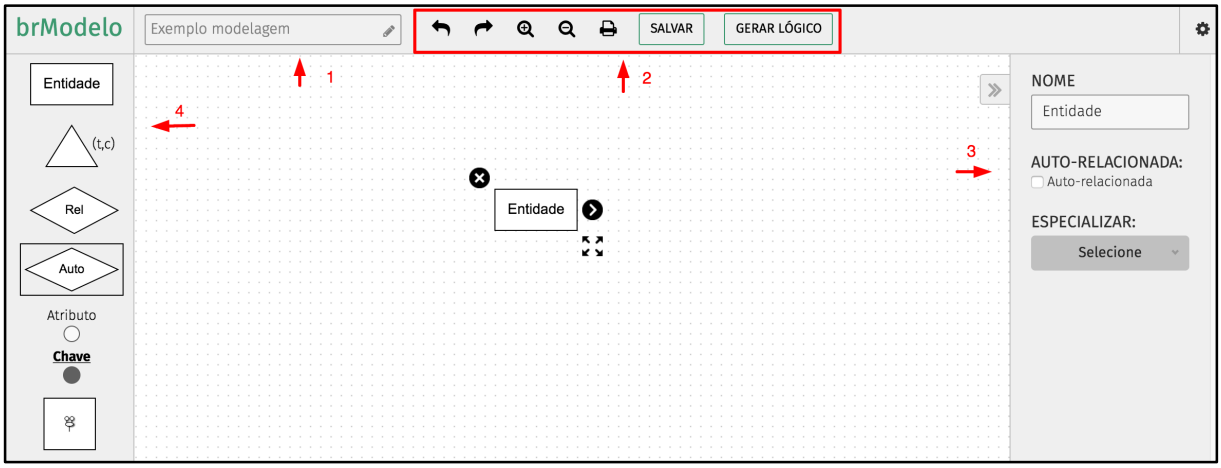


Figura 24 - Módulo de modelagem conceitual
Fonte: elaborado pelo autor.

1) Nesse campo é possível realizar a edição do nome da modelagem. Para que a ação seja persistida no BD, o botão SALVAR deve ser pressionado em seguida.

2) Nessa área são exibidas as ações de desfazer e refazer alterações realizadas nas modelagens, *zoom*, impressão, salvar e converter. Toda vez que o usuário desejar persistir suas ações no BD, o botão SALVAR deve ser pressionado.

3) Nessa área ficam as opções de edição. Para que o usuário tenha mais área de modelagem disponível, é possível manter essa área fechada enquanto os elementos são inseridos e conectados. Para que as opções de edição sejam exibidas, é necessário que um elemento esteja selecionado.

4) No lado esquerdo estão as opções de criação com uma paleta de objetos que podem ser incluídos no modelo. Essa barra de objetos disponibiliza os conceitos básicos do modelo ER: entidades; relacionamentos; entidade associativa; especialização; atributo (simples e multivalorado) e atributo identificador. Para inserir esses elementos na área de modelagem, basta arrastá-los para a área de edição. Para realizar conexões entre elementos existem duas maneiras: é possível arrastar um elemento em cima de um já existente ou selecionar o elemento, clicar no ícone de conexão e arrastá-lo até o elemento que se deseja conectar. Quando uma conexão inválida for realizada, a ferramenta desfaz a ação e avisa o usuário por meio de uma mensagem de erro. Ao selecionar cada elemento, um conjunto de ações comuns a todos são exibidos como: exclusão, edição de nome, conexão e redimensionamento de tamanho. As ações personalizadas dos principais elementos são:

   a) **Entidade:** ao selecionar uma entidade, uma janela semelhante a exibida na Figura 25 é mostrada, onde o nome da entidade é informado. Também é possível declarar a entidade como auto relacionada. Neste tipo de ação, um relacionamento é automaticamente criado. Ainda, pode-se especializar a entidade indicando o tipo de especialização desejada: total e compartilhada (t,c), total e disjunta (t,d), parcial e compartilhada (p,c) e parcial e disjunta (p,d). Essa ação também pode ser realizada arrastando o componente de especialização para dentro da área de edição, conectando-o com as entidades.

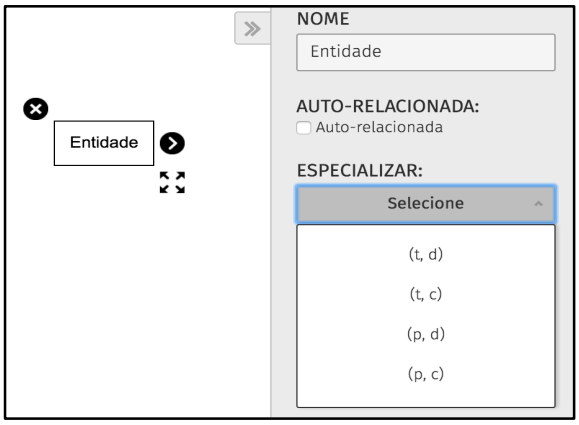


Figura 25 - Configurações de entidade
Fonte: elaborado pelo autor.

b) **Atributo:** ao selecionar um atributo, as opções de CARDINALIDADE e TIPO são exibidas, conforme exemplifica a Figura 26. Ao identificar um atributo como composto, a ferramenta cria e conecta este atributo a outros dois atributos para facilitar a visualização. No entanto, o usuário pode continuar adicionando quantos atributos achar necessário na composição. A opção CARDINALIDADE serve para definir um atributo multivalorado.

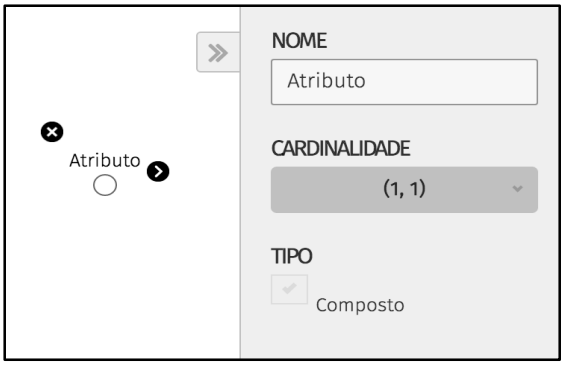


Figura 26 - Configurações de atributo
Fonte: elaborado pelo autor.

c) **Relacionamento:** para realizar a edição de um relacionamento, ao invés de se clicar no elemento relacionamento, deve-se clicar na linha que conecta o relacionamento com o elemento entidade, conforme exemplificado na Figura 27. Essa decisão de usabilidade foi tomada, pois um relacionamento pode ter muitas conexões, tornando a ação confusa. As opções exibidas são de CARDINALIDADE, que representa o número mínimo e máximo de ocorrências desta entidade. Nessa

edição também é possível marcar o relacionamento como FRACO. Neste caso, define-se uma entidade fraca, ou seja, as ocorrências da entidade cujo o relacionamento é marcado por uma linha mais densa somente irá existir quando ocorrências da entidade relacionada a elas existirem.

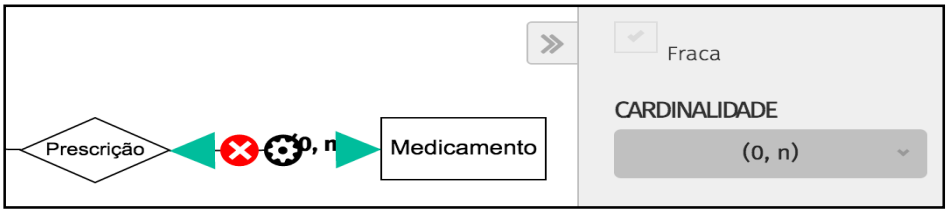


Figura 27 - Configurações de relacionamento
Fonte: elaborado pelo autor.

4.4.6 Módulo modelagem lógica

A Figura 28 exibe a captura de tela do módulo lógico. Esse módulo partilha das mesmas funcionalidades de configuração (desfazer, *zoom* etc.) do módulo de modelagem conceitual. Assim sendo, essas funcionalidades não são novamente descritas nessa seção. As funcionalidades exclusivas à modelagem lógica são as seguintes:

- Para adicionar elementos à modelagem, basta arrastá-los à área de edição. Diferentemente das versões anteriores da ferramenta *brModelo*, onde era possível também adicionar chaves à modelagem, apenas elementos do tipo tabela são possíveis de serem adicionados nesta primeira versão da *brModeloWeb*. As chaves e os relacionamentos são definidos ao realizar conexões ou ao se editar um elemento.

- Ao selecionar uma tabela, é possível editar seu nome, bem como adicionar, excluir ou editar colunas. Ao selecionar uma coluna na barra de edição, é possível editar seu tipo, transformá-la em chave (primária, estrangeira ou ambas) e inserir observações. Assim como na etapa de modelagem conceitual, é possível editar a cardinalidade das ligações. No entanto, o projeto lógico não permite mais ligações do tipo N:N (muitos-para-muitos),

pois tais ligações já são caracterizadas por tabelas associativas no nível lógico.

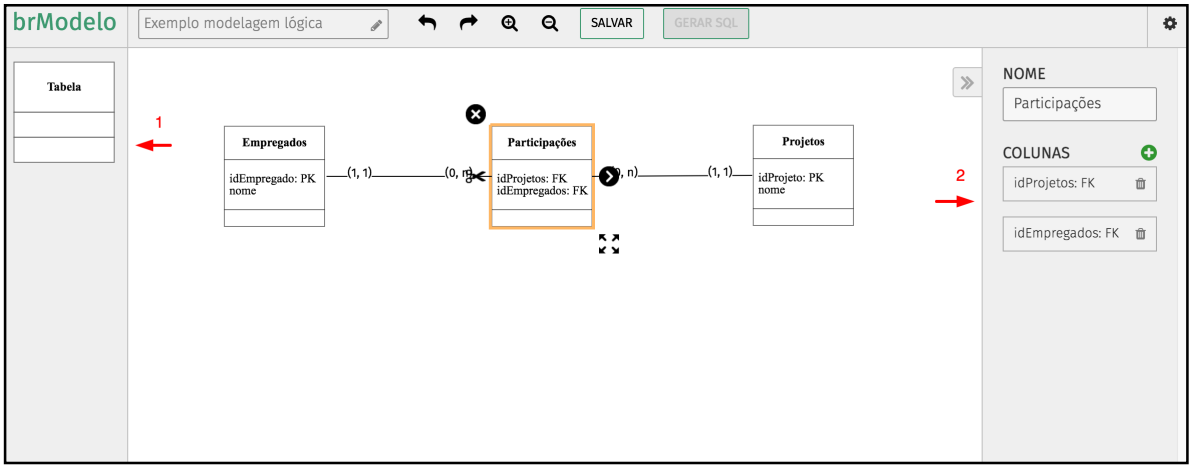


Figura 28 - Módulo de modelagem lógica
Fonte: elaborado pelo autor.

4.4.7 Conversão conceitual-lógico

O processo de conversão de uma modelagem conceitual para uma modelagem lógica inicia-se quando o usuário pressiona o botão GERAR LÓGICO na tela de edição de uma modelagem conceitual. Esse processo acontece de forma semiautomática, ou seja, a ferramenta auxilia o usuário no mapeamento de atributos, especializações e relacionamentos, quando há mais de uma alternativa de mapeamento, interagindo por meio de janelas que apresentam alternativas de escolha de mapeamento, conforme exemplificado na Figura 29. Essa etapa tem ênfase na eficiência do armazenamento dos dados, procurando evitar muitas tabelas e/ou evitar tabelas subtilizadas no BD relacional.

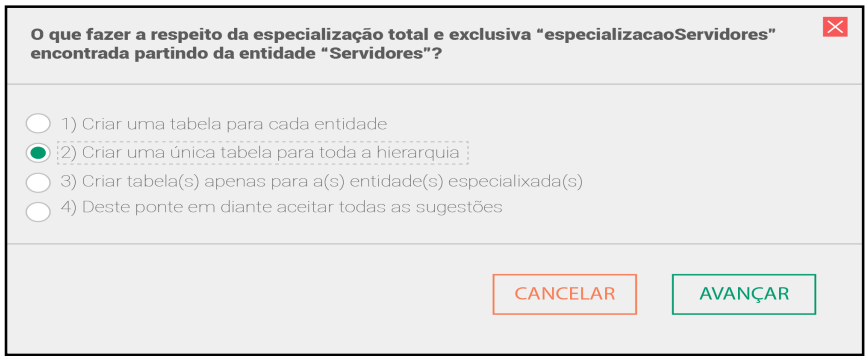


Figura 29 - Janela de ajuda de conversão
Fonte: elaborado pelo autor.

O mapeamento de uma modelagem conceitual para uma modelagem lógica ocorre na seguinte ordem, conforme apresentado por Heuser (2009):

1) Mapeamento de entidades e seus atributos;
2) Mapeamento de especializações;
3) Mapeamento de relacionamentos e seus atributos.

4.4.8 Conversão lógico-físico

O processo de conversão de uma modelagem lógica para uma modelagem-física é executado quando o usuário pressiona o botão GERAR SQL na tela de edição de uma modelagem lógica. O esquema lógico é convertido em uma especificação SQL/DDL de forma automática. A Figura 30 exemplifica o resultado desse processo de conversão realizado pela ferramenta.

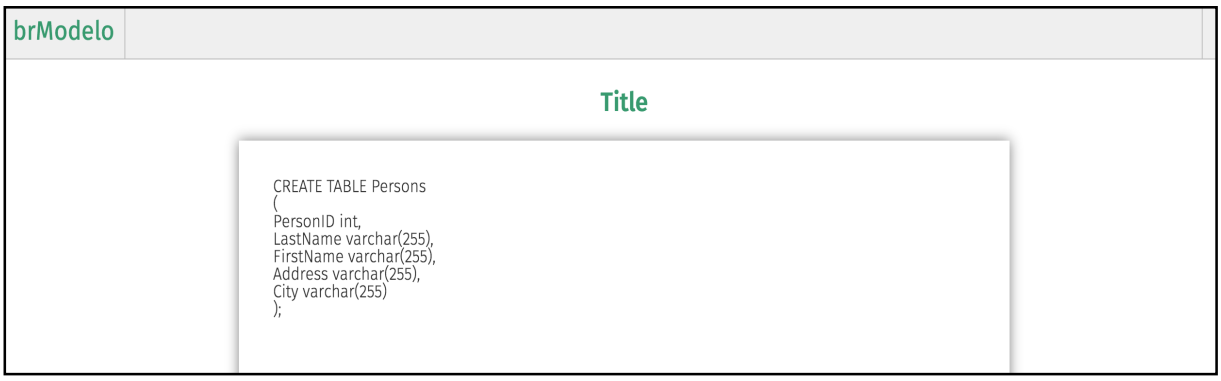


Figura 30 - Tela geração da modelagem física na linguagem SQL
Fonte: elaborado pelo autor.

Este capítulo apresentou os requisitos levantados para o projeto, as tecnologias e bibliotecas utilizadas no desenvolvimento da ferramenta, bem como a arquitetura da ferramenta e suas funcionalidades. De modo a não tornar este capítulo muito extenso, apenas as principais funcionalidades foram apresentadas.

Os *links* para a ferramenta e para o site tutorial podem ser encontrados no repositório *GitHub* da aplicação, onde é possível encontrar também todo o código, fazer sugestões e até colaborar com novas implementações. Esse repositório encontra-se disponível em <https://github.com/miltonbsn/brmodelo>.

5 ESTUDO DE CASO

Este capítulo apresenta um estudo de caso de utilização da ferramenta *brModeloWeb* na modelagem de um BD relacional. Dado um problema real, ele é modelado passo a passo na ferramenta. O capítulo está dividido em duas seções: a primeira descreve textualmente o problema e a segunda apresenta uma alternativa de modelagem gerada por meio da ferramenta. O domínio exemplo para esse estudo de caso é de um Museu. Apenas parte dos requisitos de dados deste domínio são detalhados na próxima seção.

5.1 DOMÍNIO: MUSEU

Cada obra no museu possui um código, um título e um ano. Obras são pinturas ou esculturas. No primeiro caso, é importante saber a área (m^2) e o tipo (por exemplo, impressionista). No caso das esculturas, elas se classificam em esculturas de pedra ou de metal, podendo uma escultura ser de ambos os materiais. Para qualquer escultura, é importante saber peso e altura. Para esculturas de metal, é importante saber o seu tempo de vida estimado e o tipo de metal. Para esculturas de pedra, é importante saber o tipo de pedra e o país de origem, isto é, de onde a pedra foi extraída.

Uma obra pode estar exposta em um único salão do museu, em uma posição específica neste salão. Um salão, que geralmente abriga várias obras, é identificado por um número e está em um andar do museu. Certos dados a respeito dos autores de cada obra também devem ser mantidos: código, nome, sexo e nacionalidade (país). Uma obra é produzida por apenas um autor, porém, pode existir mais de uma obra de um mesmo autor no museu.

No museu trabalham restauradores de obras, que possuem ID, CPF, nome e salário. Além disso, é necessário saber quais são os seus dias de atendimento no museu durante a semana. Um restaurador pode estar trabalhando em uma única obra, ou seja, realizando sua manutenção. Uma obra, caso esteja em manutenção, está nas mãos de apenas um restaurador. Para cada manutenção, deve-se registrar a data de início e a data prevista de término do trabalho, uma descrição do serviço a ser feito e um custo previsto para realizar a manutenção. Uma manutenção pode estar utilizando uma ou mais matérias-primas que, por sua vez, possuem um código, um nome e uma

quantidade em estoque. Uma matéria-prima pode estar sendo utilizada em várias manutenções, em uma certa quantidade. Fornecedores são responsáveis pelo fornecimento de uma ou mais matérias-primas ao museu. Nada impede que a mesma matéria-prima seja fornecida por mais de um fornecedor. Para cada fornecedor, é importante saber CNPJ, o nome e o endereço (rua, número, complemento, cidade, UF e CEP).

5.2 MODELAGEM

Os passos necessários para realizar o projeto de BD do domínio apresentado na sessão 5.1 por meio da ferramenta *brModeloNext* são:

- **Passo 1: Criar conta;**
- **Passo 2: Fazer *login;***
- **Passo 3: Criar modelagem conceitual;**
- **Passo 4: Modelagem conceitual;**
- **Passo 4.1: Identificar entidades;**

A partir da descrição do problema, foi possível identificar as entidades: ‘Salão’, ‘Obra’, ‘Restaurador’, ‘Matérias-Primas’, ‘Fornecedores’ e ‘País’. Para definir essas entidades utilizando a ferramenta, entidades do menu da esquerda devem ser arrastados para a área de modelagem. É importante renomear as entidades arrastadas para que elas façam sentido na modelagem, selecionando o objeto e editando o campo NOME no menu da direita.

Uma situação especial identificada no texto é a especialização: obras são Pinturas ou Esculturas. É possível realizar essa especialização de duas maneiras: selecionando a entidade ‘Obra’ já criada e escolhendo uma das opções no combo ESPECIALIZAR, ou arrastando duas novas entidades e um objeto do tipo especialização, conectando-os a elas. Neste caso, é importante que a primeira entidade ligada ao objeto que representa a especialização seja ‘Obra’, para que o programa possa identificar que esta entidade seja a especializada. O mesmo deve ser feito para a entidade ‘Escultura’, uma vez que Escultura se especializa em escultura

de Pedra ou de Metal. O resultado esperado da interação é apresentado na Figura 31.

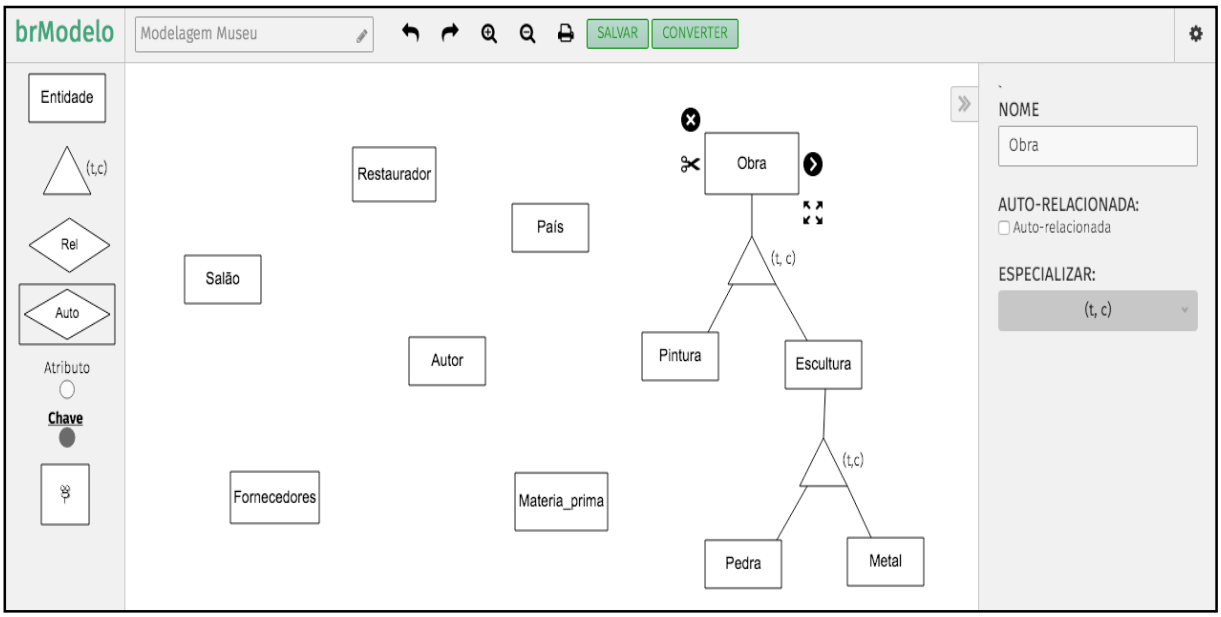


Figura 31 - Estudo de caso: definição de entidades
Fonte: elaborado pelo autor.

- **Passo 4.2: Identificar atributos;**

Após a análise do texto, foi possível identificar os seguintes atributos para as entidades criadas:

- Autor: Código (identificador) e Nome.
- Salão: Número (identificador) e Andar.
- Restaurador: ID (identificador) Salário, CPF, NOME e Dias_Atendimento.
- Autor: Código (identificador), Nome e Sexo.
- Fornecedores: CNPJ (identificador), Nome e Endereço (CEP, Cidade, Complemento, Número, Rua, UF).
- Matéria_Prima: Código (identificador), Nome e Quantidade.
- Obra: Código (identificador), Título, Ano e Descrição.
- Escultura: Peso e Altura.
- Pintura: Área e Tipo.
- Metal: Tipo_Metal e Tempo_Vida.
- Pedra: Tipo_Pedra.

Para conectar um atributo ou chave a uma entidade, basta arrastá-lo sobre a entidade. Também é possível arrastar o atributo até a área de modelagem e, após isso, conectá-los manualmente, selecionando um dos elementos e arrastando a ligação até o outro.

Duas situações especiais aparecem na modelagem de atributos. Primeiro, o atributo ‘Endereço’ da entidade ‘Fornecedores’ caracteriza-se como composto (Rua, Complemento, Número etc.). As duas formas de realizar essa modelagem são: ligar esses atributos derivados ao atributo ‘Endereço’, ou selecionar ‘Endereço’ e marcar o *checkbox* COMPOSTO na área de edição à direita. O outro caso especial é o atributo ‘Dias_Atendimento’ da entidade ‘Restaurador’. Este atributo caracteriza-se por ser multivalorado. Para marcar o atributo como multivalorado, a CARDINALIDADE na área de edição (1, N) deve ser selecionado. Após a definição dos atributos, a modelagem resultante é mostrada na Figura 32.

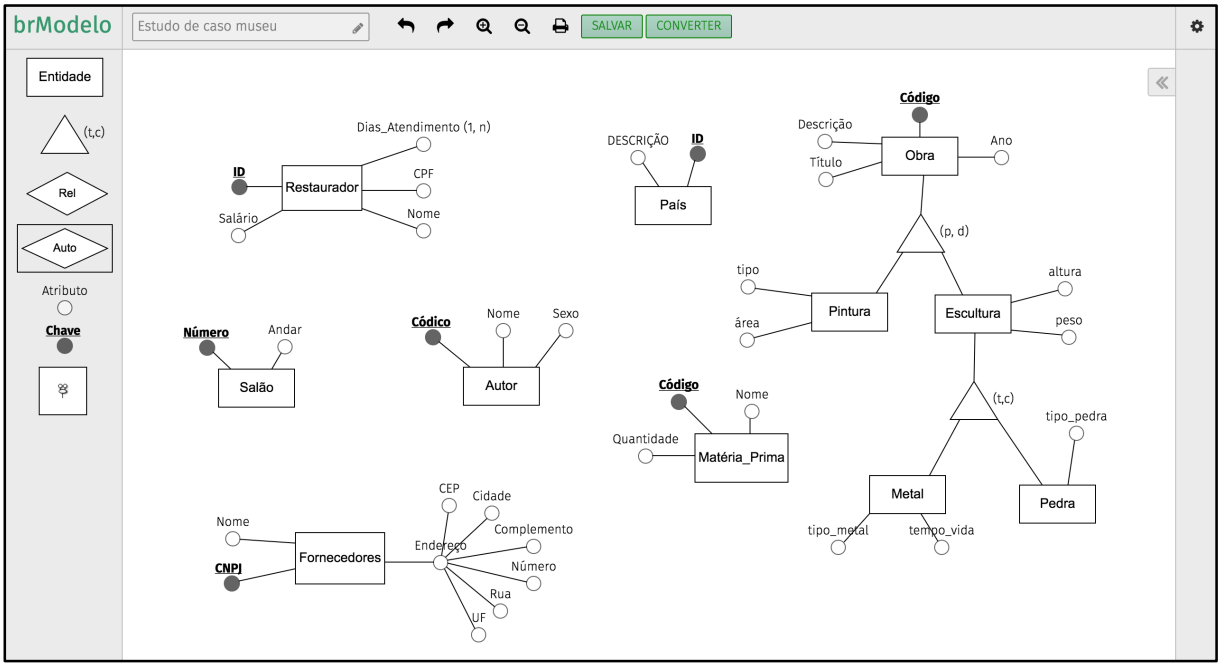


Figura 32 - Caso Museu: definição de atributos
Fonte: elaborado pelo autor.

- **Passo 4.3 – Identificar relacionamentos;**

Por fim, é necessário definir os relacionamentos e suas cardinalidades. As alternativas para se relacionar entidades na ferramenta são selecionar uma entidade e puxar a linha de ligação até a entidade que se deseja relacionar (essa alternativa

cria um relacionamento automaticamente); trazer o relacionamento para a área de modelagem e fazer a ligação entre os elementos de forma manual; ou, ainda, arrastar uma entidade sobre outra (esta alternativa também cria um relacionamento). Para definir as cardinalidades, é preciso passar o *mouse* sobre a linha que liga a entidade com o relacionamento, apertar no ícone de engrenagem e, então, selecionar a cardinalidade desejada.

Embora existam outras formas de se modelar o fato ‘manutenção’ em nível conceitual, surgiu a oportunidade se fazer uso do conceito de entidade associativa para representá-la, considerando-a como um relacionamento entre as entidades ‘Obra’ e ‘Restaurador’ que, por sua vez, associa-se com ‘Matéria_Prima’. Para realizar esta operação, basta selecionar o relacionamento e clicar em “Transformar em associativa" na área de edição. A Figura 33 apresenta a modelagem conceitual completa para o domínio do Museu.

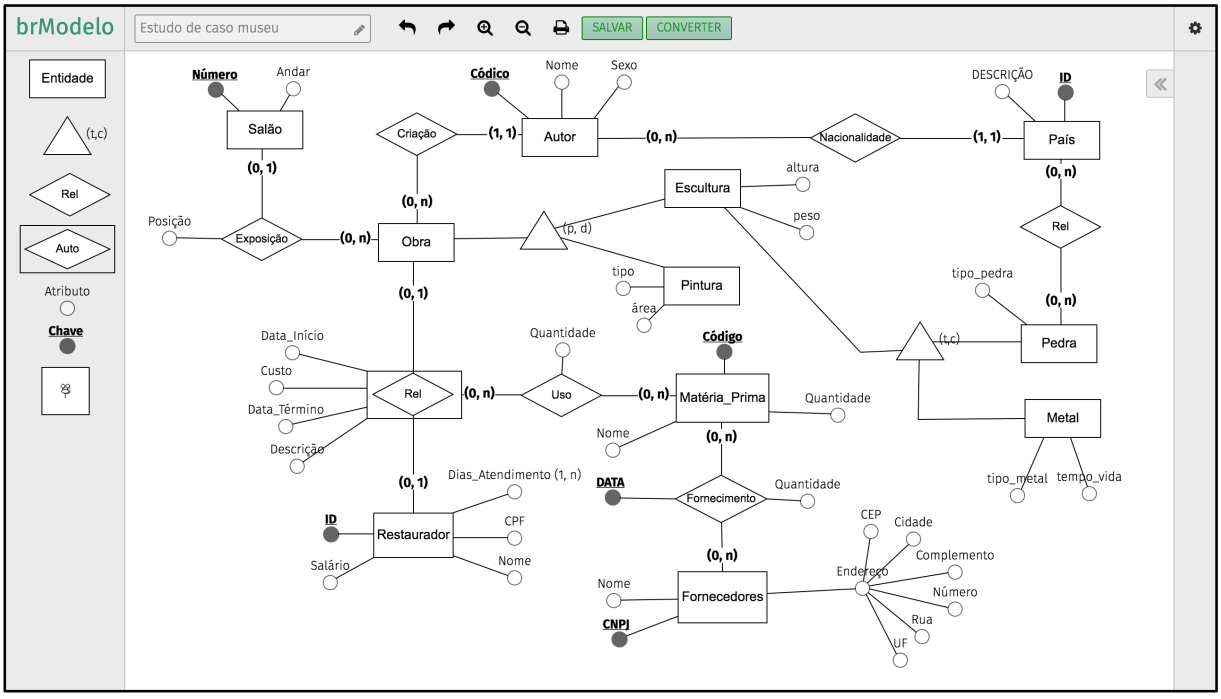


Figura 33 - Caso museu: definição de relacionamentos
Fonte: elaborado pelo autor.

- **Passo 5: Gerar projeto lógico;**

O próximo passo é a geração da modelagem lógica. Para iniciar o processo de conversão, basta pressionar o GERAR LÓGICO no menu de opções. A ferramenta interage diversas vezes com o usuário para solucionar o mapeamento do atributo

composto Endereço, do atributo multivalorado Dias_atendimento, das especializações Obra e Escultura, da entidade associativa Manutenção e dos demais relacionamentos. A cada interação, a ferramenta apresenta algumas alternativas e sugere a mais propícia para cada situação. Para prosseguir, basta pressionar o botão CONTINUAR nas janelas abertas.

Após o final da conversão, pode ser necessário ajustar manualmente a posição das tabelas geradas. É possível editar o TIPO das colunas geradas a partir do mapeamento dos atributos, tornando a geração do projeto físico mais fiel ao mundo real. Para realizar essa operação, deve-se selecionar uma coluna e escolher uma das opções do campo TIPO. O resultado da modelagem lógica é apresentado na Figura 34.

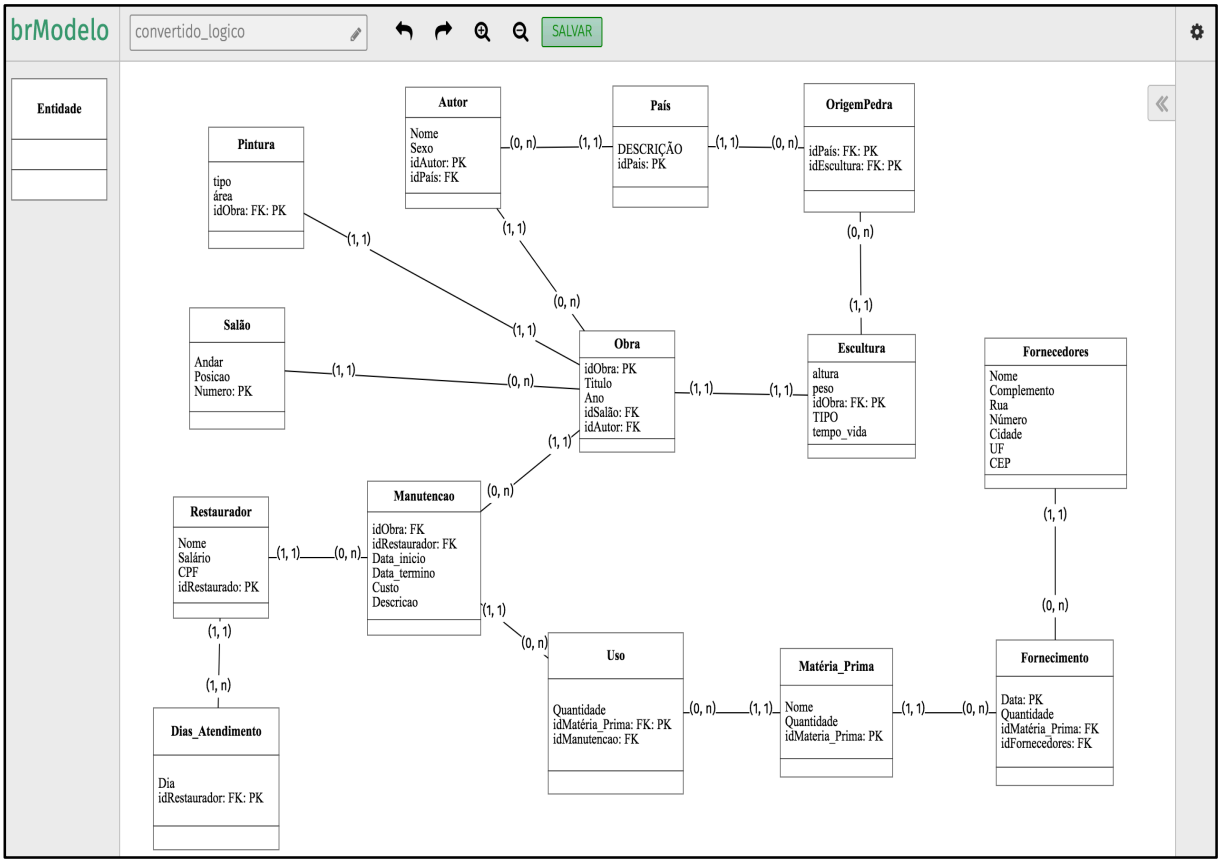


Figura 34 - Resultado da modelagem lógica
Fonte: elaborado pelo autor.

- **Passo 6: Gerar projeto físico;**

A modelagem física pode ser gerada utilizando o botão GERAR SQL, no *menu* de opções na tela de modelagem lógica. Como produto desta conversão, um código

SQL genérico do tipo ANSI é gerado, concluindo-se, assim, o processo de modelagem do BD relacional.

Um exemplo de um trecho de código SQL gerado para este estudo de caso é mostrado na Figura 35.

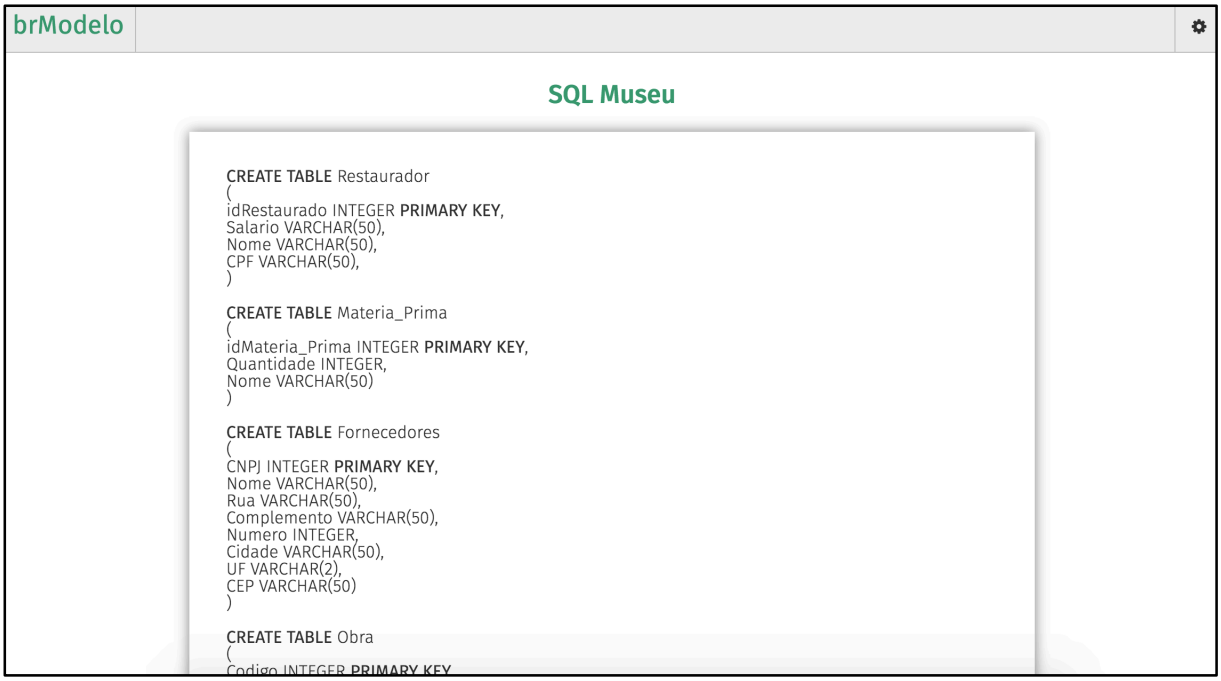


Figura 35 - *Script* de criação das tabelas do BD Museu
Fonte: elaborado pelo autor.

6 AVALIAÇÃO

Este capítulo apresenta uma avaliação de usabilidade realizada na ferramenta *brModeloWeb* por alunos da disciplina INE5613 - Bancos de Dados I, pertencente ao curso de Sistemas de Informação. O objetivo desta avaliação é validar a execução do estudo de caso proposto no capítulo 5, bem como testar a usabilidade da ferramenta a fim de encontrar pontos a serem melhorados. As perguntas respondidas com a avaliação foram as seguintes:

- Os usuários conseguiram realizar a modelagem do Museu na *brModeloWeb*?
- A pontuação média do SUS, questionário para medir a usabilidade de sistemas, foi superior a 75%? (Ver Seção 6.1)
- Quais os pontos positivos do sistema?
- Quais os pontos negativos do sistema?
- Em quais etapas notou-se dificuldade na interação?

6.1 QUESTIONÁRIO SUS

Para realizar a avaliação foi utilizado o questionário SUS - System Usability Scale - SUS (BROOKE, 1996). Esse método é considerado um padrão para testes de usabilidade de *software* por apresentar um balanço entre ser validado cientificamente e ao mesmo tempo não ser muito complexo. O SUS pode ser utilizado para avaliar produtos, serviços, *hardware*, *software*, *websites*, aplicações—e qualquer outro tipo de interface. Os critérios que o SUS ajuda a avaliar são:

- Efetividade;
- Eficiência;
- Satisfação.

O questionário consiste de 10 perguntas, e para cada uma delas o usuário pode responder em uma escala de 1 a 5, onde 1 significa Discordo Completamente e 5 significa Concordo Completamente. O cálculo da pontuação final do usuário é realizado da seguinte maneira:

1) Para as respostas ímpares (1, 3, 5, 7, 9), subtrair 1 da pontuação dada pelo usuário;
2) Para as respostas pares (2, 4, 6, 8, 10), aplicar o módulo da subtração de 5 e a resposta do usuário;
3) Somar todos os valores das dez perguntas e multiplicar por 2.5.

A é sua pontuação final pode ir de 0 a 100.

Com o intuito de se encontrar pontos a serem melhorados na ferramenta, ao final do questionário SUS foram acrescidas as seguintes perguntas:

1. O que eu mais gostei do sistema?
2. O que eu menos gostei do sistema?
3. Sugestão de melhoria para o sistema?

O questionário SUS aplicado aos alunos da referida turma encontra-se no Anexo I.

6.2 EXECUÇÃO DA AVALIAÇÃO

A execução da avaliação aconteceu no dia 10 de novembro de 2016 durante uma aula da referida disciplina, ministrada pelo orientador deste projeto. O teste foi realizado no laboratório 9 do CTC – Centro tecnológico da Universidade Federal de Santa Catarina com 18 alunos. Os alunos foram orientados sobre o funcionamento e propósito do teste, realizaram a modelagem do Museu e, no final, responderam o questionário SUS. Cabe salientar que o assunto Modelagem de Banco de Dados faz parte do plano de ensino da referida disciplina e que os alunos da turma já haviam tido aulas teóricas e práticas sobre este assunto.

6.3 RESULTADOS DA AVALIAÇÃO

Como resultado da analise de eficácia, todos os usuários conseguiram completar a modelagem proposta como roteiro do teste, garantindo uma eficácia de 100% para a avaliação.

A Figura 36 representa o gráfico de pontuação do SUS por usuário. A avaliação apresentou um desempenho de pontuação média de 82%, enquanto o valor esperado pelos requisitos de usabilidade correspondia a 75%. Segundo Sauro (2011), a média dos questionários SUS é de 68%. Assim sendo, o resultado ficou 14% acima da média dos questionários SUS e 7% acima do valor esperado, confirmando a aceitação dos usuários.

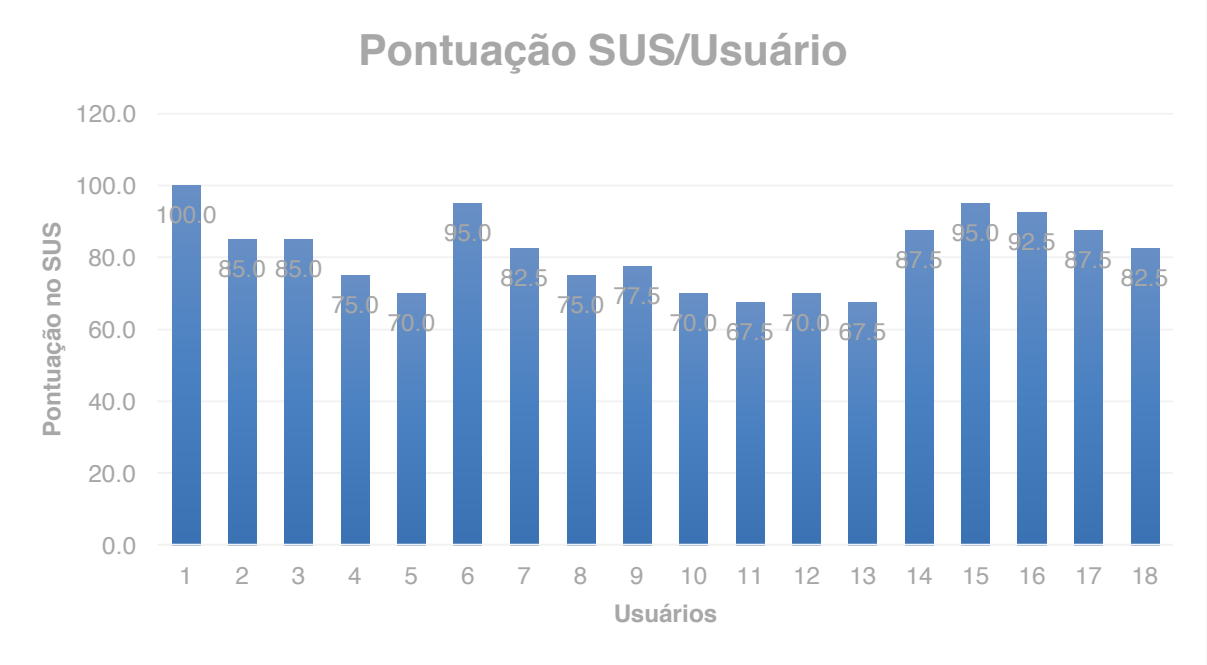


Figura 36 - Gráfico Pontuação SUS/Usuário para a ferramenta brModeloWeb.
Fonte: elaborado pelo autor.

As três perguntas acrescentadas ao questionário do SUS, mencionadas no início deste capítulo, contribuíram para identificar os pontos fortes, fracos e novas melhorias para a ferramenta, conforme indicado a seguir.

Pontos Fortes

Dentre os vários pontos citados pelos usuários, os que mais se repetiram foram:

1. Interface simples e intuitiva;
2. Facilidade em adicionar elementos de modelagem por meio do *drag and drop*;
3. Facilidade em iniciar as modelagens.

Pontos Fracos

Dentre os problemas criticados pelos usuários destacam-se:

1. Os atributos não movem junto com a entidade quando a mesma é movida;
2. Dificuldade em editar o nome dos componentes;
3. Alguns ícones são muito pequenos;
4. O processo de modelagem exige muitos cliques.

Sugestões de Melhorias

Segundo a opinião dos usuários, as seguintes melhorias ao software deveriam ser feitas:

1. Mover e dimensionar atributos ao mover entidades;
2. Salvar automaticamente modelagem no processo de conversão;
3. Implementar teclas de atalho;
4. Selecionar mais de um elemento ao mesmo tempo.

6.4 DISCUSSÃO

Esse capítulo apresenta uma avaliação realizada sobre a ferramenta proposta neste projeto e seus resultados. Os resultados dos testes de usabilidade indicam que a ferramenta pode realizar as etapas de modelagem de forma funcional e prática. A ferramenta foi considerada como boa pelos entrevistados, destacando-se pela simplicidade, possuir interface gráfica bastante intuitiva e ser fácil de se iniciar o uso. Após o teste, muitos dos pontos sugeridos como melhoria, como o salvamento automático durante as conversões, multi-seleção de elementos e implementação de teclas de atalho, foram implementados.

7 CONCLUSÃO

Este trabalho de conclusão de curso teve como objetivo principal o desenvolvimento de uma ferramenta *web* para auxiliar o processo de projeto de BD relacional, com ênfase no ensino e no aprendizado de técnicas de modelagem de BD baseado no modelo entidade-relacionamento. A principal motivação para este trabalho é a sua utilização em disciplinas da área de BD para atividades práticas de modelagem de dados.

Para atingir esse objetivo, foi realizado um levantamento bibliográfico a respeito de projeto de BD relacional, um estudo teórico sobre as tecnologias envolvidas no processo de desenvolvimento de *software web* e uma análise dos trabalhos relacionados ao tema de projeto de BD, especificamente as ferramentas desktop *brModelo* e *brModeloNext,* e ferramentas *web* similares, a fim de se levantar os requisitos e potenciais diferenciais da nova ferramenta, batizada de *brModeloWeb*.

A *brModeloWeb* destaca-se das demais ferramentas disponíveis no mercado de *software* pelas seguintes razões: ser *web*, estar, desse modo, disponível a qualquer dispositivo com acesso a um navegador; cobrir as três etapas de projeto de BD (conceitual, lógica e física); realizar a conversão entre modelagens; armazenar modelagens; apresentar tutoriais; interagir com o usuário, sempre que necessário, durante a realização de um projeto; e, apresentar interface gráfica simples e agradável, além de ser totalmente gratuita.

A *brModeloWeb* foi projetada com técnicas modernas de desenvolvimento *web* que possibilitaram o cumprimento dos requisitos funcionais e não funcionais apresentados na seção 4.1, resultando em uma ferramenta leve, escalável e fácil de se usar.

O estudo de caso apresentado no capítulo 5 e a análise de usabilidade realizada no capítulo 6 demonstram que, por meio da *brModeloWeb*, é possível modelar um problema real passando por todas as fases de projeto de BD e a ferramenta é viável do ponto de vista de utilização. Dessa forma, acredita-se que este trabalho cumpriu seu objetivo e contribui para a comunidade acadêmica por facilitar o ensino e o aprendizado de teorias e técnicas de projeto de BD.

Sabe-se que algumas funcionalidades podem ser adicionadas, a fim de agregar maior maturidade à ferramenta, possibilitando assim sua melhoria contínua. Dessa forma, apresentam-se as seguintes possibilidades de trabalhos futuros:

- Criação de um módulo para colaboração entre usuários em tempo real;
- Adição, na geração do esquema físico, de suporte para vários SGBDs como *Oracle*, *SQL*, *Server*, *MySQL* e *PostgreSQL*;
- Implantação de outras propostas de notação na modelagem conceitual;
- Geração de modelagens lógicas e físicas para outros modelos de BD e SGBDs relacionados, como por exemplo, *BD NoSQL* e o *SGBD MongoDB*;
- Internacionalização do *software* gerando versões em língua inglesa e espanhola.

REFERÊNCIAS

BROOKE, J. (1996). "SUS: a "quick and dirty" usability scale". In P. W. Jordan, B. Thomas, B. A. Weerdmeester, & A. L. McClelland. Usability Evaluation in Industry. London: Taylor and Francis.

CÂNDIDO, Carlos Henrique. **Aprendizagem em banco de dados:** implementação de ferramenta de modelagem E.R. 2005. Monografia (Especialização em Banco de Dados) - Pós-Graduação em Banco de Dados, Universidade Federal de Santa Catarina - UFSC, Universidade de Várzea Grande - UNIVAG, Várzea Grande-MT, 2005.

CHEN, Peter Pin-Shan. **The entity of relationship model:** toward a unified view of data. Massachusetts Institut of Technology. Association for Computing Machinery: Massachusetts, 1976. Disponível em: <http://www.csc.lsu.edu/~chen/pdf/erd-5-pages.pdf>. Acesso em: 20 set. 2016.

CLIENT.IO. **JointJS**. 2013. Disponível em: <http://www.jointjs.com/> Acesso em: 10 nov. 2015.

CROCKFORD, Douglas. **JavaScript:** the good parts. Sebastopol: O'Reilly, 2008.

JEFF SAURO. Measuring Usability with the System Usability Scale (SUS) Disponível em: <http://www.measuringusability.com/sus.php>. Acesso em: 15 ago. 2013

FERREIRA, Janaina Liziane; ARANTES, Álisson Rabelo. LEaRning: uma ferramenta didática para projeto conceitual de bancos de dados. In: II SIMPÓSIO MINEIRO DE SISTEMAS DE INFORMAÇÃO, 2005, Belo Horizonte - MG. **Anais...** Belo Horizonte: II Simpósio Mineiro de Sistemas de Informação, 2005. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/smsi/2005/002.pdf>. Acesso em: 25 ago. 2016.

FIELDING, R. **Architectural styles and the design of network-based software architectures**. 2000. 100f. Tese (Doutorado) - University of California, Los Angeles, CA, 2000.

FLANAGAN, David. **JavaScript:** the definitive guide. 6. ed. Sebastopol: O'reilly, 2011.

FOWLER, M. **Patterns of enterprise application architecture.** [S.l.]: Addison-Wesley Professional, 2003.

HEUSER, C. A. **Projeto de banco de dados.** Porto Alegre: Bookman, 2009.

MAGALHÃES, Rafael L.; NETO, Michelle M. F. **AprenDER:** ferramenta de apoio à construção de diagrama entidade relacionamento para deficientes visuais. 2007.

Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/smsi/2005/002.pdf>. Acesso em: 25 ago. 2016.

MENNA, Otávio Soares; RAMOS, Leonardo Antonio; MELLO, Ronaldo Dos Santos. BrModeloNext: **Nova Versão de uma Ferramenta para Modelagem de Bancos de Dados Relacionais**. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS (SBBD), 16., 2011, Florianópolis: S.e., 2011. p. 1-7.

NODE.JS. **About Node.js.** 2012. Disponível em: <http://nodejs.org/about/>. Acesso em: 25 ago. 2016.

PEREIRA, Juliana; RESENDE, Antonio M. P. **Uma análise dos ambientes de ensino de banco de dados.** 2012. Disponível em: <http://homepages.dcc.ufmg.br/~juliana.pereira/papers/2012_SBSI/SBSI.pdf>. Acesso em: 19 ago. 2016.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. **Sistema de banco de dados.** 5. ed. São Paulo: Makron Books, 1999.

TANENBAUM, D. W. A. S. **Computer Networks**. 4. ed. Upper Saddle River, NJ, USA: Prentice Hall

ANEXO I – QUESTIONÁRIO SUS (SYSTEM USABILITY SCALE)

SYSTEM USABILITY SCALE

	Discordo 100%				Concordo 100%
1. Eu penso que usarei este sistema freqüentemente	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
2. Eu achei o sistema desnecessariamente complexo	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
3.Achei que foi fácil de usar o sistema	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
4. Eu penso que precisaria de ajuda para utilizar este sistema	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
5. Achei que as funções estavam bem integradas	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
6. Achei que havia muita inconsistência nesse sistema	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
7. Acredito que a maioria das pessoas aprenderia a utilizar rapidamente *	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
8. Tive muito incomodo em usar este sistema *	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
9. Senti muito seguro em utilizar este sistema	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5
10. Eu precisarei aprender muitas coisas antes de utilizar este sistema *	☐ 1	☐ 2	☐ 3	☐ 4	☐ 5

11. O que eu mais gostei no sistema .

R: _____

12. O que eu menos gostei no sistema.

R: _____

13. Sugestões de melhorias para o sistema.

R: _____