UNIVERSIDADE FEDERAL DE PERNAMBUCO

GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CENTRO DE INFORMÁTICA

ERE-CASE: FERRAMENTA CASE DE MODELAGEM CONCEITUAL COM ENTIDADE-RELACIONAMENTO ESTENDIDO PARA BANCO DE DADOS

TRABALHO DE GRADUAÇÃO

Novembro de 2008

ERE-CASE: Ferramenta CASE de Modelagem Conceitual com Entidade-Relacionamento Estendido para Banco de Dados

por

Renan Pereira Gouveia de Lima

Monografia apresentada ao Centro de Informática
da Universidade Federal de Pernambuco como
requisito parcial para obtenção do Grau de
Bacharel em Ciência da Computação.

Orientador: Fernando da Fonseca de Souza

Co-Orientador: Robson do Nascimento Fidalgo

Novembro 2008

“O Senhor é meu pastor, nada me faltará”

Salmo 23

“Possuir um bem, sem o partilhar, não tem qualquer atractivo.”

Séneca

“Hakuna Matata”

Timão e Pumba

Agradecimentos

Primeiramente a Deus por tornar possíveis os meus sonhos e ter colocado todas essas pessoas que cito aqui em minha vida, me ajudando, em alguns momentos “por linhas tortas”, mas necessárias, a trilhar meu caminho.

As duas pessoas mais importantes na minha vida: meu pai, José Airton, e minha mãe, Maria Auxiliadora. Foi com o carinho e dedicação de minha mãe e a serenidade e sabedoria de meu pai, que consegui passar por todos os problemas que tentaram me abater durante minha graduação.

Aos amigos que conquistei dentro do Centro de Informática que certamente serão lembrados por todos os momentos que compartilhamos juntos dentro destes quatro anos e meio de convívio. Em especial a dois grupos que me sinto integrante vitalício, cujos são compostos por pessoas excepcionais que me orgulho de ter como amigos: o G4 (e todos os seus agregados) e a Commit Solutions.

Aos companheiros da D’Accord, que desde o quarto período da Faculdade estiveram presentes me apoiando nas decisões tomadas e compreendendo nos momentos de preocupação com os projetos a entregar e provas a realizar.

Aos professores e funcionários do Centro de Informática, que sempre com dedicação se tornaram grandes responsáveis pelo meu sucesso. Em especial, ao orientador deste trabalho de graduação, professor Fernando Fonseca, que me ajudou e esteve sempre presente quando precisei.

Resumo

A grande quantidade de ferramentas que contemplam o modelo Entidade Relacionamento proposto por Chen [Chen, 1979], ofusca a evolução e criação de ferramentas que trabalham com o modelo Entidade-Relacionamento Estendido para bancos de dados mais complexos que acompanham a evolução das aplicações que necessitam de uma modelagem mais completa para seus dados. Com esta carência de ferramentas para modelagem conceitual com Entidade-Relacionamento Estendido, tanto em meios acadêmicos quanto mercadológicos, surge a proposta deste trabalho de graduação para a criação da ferramenta ERE-CASE.

A ERE-CASE, que tem código aberto e é multiplataforma, permite que o projetista de banco de dados crie de maneira intuitiva e fácil, modelos conceituais para o armazenamento de dados que supram as necessidades de suas aplicações, das mais simples, que apenas precisem da modelagem Entidade-Relacional, às mais complexas que necessitem de um modelo mais rico como o Entidade-Relacional Estendido.

Palavras chaves: banco de dados, modelagem conceitual, Entidade, Relacionamento, ERE, ER, CASE, Objeto-Relacional, orientação a objetos.

Abstract

The large amount of tools that uses the Entity Relationship model proposed by Chen [Chen, 1979] overshadows the development and creation of tools to work with Enhanced Entity Relationship model for more complex databases. Such systems follow the evolution of applications that need a more complete modeling system. This lack of conceptual tools for modeling Enhanced Entity Relationship model, in both academic and business environments has motivated the creation of the ERE-CASE tool.

ERE-CASE, which is open source and multiplatform, allows the database designers to create easily and intuitively, conceptual models for the storage of data that supply the needs of their applications, from the simpler ones, which only need the Entity Relationship modeling, to the more complex that require a richer model such as the Enhanced Entity Relationship model.

Keywords: database, conceptual modeling, entity, relationship, EER, ER, CASE, object oriented.

Sumário

1. Introdução 11
1.1. Motivação 12
1.2. Objetivos 13
1.3. Ferramentas CASE 13
1.4. Estrutura do Trabalho 14
2. Fundamentação conceitual 16
2.1. Modelagem Conceitual 16
2.2. Modelo ER e Banco de Dados Relacionais 17
2.3. Modelo ERE e Banco de Dados Orientados a Objetos 19
3. A Ferramenta ERE-CASE
3.1. Funcionalidades
3.1.1. Abrir e Salvar modelo em formato XML 24
3.1.2. Modelagem Conceitual do modelo Entidade relacionamento Estendido 25
3.1.3. Edição dos elementos do modelo 25
3.1.4. Geração automática de scripts para Banco de Dados 25
3.1. Funcionalidades 24
3.2. Especificação e Implementação
3.2.1. Diagramas 26
3.2.2. Padrões de Projeto 32
3.2.3. Persistência 33
3.2.4. Interface 33
3.2. Especificação e Implementação 25
3. A Ferramenta ERE-CASE 24
4. Um estudo de caso
4.1. Minimundo geral do modelo 36
4.2. Modelo Conceitual 37

4. Um estudo de caso 36
5. Conclusão 43
5.1. Contribuições 43
5.2. Limitações 44
5.3. trabalhos Futuros 44
Apêndice A - Persistência 49
Data e Assinaturas 51
Referências Bibliográficas 46

Lista de figuras

Figura 2.1 - Diagrama ER 17

Figura 2.2 – Diagrama ER – Estendido (Herança) 20

Figura 2.3 – Exemplos de restrição de disjunção e restrição de integralidade 21

Figura 2.4 – herança com subclasse tipo união 22

Figura 2.5 – Modelo ERE de uma galeria de arte multimídia....23

Figura 3.1 Diagrama de Pacotes....26

Figura 3.2 – Pacote model (diagrama de classes) 27

Figura 3.3 – Pacote *persistence* (diagrama de classes)....28

Figura 3.4 – Pacote *model_draw* (diagrama de classes)....29

Figura 3.5 – Pacote *graph* (diagrama de classes)....30

Figura 3.6 – Pacote *model_edit* (diagrama de classes)....31

Figura 3.7 – Pacote *gui* (diagrama de classes) 31

Figura 3.8 – Projeto de interface 34

Figura 3.9 – Interface final da ferramenta ERE-CASE....34

Figura 4.1 – Inserindo Entidades 37

Figura 4.2 – Tabela de edição de entidades 37

Figura 4.3 – inserindo heranças....38

Figura 4.4 – Tabela de edição de heranças 39

Figura 4.5 – Inserindo atributos....39

Figura 4.6 – Tabela de propriedades de um atributo....40

Figura 4.7 – Inserindo relacionamentos 41

Figura 4.8 – Tabelas de edição das conexões e de edição dos relacionamentos 41

Lista de Quadros

Quadro 1.1 – Análise de Competidores 14

Quadro 3.1 – Mapeamento entre classes do pacote *model* e pacote *graph* 29

1. Introdução

A etapa de análise e projeto no desenvolvimento de software é uma das mais importantes e, cada vez mais, vem crescendo e ampliando suas ferramentas e técnicas. A modelagem apóia o desenvolvedor na obtenção de uma visão mais clara dos requisitos previamente elicitados, simplificando e dando uma visão geral do que virá a ser a aplicação no futuro. Pode-se dividir a modelagem do software em dois grandes grupos: a modelagem da aplicação, na qual se tem, por exemplo, os modelos e diagramas de classe, fluxo da aplicação, entre outros e a modelagem do repositório de dados, onde o banco de dados é analisado e projetado, construindo-se seus modelos conceitual, lógico e físico [Sommerville, 2004]. O foco deste trabalho de graduação é a modelagem conceitual de dados.

Por se tratar de uma etapa intermediária do desenvolvimento de software, à modelagem, nem sempre é dada a importância que se deveria, sendo muitas vezes superficialmente analisada no ciclo de desenvolvimento. Com isso aumentam as possibilidades da construção mal sedimentada do software, aumentando bastante a incidência de falhas que somente serão detectadas futuramente, agregando com isso um maior custo na manutenção da aplicação. Para deixar mais rápido e prático o desenvolvimento, principalmente dessa fase de análise e projeto de software, existem as chamadas ferramentas CASE (Computer-Aided Software Engineering - Engenharia de software com o auxílio de Computador), que são um conjunto de aplicativos que auxiliam em toda fase de desenvolvimento do software, tanto na construção dos diagramas, quanto na geração de documentação e possivelmente de códigos-fonte.

Unindo a área de modelagem de banco de dados e as principais ferramentas CASE existentes, verifica-se que existe um grande número de ferramentas que auxiliam na modelagem do modelo Entidade-Relacionamento (ER) [Chen, 1979], que é um modelo de alto nível, bastante difundido, utilizado e conhecido por todos aqueles que usam bancos de dados como base de armazenamento de dados. Por toda essa popularidade, esse modelo conta com boa variedade de ferramentas de apoio à modelagem. As mais conhecidas ferramentas são: Rational Rose [Rose, 2008], Microsoft Office Visio [Visio, 2007], brModelo [brModelo, 2008] e JUDE [Jude, 2008].

Muitas dessas ferramentas CASE existentes apresentam funcionalidades bem específicas para modelagem de banco de dados, como é o caso da brModelo [brModelo, 2008], que é uma ferramenta voltada para modelagem de modelos Entidade-Relacionamento. Outras, como é o caso do Microsoft Office Visio [Visio, 2007], podem ser adaptadas para a modelagem de banco de dados, mas não são especificamente projetadas para tal.

Com a evolução das aplicações existentes, fica cada vez mais difícil para o projetista de banco de dados modelar o armazenamento de informações complexas utilizando o modelo ER tradicional. Com isso, desde o final dos anos 70, vários outros modelos semânticos de dados surgiram para que essas novas informações fossem modeladas, e juntamente com esses novos modelos semânticos surgiu o conceito de estender o modelo ER visando à modelagem orientada a objetos, criando assim o conceito de Entidade-Relacionamento Estendido (ERE) [Elmasri & Navathe, 2005].

Por tratar-se de um conceito bem menos difundido que o modelo ER, não é comum encontrar-se ferramentas que auxiliem a modelagem Entidade-Relacional Estendida (ERE). Sistemas de Bancos de dados mais complexos que o tradicional sistema de banco de dados relacional (BDR), como é o caso dos sistemas de bancos de dados Orientados a Objetos (BDOO) e mesmo o Objeto-Relacional (BDOR), carecem de ferramentas simples e intuitivas para sua modelagem. Esta necessidade é realçada pelo crescimento de aplicações complexas, como é o caso dos Sistemas Geográficos, Sistemas de Armazenamento de Mídias, Sistemas de Telecomunicações, entre outros [Elmasri & Navathe, 2005].

O uso do modelo ER Estendido aproxima mais o modelo conceitual de dados às características presentes nas aplicações baseada no conceito de Orientação a Objetos, por possuir características que não são observadas no modelo ER tradicional, como: herança, polimorfismo e encapsulamento. Desta forma, é mais coerente que com o uso de aplicações Orientadas a Objetos, use-se como base de dados para estas um Banco de dados Orientado a Objetos e conseqüentemente que a modelagem desse banco seja construída com um modelo que dê suporte a todas as suas peculiaridades, como é o caso do modelo Entidade-Relacionamento Estendido.

1.1. Motivação

Dentro do cenário de ferramentas CASE para a modelagem de banco de dados, não se encontra na literatura uma ferramenta aberta e multiplataforma que dê suporte ao modelo Entidade-Relacionamento Estendido, mesmo considerando sua capacidade de modelagem. Portanto, ainda, não se consegue construir de forma automatizada, com o auxilio de uma ferramenta CASE um modelo Entidade-Relacionamento Estendido e mapeá-lo para o modelo lógico no banco de dados, tendo este modelo que ser gerado de forma improvisada por outras ferramentas que não são próprias para projetos de bancos de dados, como é o caso da ferramenta Microsoft Office Visio[Visio, 2007].

Esta improvisação na criação de modelos Entidade-Relacionamento Estendido faz com que erros e falhas, que poderiam ser identificados automaticamente com o auxilio de uma ferramenta CASE específica, passem despercebidos ao projetista de banco de dados, que diante da modelagem de sistemas complexos não pode validar seu projeto por não ter

nenhuma forma automatizada que informe onde os erros aconteceram e como seria a melhor forma dos mesmos serem corrigidos.

Essa limitação no que se diz respeito a ferramentas de modelagem para bancos de dados de sistemas mais complexos utilizando o modelo Entidade-Relacionamento (ER), como a incorporação de herança, noção de objetos, entre outras funcionalidades descritas no decorrer deste trabalho, demanda o desenvolvimento de ferramentas voltadas para o modelo Entidade-Relacionamento Estendido (ERE), tanto nos meios acadêmicos quanto nos profissionais.

1.2. Objetivos

O objetivo deste trabalho de Graduação é criar uma ferramenta CASE de modelagem conceitual, utilizando o modelo Entidade-Relacionamento Estendido. Esta ferramenta permitirá que o projetista de banco de dados crie, de maneira intuitiva e fácil, modelos conceituais para o armazenamento de dados que supram as necessidades de suas aplicações, das mais simples que apenas precisem da modelagem Entidade-Relacional (ER) às mais complexas que venham a necessitar de um modelo mais rico.

A ferramenta também permitirá que o usuário salve no formato XML [XML, 2008] os modelos construídos, facilitando o uso do modelo gerado para o projeto lógico e físico da base de dados, como é o caso de outro trabalho de graduação “ERE2Oracle: Geração Automática de esquema relacional estendido para Objeto-Relacional no SGBD Oracle” [Dantas, 2008].

A interoperabilidade da ferramenta ERE-CASE com a ferramenta ERE2Oracle, resultará numa potente ferramenta *open-source* para modelagem de banco de dados Objeto-Relacional utilizando o modelo Entidade-Relacionamento Estendido e a geração de esquema objeto-relacional no SGBD Oracle [Oracle, 2008].

1.3. Ferramentas CASE

Existem no mercado diversas ferramentas CASE destinadas a auxiliar o projeto conceitual de banco de dados. Muitas delas são exclusivas para modelos ER, como é o caso do brModelo [brModelo, 2008] e outras são abrangentes e genéricas o bastante para modelar qualquer tipo de modelo de software, como é o caso do Microsoft Visio [Visio, 2007]. A proposta do ERE-CASE é construir uma ferramenta que trabalhe exclusivamente para modelagem conceitual dos modelos ER e ER-Estendido. Como pode ser visto no Quadro 1.1, esse tipo de ferramenta não é comum, sendo apenas visto nas ferramentas Umbrello [Umbrello, 2008] e Sybase Power Design [Sybase, 2008], porém a Umbrello não opera no sistema operacional Windows e a Sybase Power Design possui uma licença de uso muito cara.

Por possuírem um alto nível de maturidade, muitas dessas ferramentas apresentam funcionalidades bastante interessantes e dignas de serem futuramente implementadas nas próximas evoluções da ferramenta ERE-CASE, que está apenas no seu primeiro ciclo de desenvolvimento. Funcionalidades estas como, por exemplo, a capacidade de engenharia reversa, geração automática de código e mapeamento de diferentes modelos conceituais e lógicos.

Quadro 1.1 – Análise de Competidores

Ferramenta	Freeware	Modelo conceitual	Modelo ER-Estendido	Conexão com BD	Sistema Operacional
brModelo	Sim	Sim	Não	Não	Windows
DBDesigner	Sim	Não	Não	Não	Linux
ERWin	Não	Não	Não	Sim	Windows
Oracle Designer	Sim	Sim	Não	Sim	Windows
Rational Rose	Não	Sim	Não	Sim	Windows
Sybase Power Design	Não	Sim	Sim	Sim	Windows
Umbrello	Sim	Sim	Sim	Sim	Linux
Visio 2007	Não	Sim	Não	Não	Windows

1.4. Estrutura do Trabalho

Este trabalho de graduação, além deste capítulo introdutório, está estruturado como segue.

O segundo capitulo, Fundamentação Conceitual, abordará as bases conceituais que fundamentam a ferramenta ERE-CASE, assim como mostrará algumas aplicações existentes na área de modelagem conceitual para Banco de Dados.

No terceiro capítulo, destinado à ferramenta ERE-CASE. Todas as etapas de engenharia de software aplicadas serão descritas, assim como os modelos e padrões utilizados para a sua construção.

No quarto capítulo será tratado um exemplo de uso, construindo-se um modelo Entidade-Relacionamento Estendido utilizando a ferramenta ERE-CASE. Serão abordadas as

diferentes etapas do processo, além de analisadas as facilidades e limitações da utilização da ferramenta para construção de modelos ERE.

O quinto capítulo apresentará as conclusões, contribuições do trabalho e as limitações da ferramenta ERE-CASE, além de apontar evoluções possíveis para a ferramenta em trabalhos futuros.

Em seguida serão listadas as referências bibliográficas e apresentado o Apêndice A.

2. Fundamentação conceitual

Neste capitulo, será apresentada uma visão geral dos fundamentos conceituais que sedimentam a ferramenta ERE-CASE. Será enfatizada, também, a importância da modelagem conceitual em um projeto de banco de dados, assim como os dois modelos de dados que são possíveis de serem construídos com o auxilio da ferramenta, o modelo ER e o modelo ER estendido.

2.1. Modelagem Conceitual

O processo de projeto de banco de dados passa por várias etapas, conforme requer a engenharia de software, antes que a implementação em si do banco comece a ser desenvolvida. Inicialmente, o levantamento dos requisitos funcionais e não-funcionais da aplicação, com o auxilio de técnicas como: entrevistas, questionários, formulários, entre outras, é, em geral, o ponto de partida para o conhecimento inicial do problema pelos projetistas da mesma. Passada essa fase de elicitação de requisitos, o analista precisa então organizar as requisições feitas pelos clientes e pessoas envolvidas com o projeto, em um documento - **Documento de Requisitos -** que servirá como base para a construção da aplicação.

Após a criação do documento de requisitos, com o auxilio principalmente dos requisitos funcionais referentes aos dados da aplicação, o projetista de banco de dados está pronto para conceber o **modelo conceitual** de dados da aplicação. Este é o primeiro dos três modelos de dados a serem construídos no processo de desenvolvimento do banco de dados, seguido posteriormente pelos modelos lógico e físico. O modelo conceitual é uma descrição concisa dos requisitos de dados dos usuários e inclui descrições detalhadas de tipos de entidades, relacionamentos e restrições, usando para isso uma visão de alto nível [Elmasri & Navathe, 2005]. Por se tratar de uma visão abstrata dos dados, o modelo conceitual não apresenta detalhes de implementação, como, por exemplo, tipo de SGBD que será utilizado para implantação do banco. Porém, esse modelo tem grande utilidade em facilitar o entendimento do problema no mundo real, passando o mesmo para uma visão mais próxima do que será futuramente implementado fisicamente. Além dessa facilidade para o projetista de entender o que será implementado, o modelo conceitual também auxilia o cliente a entender como será implantada a sua idéia no projeto, de maneira mais abstrata, fazendo com que o cliente se sinta mais próximo do que será implantado no seu sistema sem que detalhes técnicos precisem ser discutidos.

Para a modelagem conceitual dos dados, é necessário, porém, que o projetista tenha em mente, geralmente baseando-se em suas experiências passadas, quais as características gerais que o banco de dados terá. Diante dessa experiência poderá escolher que tipos de

modelagem e banco de dados usará, por exemplo, Modelo ER [Chen, 1979] e Banco de Dados Relacional [Spicer, 2003] ou Modelo ER-Estendido e Banco de Dados Orientado a Objetos.

2.2. Modelo ER e Banco de Dados Relacionais

O modelo ER (Entidade-relacionamento) foi proposto por Peter Chen, Ph.D. em Ciência da Computação aplicada à Matemática pela universidade de Havard, em 1976 com a publicação do seu mais famoso artigo *The Entity-Relationship Model – Toward a Unified View of Data* [Chen, 1979]. E é utilizado até os dias atuais como o principal modelo usado para modelagem conceitual de banco de dados.

Para o modelo ER, o mundo dos dados é representado em sua totalidade por **Entidades**, que representam estruturas e elementos do mundo real e seus **Relacionamentos** (associações semânticas). Como se trata de um modelo conceitual, questões de implementação são abstraídas e deixadas para as próximas fases do processo de desenvolvimento do banco de dados. Além das entidade e relacionamentos, o modelo ER apresenta a estrutura de atributos de ambos, entidades e relacionamentos. A representação diagramática contendo as entidades, atributos e relacionamentos do modelo ER é chamado de **Diagrama ER**. Na Figura 2.1 pode-se ver um diagrama ER simples, mostrando uma relacionamento entre duas entidades e seus respectivos atributos.

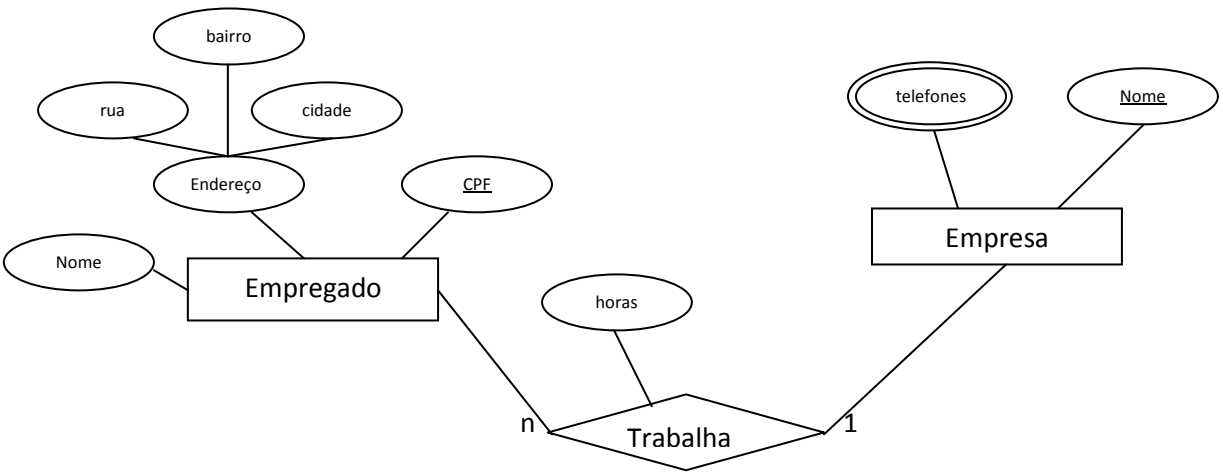


Figura 2.1 - Diagrama ER

No diagrama da Figura 2.1, tem-se duas entidades que representam estruturas do mundo real, a entidade **Empregado** e a entidade **Empresa**. Essas duas entidades estão relacionadas através do relacionamento binário (**Trabalha)** e estão representados pela cardinalidade 1-n, ou seja, uma empresa tem n empregados que trabalham para ela, mas um empregado trabalha em apenas uma única empresa**.** Um relacionamento entre entidades no modelo ER pode ser 1-N, N-1, N-M ou 1-1 [Elmasri & Navathe, 2005]. Quanto ao grau, a predominância é de relacionamentos binários, como o mostrado na Figura 2.1, em que apenas

duas entidades participam do relacionamento, porém relacionamentos de grau maior que dois também são utilizados e conhecidos como relacionamentos n-ários [Elmasri & Navathe, 2005].

Pode-se ter vários tipos diferentes de atributos especiais relacionados a uma entidade, que são classificados como: chave primária, chave estrangeira, multivalorado, composto e derivado [Elmasri & Navathe, 2005]. No exemplo mostrado na Figura 2.1, tem-se três destes. Os atributos sublinhados, CPF da entidade Empregado e nome da entidade Empresa, são considerados os atributos-chave (chave primária) das suas respectivas entidades. Estes atributos identificam e tornam cada instância das entidades Empregado e Empresa como única, impondo com isso uma restrição de unicidade a cada empregado ou empresa. Os atributos multivalorados são representados nesse diagrama por uma elipse dupla, como é o caso do atributo telefones na entidade Empresa. Um atributo também pode ter vários atributos associados, como é o caso do atributo endereço, na entidade Empregado. A este tipo de atributo dá-se o nome de atributo-composto. Os demais atributos mostrados na Figura 2.1 são atributos comuns.

O tipo Entidade ainda pode ser subdividido em entidade forte e entidade fraca. Uma entidade forte é uma entidade regular, exemplificada na Figura 2.1. Ela é caracterizada por ter atributo-chave e não depender de nenhuma outra entidade para existir. Já o tipo entidade fraca não possui atributo-chave que isoladamente a identifique unicamente e por isso depende da existência de uma entidade forte para que exista. O relacionamento entre um tipo de entidade fraca e um tipo de entidade forte (entidade proprietária) é chamado de relacionamento identificador [Elmasri & Navathe, 2005] e geralmente as entidades fracas possuem um conjunto de atributos que apesar de não terem a força de um atributo chave, servem para identificar as entidades, esses atributos são chamados de atributos-parciais.

O modelo ER é a base para a criação do esquema de banco de dados relacional, que foi definido por Edgar Frank Codd em 1970 [Spicer, 2003]. Por se tratar de um modelo de armazenamento de dados simples e bastante formalizado, o modelo relacional é bastante usado ainda nos dias atuais. Apesar de toda sua popularidade, o modelo relacional não é suficiente para a representação fiel de algumas aplicações mais complexas, que vêm se tornando comum no cenário de desenvolvimento de software. Aplicações baseadas em orientação a objetos, por exemplo, apresentam dificuldades para representar algumas particularidades em um banco de dados relacional. Desta forma, não é possível representar apropriadamente as heranças entre classes usando simplesmente o modelo ER e o banco de dados relacional. Por conta dessa e outras limitações, do modelo ER e dos sistemas de banco de dados relacional, surge a necessidade de criação de um modelo mais completo para modelagem conceitual e um modelo mais aropriado para os sistemas de armazenamento de dados, o modelo ER-Estendido (ERE) e o sistema de bancos de dados orientados a objetos, que serão abordados na próxima seção.

2.3. Modelo ERE e Banco de Dados Orientados a Objetos

Apesar de apresentar grande força no cenário de desenvolvimento de banco de dados, o modelo ER não supre todas as necessidades de modelagem apresentadas por aplicações mais complexas e que necessitem de restrições que representem os dados de forma mais precisa. Diante disso, no final dos anos 70 e inicio dos anos 80, por conta das grandes inovações de hardware, o paradigma de desenvolvimento de software mudou e os sistemas relacionais do momento tornaram-se limitados para a manipulação de dados mais complexos, por exemplo, informações multimídia, volumes maiores de dados estruturados e conseqüentemente transações mais longas [Boscarioli *et. al.* 2006].

Como os sistemas de banco de dados tornaram-se mais complexos, o modo como a modelagem desses dados passou a ser feita, em bases conceituais, também passou por mudanças. Um dos modelos que surgiu na época para a modelagem conceitual desse novo paradigma foi a extensão do modelo ER, surgindo assim o modelo Entidade-Relacionamento **Estendido** (ou Modelo ERE), que além das características existentes no modelo ER, apresenta suporte para novas características que surgiram com o advento da evolução dos sistemas de banco de dados, como: herança, polimorfismo e encapsulamento.

As entidades, conceito bastante usado no modelo ER, passam a ser melhor caracterizadas no modelo ER Estendido como **classes**, por possuir a característica de herança. Uma classe dentro do modelo ER Estendido pode possuir generalizações (superclasses) e/ou especializações (subclasses) e associado ao conceito de herança, está o conceito de herança de atributo e de relacionamentos, em que estes são herdados durante o processo de generalização ou especialização. Na Figura 2.2 tem-se um exemplo simples de um modelo ER Estendido mostrando os relacionamentos entre superclasses e subclasses.

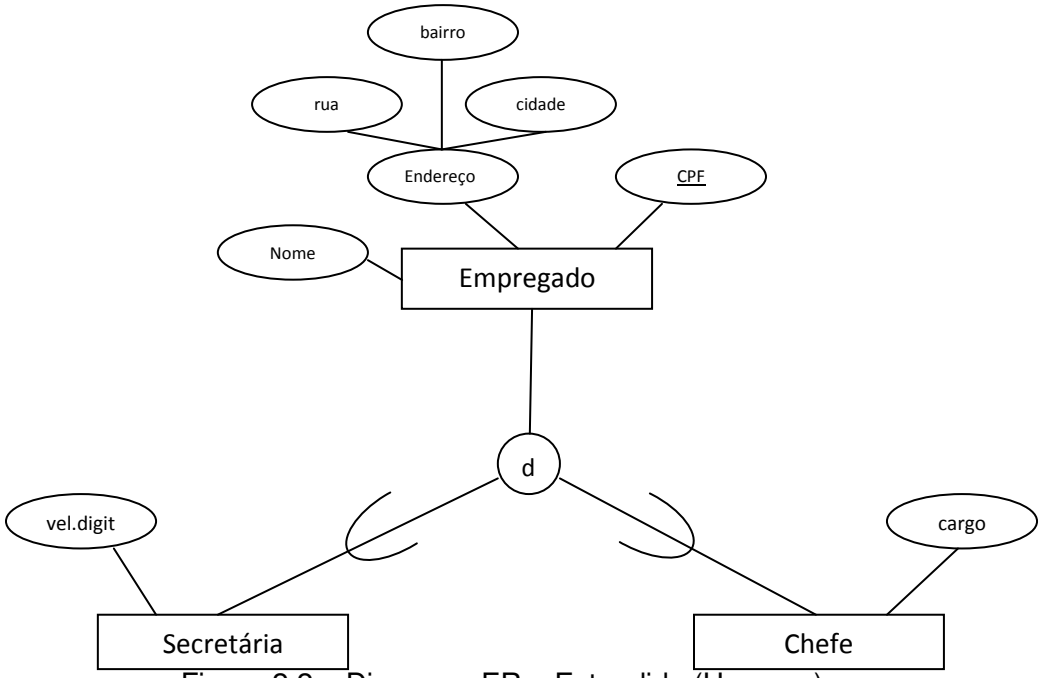
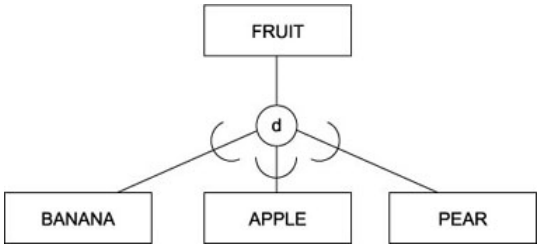


Figura 2.2 – Diagrama ER – Estendido (Herança)

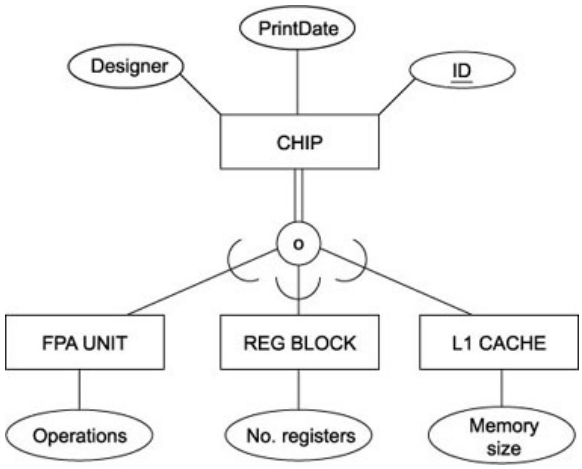
No modelo ER-Estendido mostrado na Figura 2.2, a entidade ou classe Empregado é a generalização das entidades ou classes Secretária e Chefe, no qual o contrário também é verdadeiro, em que as entidades Secretária e Chefe são especializações da entidade Empregado. A ordem em que esse relacionamento é estabelecido depende da maneira pela qual as classes foram originadas. Se a classe foi gerada *bottom-up* (de baixo para cima), então ocorreu uma generalização das subclasses, caso contrário, se foi uma geração *top-down* (de cima para baixo), então ocorreu uma especialização da superclasse.

As subclasses podem ser definidas de duas maneiras: definida por atributo, quando um atributo da superclasse determina a que subclasse a entidade pertence, ou definida pelo usuário, dessa forma nenhum atributo da superclasse caracteriza uma entidade como membro de alguma subclasse e cabe ao usuário dizer a que classe essa entidade pertence.

Na especialização, pode-se ter dois tipos de restrições que são: restrição de disjunção, que indica se a especialização é disjunta (*disjointness*) ou sobreposta (*overlapping*) e restrição de integralidade que indica se a especialização é total ou parcial. No diagrama ER-Estendido, a restrição de disjunção é representada pela letra d (disjointness), enquanto a letra o (overlapping) indica a sobreposição de instâncias. Essa representação é colocada dentro do círculo localizado na ligação entre a superclasse e as subclasses, como pode ser visto na Figura 2.3. Já a restrição de integralidade é representada através da linha que une a superclasse às subclasses, podendo ser dupla no caso da especialização total ou simples no caso da parcial, como também exemplificado na Figura 2.3.



Fonte: http://www.lightenna.com/book/export/s5/106



Fonte: http://www.lightenna.com/book/export/s5/106

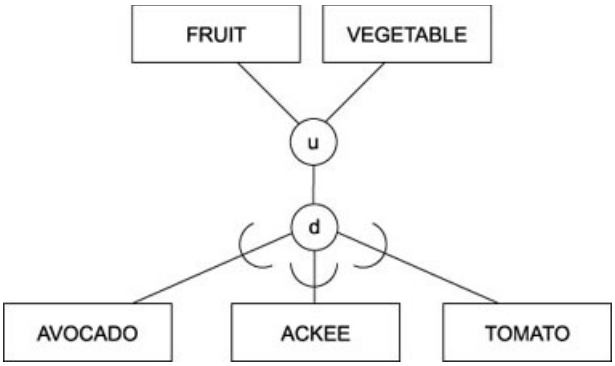
- Disjunção parcial
- Sobreposição total

Figura 2.3 – Exemplos de restrição de disjunção e restrição de integralidade

Quando existe apenas uma classe especializada (uma subclasse) não se usa a notação circulo; quando se tem mais que uma subclasse para uma superclasse, usa-se a notação circulo para indicar que tipo de restrição de disjunção está presente.

Assim como uma superclasse pode ter várias subclasses, uma subclasse também pode ter várias superclasses, ou seja, o modelo ER-Estendido dá suporte a heranças múltiplas. A característica de ter heranças múltiplas faz com que o modelo algumas vezes entre em conflito com algumas linguagens de programação que não dão suporte a esse tipo de herança. Nesses casos, artifícios de programação precisam ser aplicados. Quando se tem heranças múltiplas em um modelo ER-Estendido, isto é caracterizado como especialização de reticulado. Quando não há presença de heranças múltiplas, o modelo é então, caracterizado como especialização hierárquica [Elmasri & Navathe, 2008].

Quando ocorre a herança múltipla, uma classe é generalizada em mais de uma superclasse, herdando assim os atributos e relacionamentos de todas as suas superclasses. Existe ainda a possibilidade de poder escolher dentre as superclasses associadas a uma classe, qual delas será de fato a generalização de uma instância da classe, neste caso, a subclasse é chamada de tipo união ou categoria e é representada no modelo ER-Estendido pela letra u dentro do circulo que liga as superclasses à subclasse, como mostrado na Figura 2.4. A diferença entre uma subclasse com várias superclasses e uma subclasse do tipo união é que no tipo união, apenas os atributos e relacionamentos de uma das superclasses serão herdados pela subclasse o que não ocorre quando a herança não é do tipo união e a subclasse tem que herdar todos os atributos e relacionamentos de todas as suas superclasses.



Fonte: http://www.lightenna.com/book/export/s5/106

Figura 2.4 – herança com subclasse tipo união

Com o uso do modelo ER-Estendido, fica bem mais simples para o projetista gerar então o modelo lógico e conseqüentemente o modelo físico de um banco de dados mais complexo, como é o caso do Banco de Dados Orientado a Objetos. Nele, entidades e atributos mais complexos podem ser representados, como por exemplo, vídeos e dados com estrutura complexa, o que era difícil em bancos de dados puramente relacionais. Outra diferença dos bancos de dados orientados a objetos para os bancos relacionais é que a noção de atributo chave (chave primária) perde um pouco o valor, pois, cada objeto num banco de dados orientado a objetos possui agora um identificador único que o identifica dentre os objetos existentes no banco.

Na Figura 2.5, pode-se ver um exemplo de um modelo Entidade-Relacionamento Estendido aplicado a um sistema de informações multimídia de uma galeria de arte. Esse modelo será melhor detalhado no capitulo 4 deste trabalho, como um estudo de caso utilizando a ferramenta ERE-CASE. A modelagem dada a seguir foi feita usando-se a ferramenta Microsoft Visio 2007 [Visio, 2006].

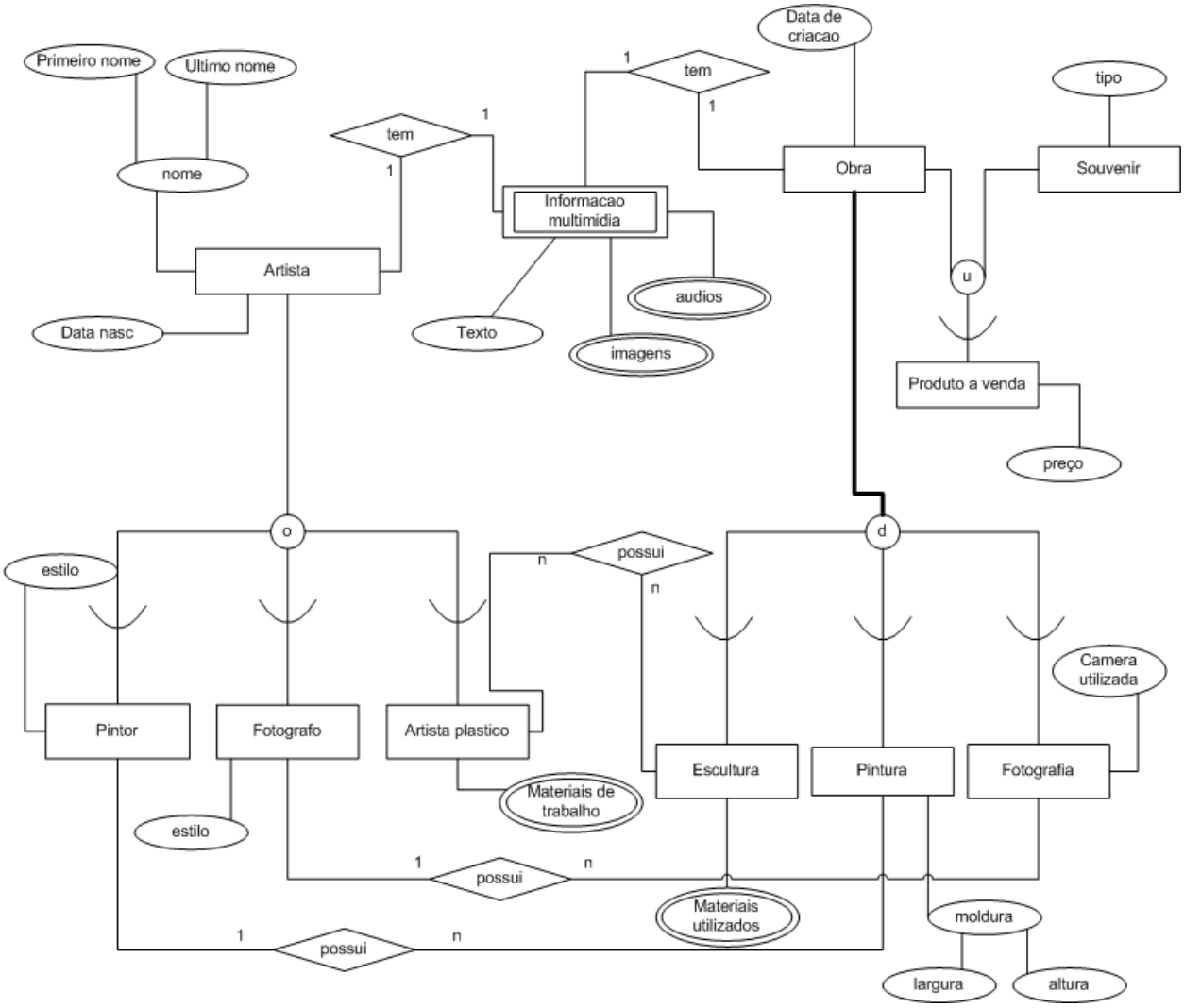


Figura 2.5 – Modelo ERE de uma galeria de arte multimídia

O mapeamento entre o modelo Entidade Relacionamento Estendido e o modelo lógico não será abordado neste trabalho, por fugir do escopo da aplicação ERE-CASE.

3. A Ferramenta ERE-CASE

ERE-CASE é uma ferramenta que tem como principal objetivo modelar conceitualmente o modelo ER-Estendido, podendo também ser utilizada para a modelagem conceitual do modelo ER. Apesar de ter o intuito de apenas modelagem conceitual, a ferramenta foi desenvolvida para permitir evoluções, pensando-se na possibilidade de extensão para uma ferramenta mais completa que além da modelagem conceitual, possa gerar os modelos lógicos e físicos para o banco de dados, por exemplo.

Dentre as ferramentas concorrentes, estudadas na Seção 1.3 deste trabalho, pode-se observar que apenas as ferramentas Umbrello [Umbrello, 2008] e Sybase Power Design [Sybase, 2008] possuem a funcionalidade de trabalhar com o modelo ER-Estendido, e mesmo assim, a ferramenta Umbrello hoje só está disponível para o sistema operacional Linux, o que restringe bastante o uso e conhecimento da mesma, enquanto a Sybase Power Design é bastante cara, sendo apenas viável para grandes empresas. Como poderá ser visto nas próximas subseções, a ferramenta ERE-CASE foi desenvolvida em Java [Java, 2008], logo a portabilidade entre sistemas operacionais estará garantida e o seu código é aberto e de livre acesso.

Será mostrado neste capítulo, detalhes de funcionalidades e implementação da ferramenta ERE-CASE, para que no próximo capítulo, um estudo de caso seja feito mostrando a utilização das funcionalidades apresentadas. A ferramenta ERE-CASE está disponível para usuários que queiram testá-la ou usá-la em seus projetos num repositório publico no endereço https://eercase.googlecode.com.

3.1. Funcionalidades

As funcionalidades apresentadas pela ferramenta ERE-CASE serão melhor detalhadas quando um caso de uso da construção de um modelo ER-Estendido for apresentado no próximo capítulo. Neste capítulo, será apresentada uma definição geral destas funcionalidades.

3.1.1. Abrir e Salvar modelo em formato XML

A ferramenta ERE-CASE permite que o usuário salve o modelo que estiver construindo e o resgate posteriormente abrindo o arquivo salvo. O uso de XML [XML, 2008] para armazenamento do modelo baseia-se na característica de extensão presente na linguagem XML .

3.1.2. Modelagem Conceitual no modelo Entidade-Relacionamento Estendido

Por existir poucas ferramentas que tenham essa funcionalidade, todas as que foram encontradas com esta característica ou não funcionam em todos os sistemas operacionais ou são extremamente caras. Este é considerado um diferencial da ferramenta ER-CASE. O modelo ER-Estendido é pouco abordado pelas ferramentas CASE existentes no mercado, mesmo tendo grande importância e participação nos projetos de modelagem de bancos de dados complexos.

3.1.3. Edição dos elementos do modelo

A edição dos elementos do modelo na ferramenta ERE-CASE acontece de forma fácil e intuitiva. Através da barra de ferramentas no canto esquerdo da tela principal da aplicação (I), Figura 3.9, o usuário pode inserir, selecionar ou remover elementos do modelo. Ao selecionar um determinado elemento, todas as informações referentes ao mesmo serão apresentadas em uma tabela presente no lado direito da interface da aplicação (IV). Estas informações podem ser editadas pelo usuário nesta mesma tabela.

3.1.4. Geração automática de scripts para Banco de Dados

Após ser gerado o modelo conceitual ER ou ER-Estendido, o usuário tem a opção de gerar automaticamente as tabelas para o banco de dados Oracle. Esta funcionalidade, entretanto, necessita da ferramenta ERE2Oracle[Dantas, 2008].

3.2. Especificação e Implementação

Como já foi citado na introdução deste capítulo, a ferramenta ERE-CASE foi completamente desenvolvida na linguagem de programação Java. Sendo assim, a ferramenta é considerada como independente de plataforma, pois funciona sobre uma máquina virtual, a Java Virtual Machine (JVM) [Java, 2008], e não se comunica diretamente com o sistema operacional, deixando este trabalho de portabilidade a cargo da JVM. Para a codificação da ferramenta foi usada a IDE (Integrated Development Environment – Ambiente de Desenvolvimento Integrado) *open-source (código aberto)*, Eclipse [Eclipse, 2008].

Tendo o problema de portabilidade sido resolvido, pôde-se concentrar no desenvolvimento em si da ferramenta. ERE-CASE obedece a uma combinação de padrões de projeto que foram escolhidos após pesquisa dentre os principais padrões existentes para desenvolvimento de aplicações. A escolha destes padrões foi baseada em exemplos conceituais e existentes no mercado que apontaram para surgimento de facilidades no momento da implementação, deixando os códigos produzidos mais flexíveis e organizados para futuras mudanças e evoluções. A seguir, na Subseção 3.2.2, podem ser vistos os padrões de desenvolvimento adotados para a ferramenta ERE-CASE e os principais motivos para a escolha dos mesmos.

Como todo bom projeto de software, a ferramenta adotou a alguns processos da engenharia de software em seu planejamento e na construção de diagramas para facilitação no entendimento e codificação da mesma, como pode ser visto na Subseção 3.2.1. Algumas tecnologias, como: XML [XML, 2008] e Java Swing [Swing, 2008] foram também utilizadas no desenvolvimento da aplicação.

3.2.1. Diagramas

Os diagramas de classe da aplicação ERE-CASE foram modelados usando a ferramenta open-source StarUml [StarUml, 2008]. Eles foram desenvolvidos para documentar a construção da ferramenta ERE-CASE e tornar fácil a sua extensão.

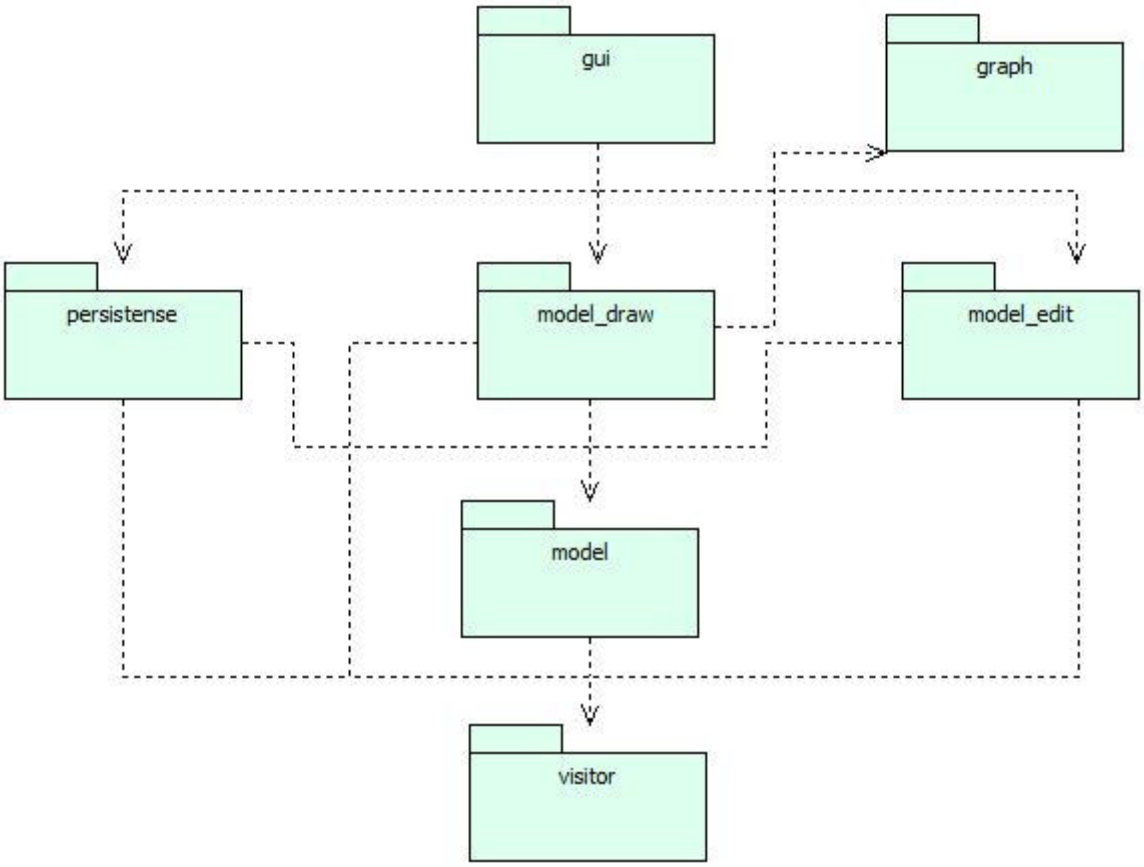


Figura 3.1 Diagrama de Pacotes

O sistema ERE-CASE apresenta a organização de seus pacotes baseada nas responsabilidades de cada conjunto de classes, como pode ser visto na Figura 3.1. O pacote *gui* apresenta as classes que têm a responsabilidade de interface entre o usuário e o sistema. Os pacotes *model_edit* e *model_draw* são responsáveis por edição e renderização, respectivamente, do modelo ER-Estendido que está representado em suas classes base do pacote *model*. O pacote *persistence* apresenta as classes com a responsabilidade de importar

e exportar o modelo, tornando-o persistente. O pacote *visitor* apresenta as interfaces responsáveis por definir como o padrão Visitor (ver seção 3.2.2) será implementado no sistema. Nos detalhamentos dos pacotes abaixo, da Figura 3.2 à figura 3.6, pode-se ver em vários deles as interfaces *VisitorERR* e *VisitableERR*, estas classes serão melhor explicadas na seção 3.2.2.

O pacote *model*, Figura 3.2, é o mais importante pacote do sistema por possuir as classes que representam os elementos do modelo ER-Estendido. Os elementos Entidade, Relação e Atributo, que formam a base do modelo ER-Estendido, são representados no sistema ERE-CASE pelas respectivas classes: *Entity*, *Relation* e *Attribute*. Complementando o modelo, tem-se a classe *SetInheritance* que representa um conjunto de heranças de uma entidade e a classe Link que representa as ligações entre um relacionamento ou herança e uma entidade.

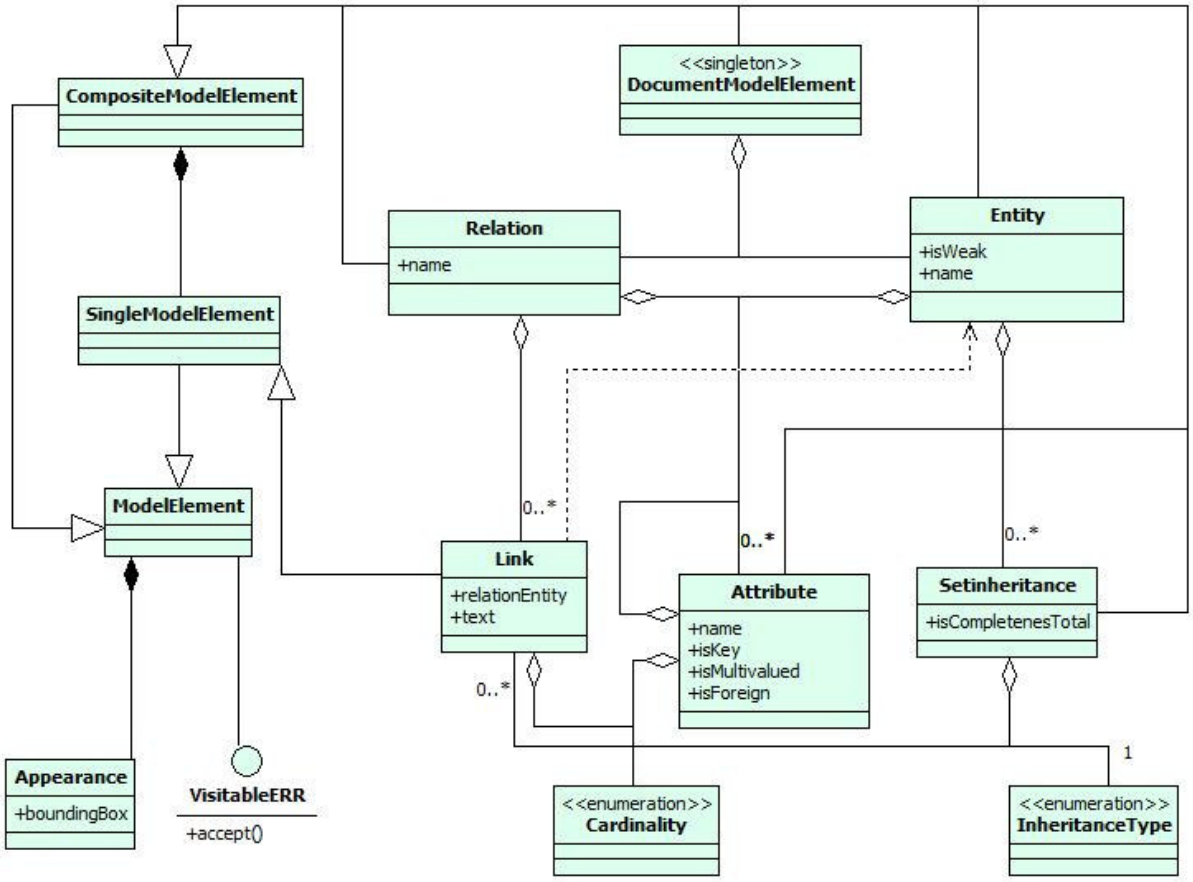


Figura 3.2 – Pacote model (diagrama de classes)

Em linhas gerais, pode-se interpretar o diagrama da seguinte forma:

- O modelo é representado pela classe *DocumentModelElement* que possui um conjunto de relacionamentos (*Relation*) e um conjunto de Entidades (*Entity*);

- Cada entidade (*Entity*) pode possuir um conjunto de atributos (Attribute) e um conjunto de heranças (*SetInheritance*),que representa a relação de herança entre ela e suas subclasses e superclasses através de ligações (*Links*);

- Cada relação (*Relation*) também pode possuir um conjunto de atributos (*Attribute*) e um conjunto de links (*Link*) para cada entidade relacionada; e

- Cada atributo (*Attribute*) pode ainda possuir um conjunto de atributos, quando este é um atributo composto.

Os elementos do diagrama *model* obedecem ao padrão de projetos *Composite*, o qual será melhor detalhado na seção 3.2.2.

A persistência do pacote *model* é de responsabilidade do pacote *persistence*, detalhado na Figura 3.3. Este pacote tem as responsabilidades de exportar, com a classe *XMLExporter*, e importar, com a classe *XMLImporter*, o modelo ER-Estendido, tornando-o persistente a várias execuções da ferramenta. O formato do arquivo trabalhado na persistência é XML, por possuir características de extensão (ver seção 3.2.3).

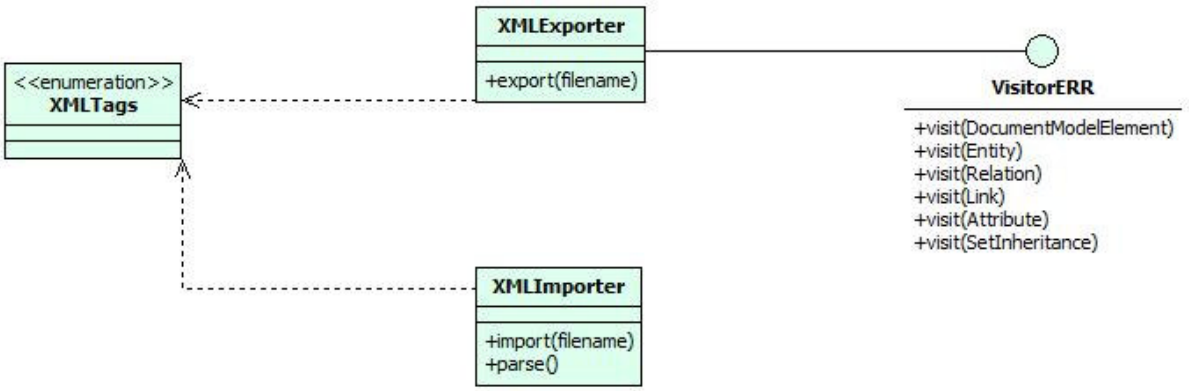


Figura 3.3 – Pacote *persistence* (diagrama de classes)

A parte gráfica de exibição do modelo construído é de responsabilidade do pacote *model_draw*, detalhado na Figura 3.4. O pacote possui a classe *EERGraph* como classe principal que tem a responsabilidade de delegar as funcionalidades de desenho do modelo na tela para o pacote *graph*, Figura 3.5.

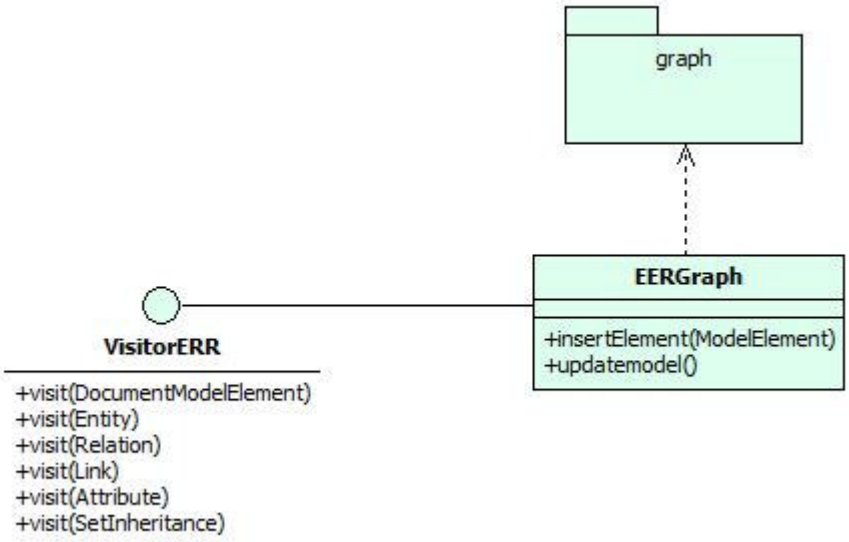


Figura 3.4 – Pacote *model_draw* (diagrama de classes)

Para representar graficamente o modelo EER foi usado o componente de renderização de grafos JGraph [jGraph, 2008]. Algumas novas classes foram definidas dentro do pacote *graph,* como pode ser visto na Figura 3.5, para adaptação dos elementos disponibilizados pelo componente *JGraph* ao contexto desejado pela ferramenta ERE-CASE. As classes que representam o modelo ER-Estendido apresentadas no pacote *model*, Figura 3.2, foram mapeados em classes do pacote *graph*. O Quadro 3.1 mostra como esse mapeamento foi realizado.

Quadro 3.1 – Mapeamento entre classes do pacote *model* e pacote *graph*

Pacote *model*	**Pacote *graph***	**Observações**
DocummentModelElement	*EERGraphModel*	Representação geral do modelo Entidade Relacionamento Estendido
Entity	*EntityVertex*	Representação das entidades
Relation	*RelationVertex*	Representação dos relacionamentos
Attribute	*AttributeVertex*	Representação dos atributos
SetInheritance	*SetInheritanceVertex*	Representação das heranças
Link	*LinkEdge*	Representação das conexões

Após o mepeamento das classes presentes no pacote *model* para as classes no pacote *graph*, formou-se o diagrama visto na Figura 3.5.

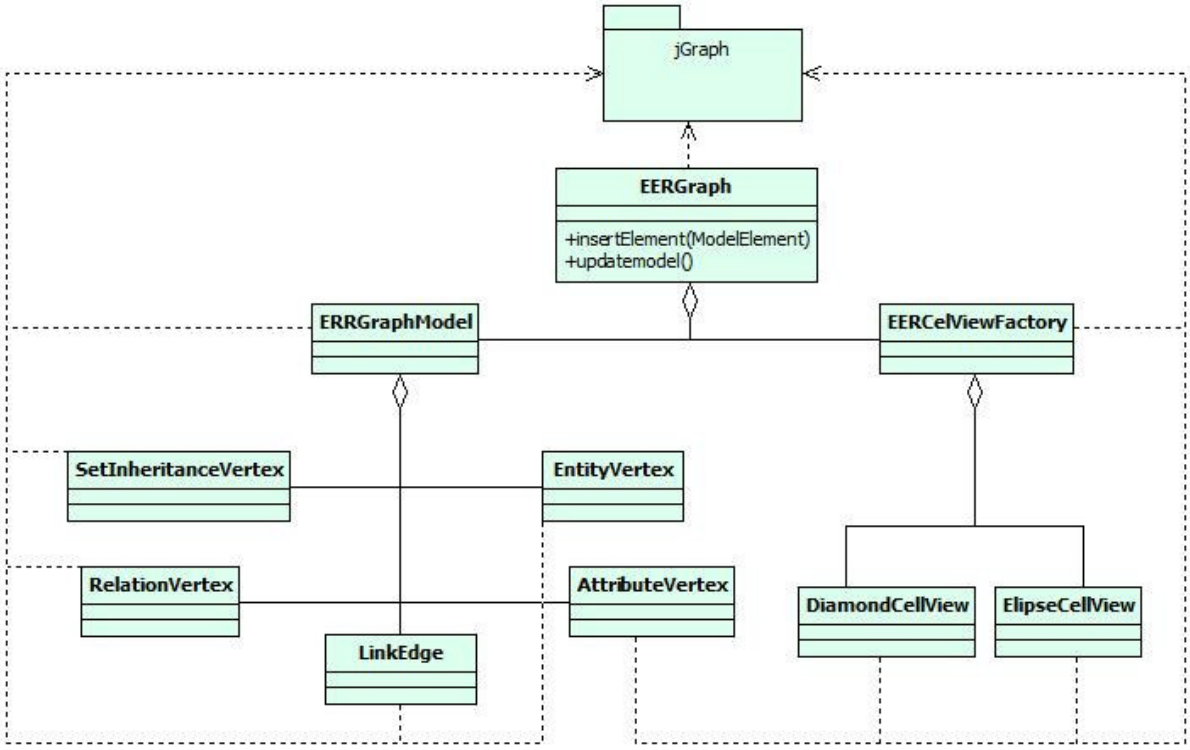


Figura 3.5 – Pacote *graph* (diagrama de classes)

A edição do modelo é de responsabilidade em parte pelo pacote *model_draw (*Figura 3.5*),* que se responsabiliza pela edição dos elementos já existentes utilizando-se do componente JGraph [JGraph, 2008] e do pacote *model_edit*, detalhado na Figura 3.6, que tem as responsabilidades de inserção e remoção de elementos. Para cada uma dessas responsabilidades do pacote *model_edit*, foi criada uma classe respectivamente, *ModelWrite* e *ModelErase*.

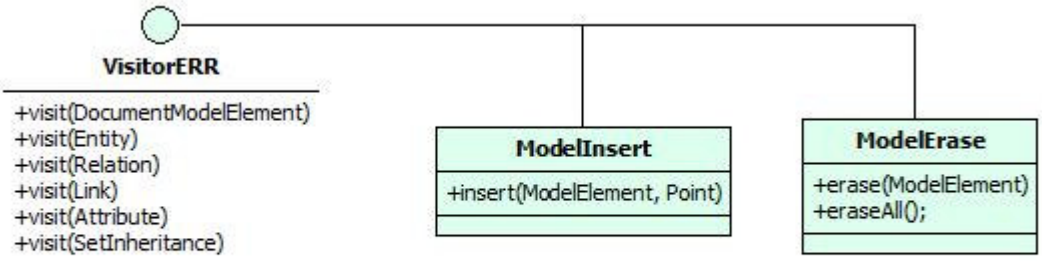


Figura 3.6 – Pacote *model_edit* (diagrama de classes)

Por fim, fazer a interface entre o usuário e a aplicação é uma responsabilidade do pacote *gui* (graphical user interface – interface gráfica com o usuário), detalhado na Figura 3.7. A classe *MainFrame* engloba todos os componentes gráficos da interface que são: menu (*EERMenuBar*), toolbars (*EERFileToolBar e EER*), painéis de edição dos elementos (*AttributesPanel, InheritancesPanel, entitiesPanel, LinksPanel, RelationsPanel*) e o grafo representando o modelo (*EERGraph*) .

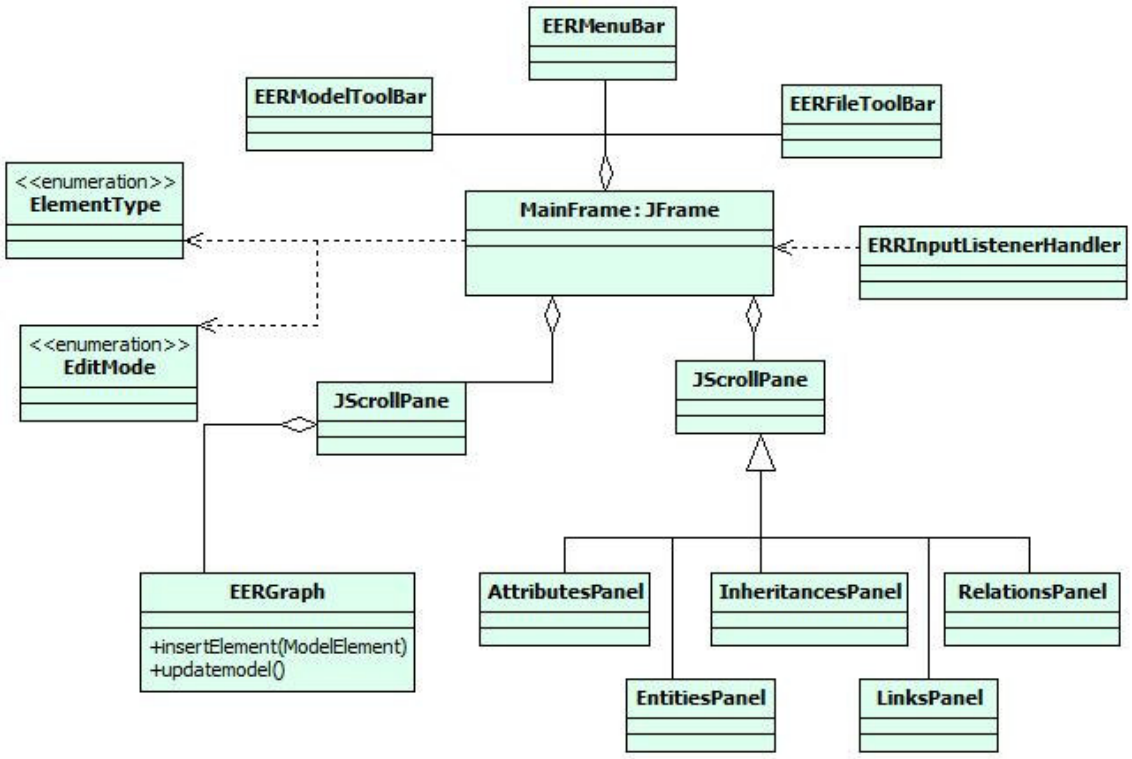


Figura 3.7 – Pacote *gui* (diagrama de classes)

3.2.2. Padrões de Projeto

Os padrões de projeto utilizados para o desenvolvimento da ferramenta foram: Composite [Cooper, 2000], Singleton [Cooper, 2000] e Visitor [Martin, 2002].

No desenvolvimento de software em que existe uma organização de suas classes em formato de árvore, o padrão Composite é indicado por conseguir compor a estrutura árvore de forma organizada. Nesse padrão, um objeto pode ser de dois tipos: um objeto folha (*single object*) ou um objeto nó (*composite object*). O objeto nó é então composto de outros objetos nós ou objetos folha, formando assim uma estrutura de árvore para os objetos da aplicação. Na ferramenta ERE-CASE este padrão foi aplicado às classes básicas do sistema que estão localizadas no pacote *model* (ver seção 3.2.1). A classe *CompositeModelElement* foi criada para representar os objetos nós e a classe *SingleModelElement* para os objetos folha. Desta forma, elementos que podem ter filhos como é o caso de: Documento (*Document*), Entidades (*Entity*), Relações (*Relation*), Atributos (*Attribute*) herdam as características da classe *CompositeModelElement* e objetos como ligações entre entidades (*Link*) e novamente atributos (*Attribute*) herdam as características da classe *SingleModelElement*.

O padrão Singleton, preza por restringir uma classe a ter apenas uma instância no sistema em execução [Cooper, 2000]. Na ferramenta ERE-CASE este padrão pode ser observado na classe *DocumentModelElement*, que representa uma instância do modelo ER-Estendido ativa no momento. Como a aplicação permite que apenas um modelo seja manipulado por vez, a classe que representa o modelo em si é representada como um objeto Singleton.

Nos pacotes que representam o controle e funcionamento das regras de negócio da ferramenta ERE-CASE, tem-se o padrão *Visitor* como base para a execução. Das responsabilidades apresentadas pela ferramenta, as principais que se pode destacar são: edição, desenho dos diagramas e persistência dos dados. Todas estas responsabilidades usam e modificam um modelo bem definido no pacote *model* (ver seção 3.2.1). Como o modelo já está bem definido, e existe uma separação entre a sua manipulação e a sua definição na ferramenta, o padrão Visitor foi o escolhido para modelar a implementação do funcionamento da ferramenta, pois, como definição deste padrão tem-se a separação entre os algoritmos de manipulação e a estruturação dos objetos [Martin, 2002]. O padrão é representado na ferramenta por duas interfaces, a interface *VisitorERR* e *VisitableERR*, que ditam como será a implementação do mesmo dentro do sistema. Toda classe que for visitada, ou seja, todo objeto do modelo (Entidades, Relações, Atributos e Links), implementa a interface *VisitableERR* e aceitará “visitas” das classes que implementam a interface *VisitorERR*. Estas visitas feitas por classes como *ModelRenderer*, *ModelEdit* e XMLExporter, aos elementos do modelo servirão para percorrer a estrutura de árvores montadas com o auxílio do padrão *Composite* [Cooper, 2000] e dependendo de qual *Visitor* estiver fazendo a “visita”, as devidas operações serão aplicadas. Outro benefício trazido pela implantação deste padrão de projetos na ferramenta

ERE-CASE é a facilidade de extensão. Sempre que novas características sejam necessárias, não será preciso mexer na estrutura básica do modelo e apenas uma nova classe *Visitor* precisará ser implantada no sistema para que novas "visitas" sejam feitas ao modelo já sedimentado.

3.2.3. Persistência

Como em todo software de modelagem, existe a necessidade da ferramenta ERE-CASE salvar e abrir os modelos construídos sempre que necessário. O sistema de persistência implementado na ferramenta ERE-CASE utiliza XML (eXtensible Markup Language) [XML, 2008]. A escolha pela utilização de XML como arquivo de persistência da ferramenta foi feita por conta da característica de extensão presente nessa linguagem, já mencionado anteriormente. Como existe o projeto de ampliação das funcionalidades desta ferramenta, é importante que esta característica de extensão seja levada em consideração para o desenvolvimento da mesma.

Basicamente, o arquivo XML manipulado pela ferramenta ERE-CASE é a representação do pacote *model* de forma estruturada de acordo com elementos e atributos de um arquivo XML. No apêndice A pode-se observar a representação do XML schema [XML, 2008] correspondente aos arquivos XML que podem ser gerados pela ferramenta. Esta representação foi gerada com a utilização da ferramenta Oxygen [Oxygen, 2008].

3.2.4. Interface

A comunicação entre a ferramenta ERE-CASE e o usuário da mesma é feita basicamente por meio de uma tela, que é a tela principal do sistema. A Figura 3.7 mostra o projeto inicial da interface com o usuário, construída utilizando-se a ferramenta Microsoft Visio 2007 [Visio, 2006].

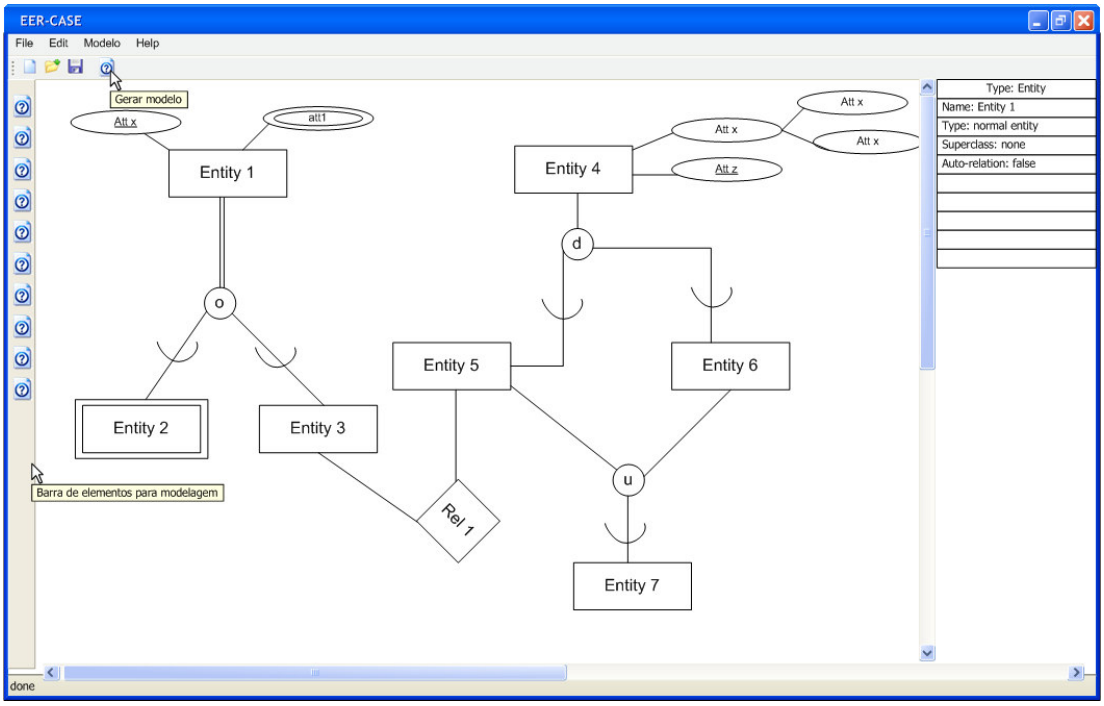


Figura 3.8 – Projeto de interface

A interface do sistema foi fortemente baseada nas interfaces de ferramentas CASE estudadas e descritas na Seção 1.3. A interface final da ferramenta, após implementação, está representada na Figura 3.8. Foi utilizado Java Swing [Swing, 2008] juntamente com a ferramenta Eclipse para a construção da mesma.

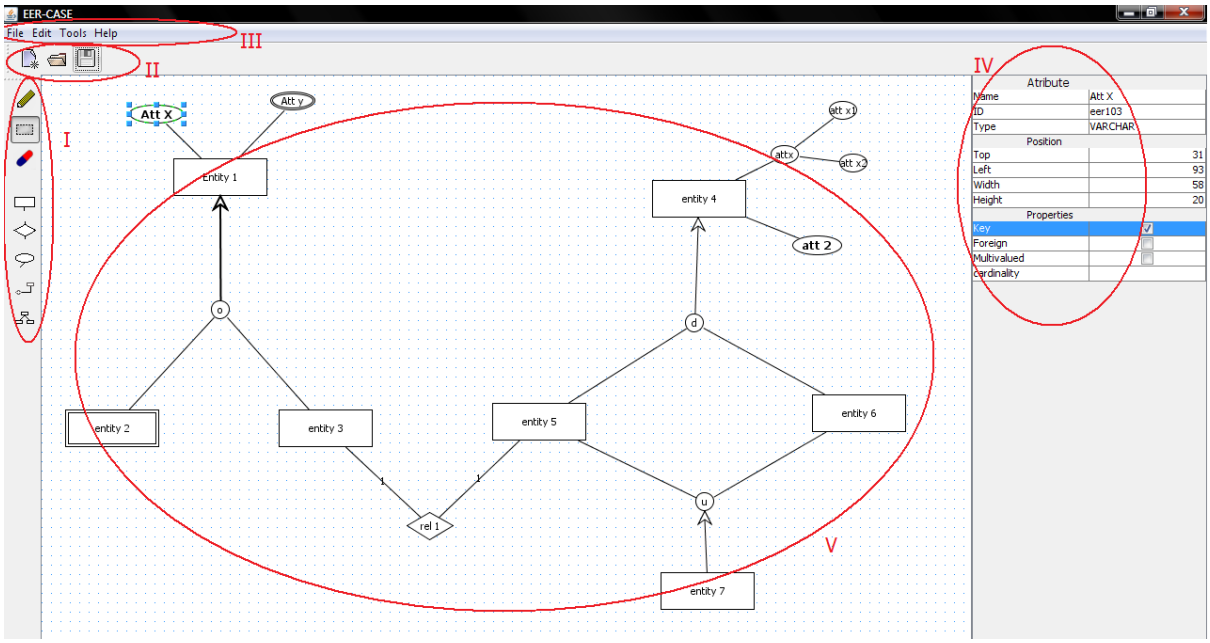


Figura 3.9 – Interface final da ferramenta ERE-CASE

Os elementos que compõem a interface principal da ferramenta são as seguintes: (I) Barra de ferramentas do modelo, que é onde estão localizados os itens de edição do modelo (inserir, editar, apagar) e os itens que representam os elementos do modelo (entidade, relacionamento, atributo, conexão e herança); (II) Barra de ferramentas de arquivo, que é onde estão localizados os itens relativos a operações com arquivo (novo, abrir, salvar); (III) Menu principal, que representa o menu geral da ferramenta; (IV) Tabela de edição dos elementos, que é onde as características de cada elemento são editadas; e (V) Área de desenho do modelo, onde o modelo é apresentado ao usuário.

4. Um estudo de caso

Para demonstrar o uso da ferramenta ERE-CASE em um caso de uso real, foi escolhido modelar o banco de dados de um sistema de informação de uma galeria de artes multimídia. Esta escolha se deu por conta do modelo de dados correspondente apresentar todos os conceitos descritos no capítulo 2 desta monografia, além das características essenciais de utilização de um banco de dados Orientado a Objetos. Com esse modelo, será possível também ver todas as funcionalidades da ferramenta ERE-CASE.

4.1. Minimundo geral do modelo

O sistema de informação da galeria de artes contém dados sobre todas as obras cadastradas na galeria, seus respectivos artistas e produtos vendidos na mesma. A seguir encontra-se a identificação e as características das entidades que fazem parte do minimundo da galeria:

- Obras de Arte – As obra de arte são os atrativos principais dentro da galeria, podendo algumas ser apenas objetos de exposição ou outras também podendo ser produtos de venda da galeria. Toda obra tem sua data de criação, e informações multimídia sobre a obra. As obras de arte podem ser de três tipos dentro desta galeria: pintura, fotografia e escultura, que além, das características comuns a toda obra de arte, possuem características especificas. Cada pintura possui informação sobre dimensões da pintura e seu pintor, cada fotografia possui informações sobre a máquina fotográfica usada e seu respectivo fotógrafo e cada escultura possui informações sobre os materiais usados para a sua construção e os artistas plásticos que trabalharam na obra;

- Artistas – Os artistas também têm destaque dentro do sistema da galeria, podendo, estes, ser consultados nos terminais de cada obra de arte que o mesmo for autor. As informações disponíveis sobre cada artista são: nome, data de nascimento e suas informações multimídia. Como no caso das obras de arte, os artistas podem ser de três categorias: Artistas plásticos, Pintores e Fotógrafos. Cada artista plástico tem uma coleção de materiais que o mesmo utiliza para suas obras e cada pintor e fotógrafo possui um estilo associado;

- *Souvenir* – A galeria, além de vender algumas de suas obras expostas, também vende *souvenires* relacionados com as obras existentes e relacionados com a própria marca da galeria. Cada souvenir tem um tipo associado e um preço; e

- Informação Multimídia – As informações multimídia que a galeria apresenta possuem as seguintes mídias: um texto explicativo, um conjunto de áudios explicativos e um conjunto de imagens.

4.2. Modelo Conceitual

Nesta seção, será mostrado o passo a passo da construção do modelo conceitual Entidade-Relacionamento Estendido para o minimundo apresentado na Seção 4.1. As funcionalidades da ferramenta ERE-CASE serão vistas na prática, auxiliando desde um usuário casual até um mais experiente, a entender a dinâmica de funcionamento da ferramenta.

As entidades identificadas no sistema foram: Artista, Pintor, Fotógrafo, Artista Plástico, Obra, Escultura, Pintura, Fotografia, Souvenir, Produto a Venda e Informação Multimídia. Na Figura 4.1, têm-se essas entidades inseridas na ferramenta ERE-CASE simplesmente clicando no item "*add Entity*" na barra de ferramentas lateral e inserindo na posição desejada.

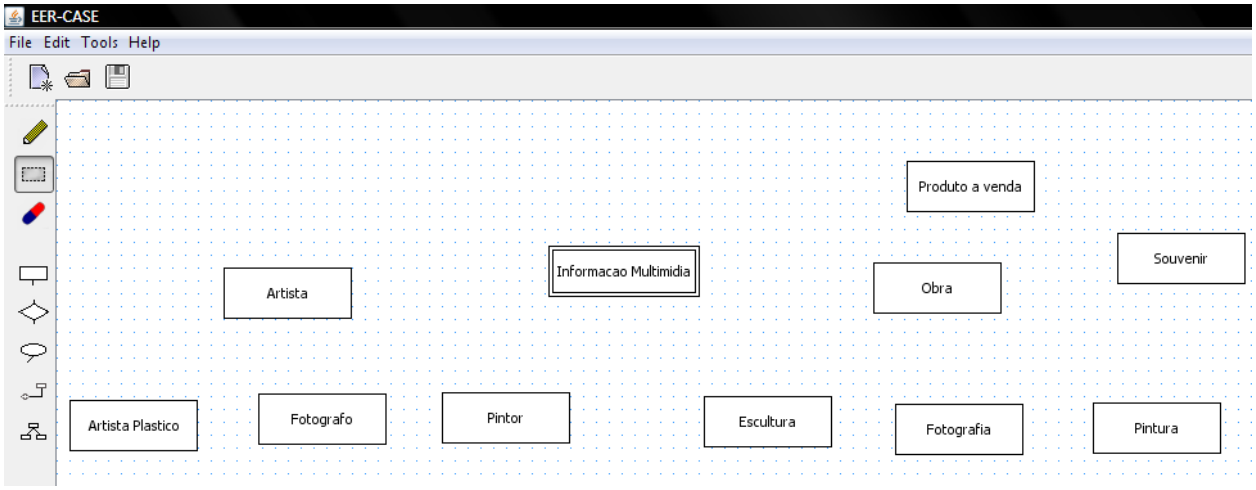


Figura 4.1 – Inserindo Entidades

As características de cada entidade podem ser editadas usando-se a tabela de edição de Entidades, Figura 4.2, que fica visível clicando-se na entidade que se deseja editar.

Entity	
Name	Informacao Multimidia
ID	eer11
Position	
Top	113
Left	403
Width	118
Height	40
Properties	
Weak	☑

Figura 4.2 – Tabela de edição de entidades

Após inseridas as entidades, são identificadas as possibilidades de especialização e generalização no modelo. Baseando-se no minimundo, foram encontradas as seguintes heranças, ilustradas na Figura 4.3:

- O Artista é especializado nas subclasses Artista plástico, Pintor e Fotógrafo por possuir atributos em comum a essas três classes. Esta especialização é classificada como sobreposição parcial, pois, um Artista pode não ser necessariamente de uma de suas subclasses e quando este for, pode representar mais de uma de suas subclasses, por exemplo, um Artista pode ser ao mesmo tempo Pintor e Escultor;

- A Obra é especializada nas subclasses Escultura, Fotografia e Pintura por estas possuírem atributos em comum. Como uma escultura só pode ser especializada em uma de suas subclasses, por exemplo, uma pintura não pode ser uma fotografia e vice versa, e toda Obra precisa se encaixar em uma dessas subclasses, trata-se então de uma especialização de disjunção total; e

- Um Produto a Venda pode ser um Souvenir ou pode ser uma Obra de Arte, porém não pode ter como superclasse ao mesmo tempo estas duas classes, desse modo têm-se as características de uma herança do tipo união.

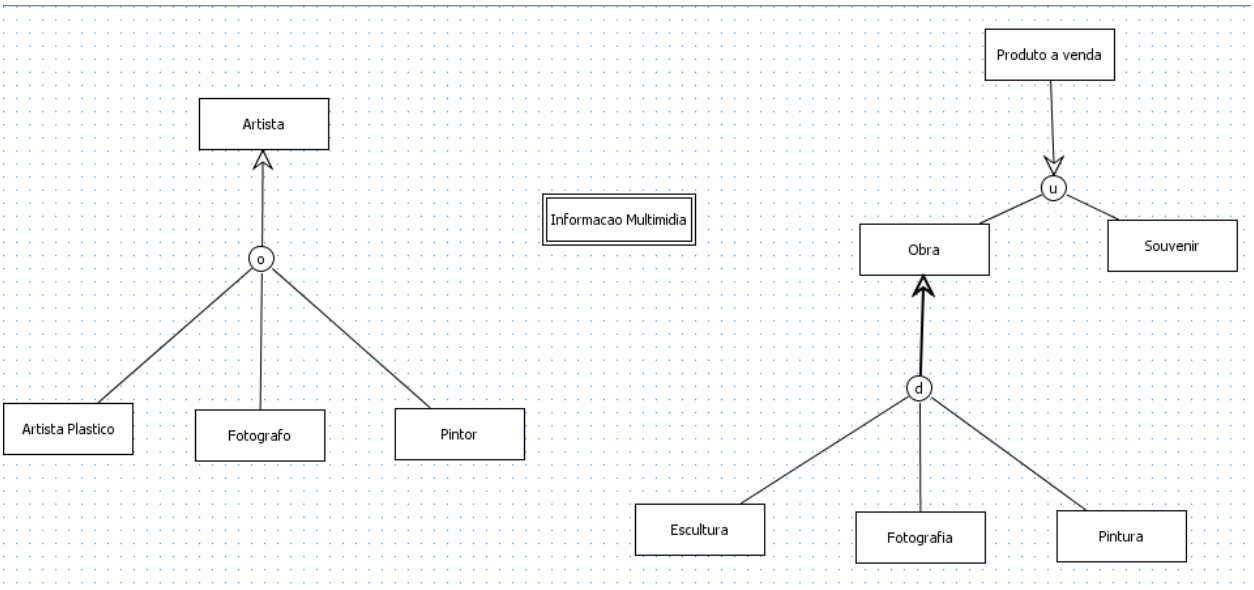


Figura 4.3 – inserindo heranças

A inserção da herança é feita clicando-se no item "*create inheritance*" na barra de ferramentas do modelo e depois clicando-se na entidade que será, a superclasse no caso das especializações de disjunção e de sobreposição e a subclasse no caso da união. A direção da seta que liga o círculo da herança à entidade indica qual é a direção da herança. O caractere dentro do circulo indica qual o tipo da herança (sobreposição – 'o', disjunção – 'd' e união – 'u'). E o estilo da linha indica se a especialização é total (linha mais escura) ou parcial (linha mais clara). As características de uma herança são editadas na tabela de edição de heranças, Figura 4.4.

Inheritance	
Text	
ID	eer50
Position	
Top	292
Left	712
Width	20
Height	20
Properties	
Total	☑
Type	Disjoint

Figura 4.4 – Tabela de edição de heranças

Depois de inseridas as entidades, são identificados os atributos que cada uma possui, seus respectivos tipos e propriedades. Para cada Entidade foram inseridos os atributos identificados no minimundo da galeria de artes e estes podem ser vistos na Figura 4.5.

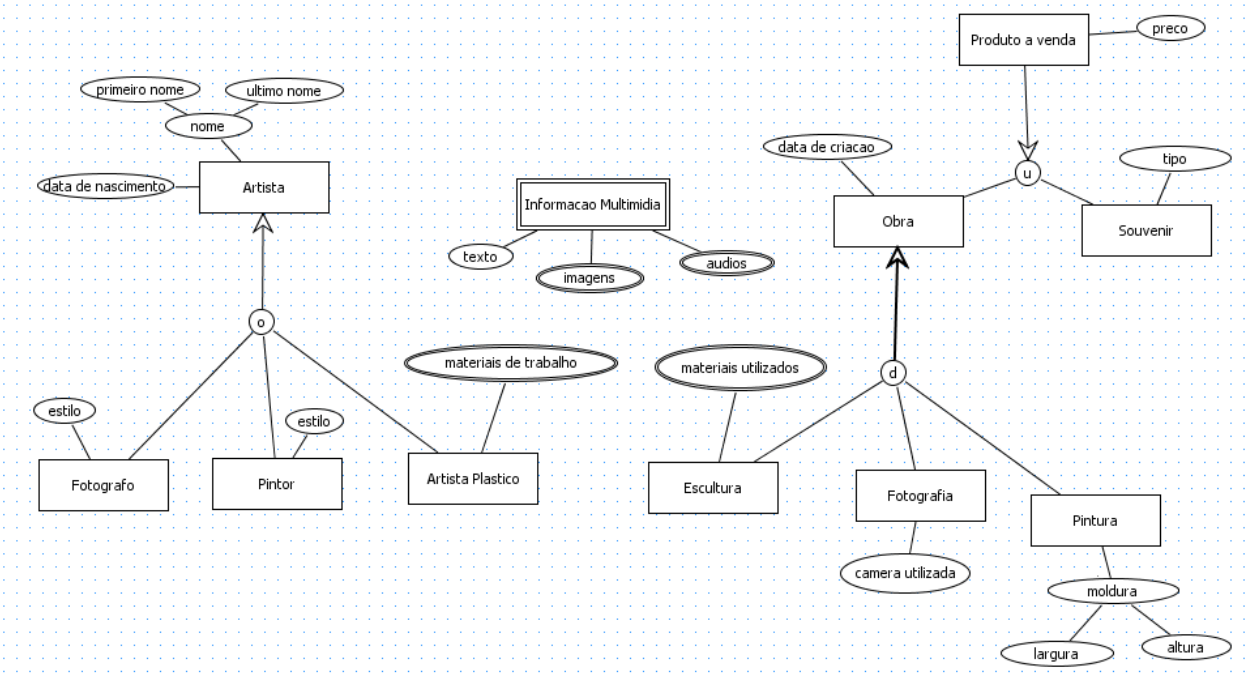


Figura 4.5 – Inserindo atributos

A inserção de atributos é realizada clicando-se no item "*add attribute*" na barra de ferramentas do modelo no lado esquerdo e clicando-se depois no elemento ao qual se deseja adicionar o atributo. Apenas os elementos Entidade, Relação e Atributo são capazes de receber atributos como "filhos". Quando um atributo é adicionado a outro atributo, este passa a se tornar um atributo composto. Atributos representados por elipses duplas são atributos multivalorados e atributos com o nome em negrito são os atributos chave. Todas as

características e propriedades dos atributos podem ser editadas na tabela de edição do lado direito da aplicação. Basta selecionar um atributo com o mouse e editar suas propriedades na tabela vista na Figura 4.6.

Atribute	
Name	Imagens
ID	eer34
Type	VARCHAR
Position	
Top	260
Left	439
Width	92
Height	31
Properties	
Key	☐
Foreign	☐
Multivalued	☑
cardinality	

Figura 4.6 – Tabela de propriedades de um atributo

A etapa final para conclusão do modelo Entidade-Relacionamento Estendido da galeria de obras de arte é a criação dos relacionamentos entre as entidades. Primeiro seleciona-se o item "*add relation*" na barra de ferramentas do modelo e insere-se o losango que representa um relacionamento em qualquer local na área de desenho do modelo, de forma semelhante à inserção de entidades. Após isso, usando-se o item "*connect elements*" também presente na barra de ferramentas do modelo, são feitas as conexões entre o losango de relacionamento e as entidades relacionadas, clicando seguidamente na entidade e no próprio losango de relacionamento. Para o modelo ERE da galeria de artes, foram criados os relacionamentos visualizados na Figura 4.7.

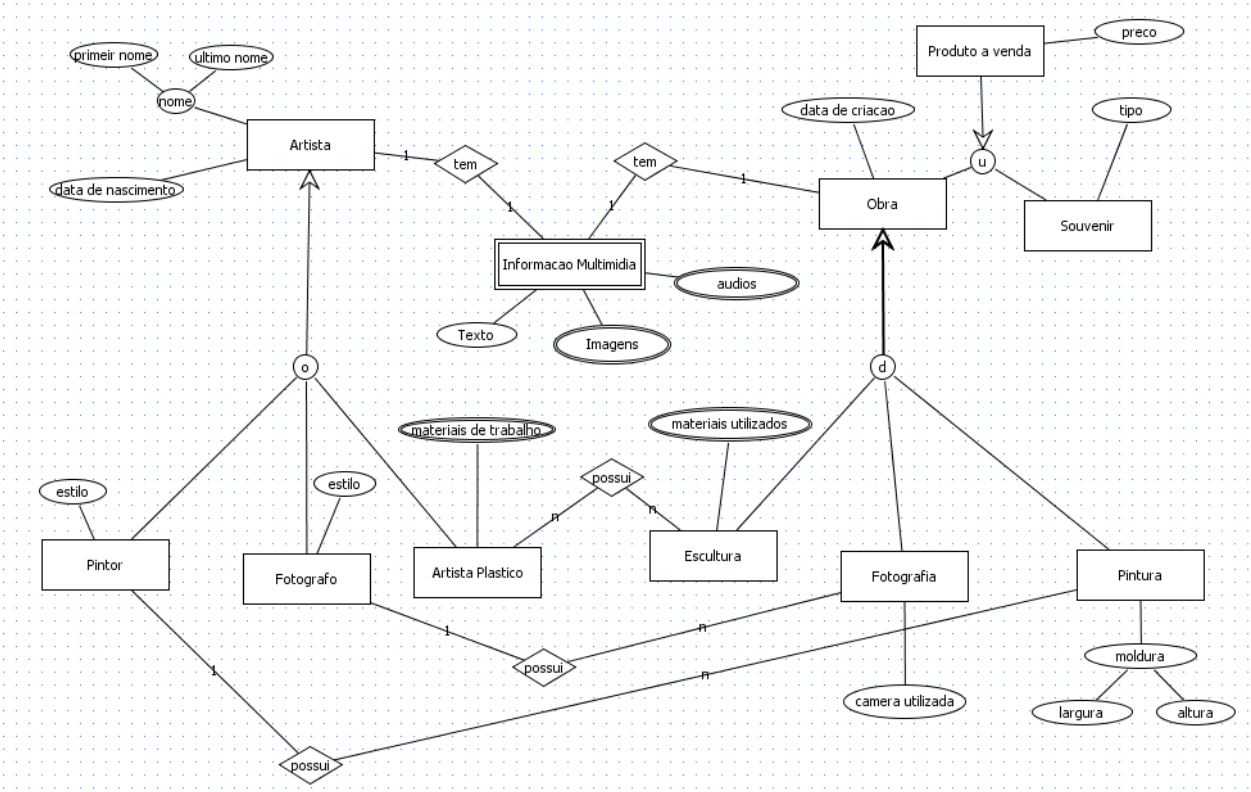


Figura 4.7 – Inserindo relacionamentos

Em cada conexão entre a entidade e o losango de relacionamento tem-se a cardinalidade deste lado do relacionamento. Apesar do modelo usado como exemplo só possuir relacionamentos entre duas entidades, é possível a criação de relacionamentos entre várias entidades. A edição das características do relacionamento é feita nas tabelas de edição das conexões e de edição dos relacionamentos, Figura 4.8.

Link	
Text	
ID	eer85
Entity ID	eer8
Properties	
cardinality	n

Relation	
Name	possui
ID	eer82
Entities count	2
Position	
Top	447
Left	245
Width	50
Height	30

Figura 4.8 – Tabelas de edição das conexões e de edição dos relacionamentos

Assim, o modelo conceitual Entidade-Relacionamento Estendido do sistema da galeria de arte multimídia encontra-se modelado. Para persistir o modelo, este pode ser salvo pela

ferramenta através da barra de ferramentas de arquivo localizada no canto superior esquerdo da ferramenta ERE-CASE.

O mapeamento do modelo conceitual em modelo lógico (e físico) no Oracle [Oracle, 2008] pode ser feita com o auxilio da ferramenta ERE-CASE2Oracle [Dantas, 2008]. O acesso a esta ferramenta é feito de forma automática pela ferramenta ERE-CASE através da barra de menu no caminho “tools>>create tables”.

5. Conclusão

Durante as pesquisas feitas neste trabalho e do desenvolvimento da ferramenta ERE-CASE, foram identificadas algumas ferramentas e esforços existentes na área de banco de dados que contemplam a utilização e difusão do modelo Entidade-Relacionamento Estendido (ERE). Porém, por conta do grande sucesso e utilização do modelo Entidade Relacionamento (ER) e de bancos de dados relacionais, atualmente ainda os mais utilizados nos projetos de banco de dados, a modelagem ERE ainda depende de uma maior aceitação na área de banco de dados.

Empresas que desenvolvem sistemas complexos que muitas vezes apresentam as características de utilização de bancos de dados orientados a objetos e de uma modelagem mais completa usando ER-Estendido, optam por adaptar o sistema a um banco de dados relacional para fugir da complexidade de modelagem de um banco de dados mais adequado. Muito disso por conta da falta de ferramentas de fácil acesso e utilização para modelagem de dados desses bancos.

Com a conclusão da ferramenta ERE-CASE e a abertura de sua utilização (disponível em https://eercase.googlecode.com) tanto para os meios acadêmicos quanto os meios mercadológicos, e com as novas pesquisas e aberturas feitas na área de bancos de dados orientados a objetos, espera-se que modelos mais complexos de dados passem a acompanhar a grande evolução alcançada e continuamente atualizada dos sistemas e aplicações de software orientado a objetos.

5.1. Contribuições

Como principal contribuição alcançada com a conclusão deste trabalho de graduação, tem-se o lançamento de uma ferramenta multiplataforma e open-source para modelagem conceitual do modelo Entidade-Relacionamento Estendido para bancos de dados. Durante as pesquisas para construção da ferramenta ERE-CASE, não foi encontrada nenhuma ferramenta que contemplasse essas características, indicando a ferramenta como única.

O desenvolvimento da ferramenta também oferece neste trabalho de graduação uma base e até mesmo um manual de estudos para possíveis expansões de suas funcionalidades. Esta preocupação de extensão da ferramenta foi uma das restrições fortemente obedecidas durante a construção da mesma, procurando-se sempre utilizar padrões de projetos e tecnologias que permitissem uma maior facilidade durante um futuro processo de expansão. A integração da ferramenta com a ferramenta ERE2Oracle de geração de scripts de criação de elementos OR para o SGBD Oracle, com a parceria de outro trabalho de graduação desta mesma instituição [Dantas, 2008], demonstra como a ferramenta oferece disponibilidade para integração e expansão de suas funcionalidades de maneira fácil e rápida.

5.2. Limitações

Apesar de ser uma ferramenta nova, sendo considerada um protótipo e por ainda precisar passar por momentos de maturação, a ERE-CASE apresenta poucas limitações conceituais, no que se diz respeito à modelagem conceitual para o modelo ER-Estendido, apresentando a possibilidade de modelar todos os elementos definidos por este modelo.

Porém, algumas limitações podem ser observadas em questões de usabilidade da mesma, que, por exatamente ainda não ter tido tempo de passar por nenhum processo profundo de estudo de usabilidade, até a conclusão deste trabalho, ainda apresenta disfunções na forma de inserção de alguns elementos na tela, ou comportamentos no que se diz respeito à comunicação com o usuário. Espera-se que estas limitações sejam sanadas assim que testes de usabilidade assistidos por profissionais sejam feitos na mesma.

Uma outra limitação da ferramenta é que esta não gera a representação gráfica do modelo lógico de dados a partir do modelo conceitual construído.

5.3. Trabalhos futuros

Como trabalhos futuros da ferramenta ERE-CASE, pode-se apontar:

i) Primeiramente sanar as limitações citadas na Seção 5.2;

ii) A geração gráfica do modelo lógico a partir do modelo conceitual gerado pela ferramenta ERE-CASE é um passo importante para que o usuário entenda como o modelo físico foi gerado e para que também possa modificar e inserir detalhes que não puderam ser contemplados durante a construção do modelo conceitual, características que apenas o modelo lógico pode ter conhecimento;

iii) Outra funcionalidade interessante de ser aplicada à ferramenta ERE-CASE é a possibilidade de engenharia reversa, em que tendo o banco de dados já implantado, a ferramenta fosse capaz de gerar os modelos conceituais e lógicos deste banco, gerando com isso uma documentação até então inexistente para o mesmo;

iv) A ferramenta precisa passar por uma bateria de testes para ser lançada e utilizada de forma segura e confiável. Testes funcionais precisam ser ministrados utilizando-se a ferramenta para que todas as falhas sejam identificadas e corrigidas antes de entrar num processo de produção e liberação para uso comercial e acadêmico; e

v) A extensão para integração com outras ferramentas que gerem os scripts de criação de tabelas para outros sistemas de gerenciamento de bancos de dados

além do Oracle, também é considerado um trabalho futuro importante para ser implantado.

Referências Bibliográficas

[Banco de dados, 2008] BANCO DE DADOS. Centro de Informática. Desenvolvido por Fernando Fonseca. Apresenta as informações sobre a disciplina de Banco de Dados no Centro de Informática. Disponível em: <http://www.cin.ufpe.br/~in940/frmain.htm>. Acesso em: 25 ago. 2008.

[Boscarioli et. al. 2006] BOSCARIOLI, C.; BEZERRA, A.; BENEDICTO, M.; DELMIRO, G. **Uma reflexão sobre Banco de Dados Orientados a Objetos.** IV CONGED, 2006.

[BrModelo, 2008] BRMODELO, Ferramenta para modelagem de BDR. Disponível em: <http://www.sis4.com/brModelo/>. Acesso em: 25 ago. 2008.

[Candido, 2008] CÂNDIDO, C. H. **brModelo: Ferramenta de Modelagem Conceitual de Banco de Dados**. Trabalho de Conclusão de Curso de Pós-Graduação (Banco de dados) – UFSC, Santa Catarina. Disponível em: <http://www.sis4.com/brModelo/monografia/Monografia.htm>. Acesso em: 01 out. 2008.

[Chen, 1976] CHEN, P. **The Entity-Relationship Model – Toward a Unified View of Data.** Artigo – MIT, 1979.

[Cooper, 2000] COOPER. J. W. **Java Design Patterns: A Tutorial.** Addison Wesley, 2000.

[DBDesigner, 2008] DBDESIGNER. Ferramenta de modelagem para banco de dados. Disponível em: <http://dbdesigner.sourceforge.net/>. Acesso em: 12 out. 2008.

[Dantas, 2008] DANTAS, R. T. **ERE2Oracle: Geração Automática de esquema relacional estendido para Objeto-Relacional no SGBD Oracle.** 2008. Trabalho de Graduação (Ciência da Computação) – Centro de Informática – UFPE, Recife

[Eclipse, 2008] ECLIPSE. Ambiente aberto de desenvolvimento de software integrado. Disponível para download em: < http://www.eclipse.org/>. Acesso em: 17 out. 2008.

[Elsmari & Navathe, 2005] ELSMARI, R.; NAVATHE, S. B. **Sistema de Banco de Dados**. 4. ed. São Paulo: Pearson Addison Wesley, 2005.

ERWIN. Ferramenta de modelagem. Disponível em: <http://www.ca.com/us/products/product.aspx?ID=260>. Acesso em: 12 out. 2008.

[GDI, 2008] GDI - GERENCIAMENTO DE DADOS E INFORMAÇÃO. Centro de Informática. Desenvolvido por Fernando Fonseca, 2008. Apresenta material dado durante o a disciplina ministrada na UFPE. Disponível em: < http://www.cin.ufpe.br/~if685/>. Acesso em: 09 out. 2008.

[Java, 2008] JAVA, Sun Developer Network. Home Page da linguagem de programação Java, desenvolvida pela Sun Microsystems. Disponível em: <http://java.sun.com/>. Acesso em: 17 out. 2008.

[JGraph, 2008] JGRAPH, Java Graphic Visualization and Layout. Componente para renderização de grafos em Java. Disponível em: <http://www.jgraph.com/jgraph.html>, acesso em : 01 Nov. 2008.

[Jude, 2008] JUDE, Ferramenta para modelagem UML, Disponível em: <http://jude.change-vision.com/jude-web/index.html>, Acesso em: 25 ago. 2008.

[Martin, 2002] MARTIN. R. C. **Visitor.** 2002.

[Nascimento, 2007] NASCIMENTO, M. C. **Br2Oracle: Geração automática de esquema relacional a partir da ferramenta BrModelo para o SGBD Oracle**. 2007. Trabalho de Graduação (Ciência da Computação) - Centro de Informática – UFPE, Recife.

[Oracle, 2008] ORACLE, Site do Oracle Database.Disponível em: < http://www.oracle.com/index.html>. Acesso em: 12 out. 2008.

[Oracle Design, 2008] ORACLE DESIGNER. Ferramenta de modelagem para Banco de Dados. Disponível em: <http://www.oracle.com/technology/products/designer/index.html>. Acesso em: 12 out. 2008.

[Oxygen, 2008] OXYGEN XML, ferramenta de edição de XML. Disponível em: < http://www.oxygenxml.com/>. Acesso em 12 out. 2008.

[Rose, 2008] RATIONAL Rose, versão 6. Disponível em: <http://www-01.ibm.com/software/awdtools/developer/rose/index.html>. Acesso em: 25 ago. 2008.

[Sommerville, 2004] SOMMERVILLE, I. **Engenharia de Software**. 6. ed. São Paulo: Pearson Addison Wesley, 2004.

[Spicer, 2003] SPICER, J. From the Editor: Edgar (Ted) Codd, 1923-2003. **Oracle Magazine**. Julho/Agosto 2003. Disponível em: <http://www.oracle.com/technology/oramag/oracle/03-jul/o43edit.html>. Acesso em: 14 out. 2008.

[StarUml, 2008] STARUML, ferramenta de edição de UML. Disponível em: < http://staruml.sourceforge.net/en/>. Acesso em: 12 out. 2008.

[Swing, 2008] SWING. Java Swing – Projeto, parte da Java Foundation Classes (JFC), para desenvolvimento de interfaces gráficas. Disponível em: < http://java.sun.com/j2se/1.5.0/docs/guide/swing/>. Acesso em: 17 out. 2008.

[Sysbase, 2008] SYBASE POWER DESIGN. Versão 12.5.0.2169. Power Design Studio Enterprise. Disponível em: <http://www.sybase.com/>. Acesso em: 19 out. 2008

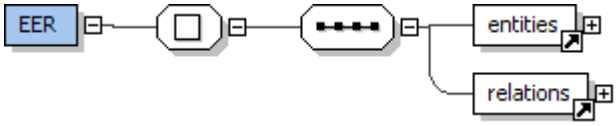
[Umbrello, 2008] UMBRELLO. Ferramenta de modelagem UML. Disponível em: <http://uml.sourceforge.net/>. Acesso em: 12 out. 2008.

[Visio, 2006] VISIO, Microsoft Office 2007. Software de modelagem. Microsoft Corporation, 2006. 1 CD-ROM.

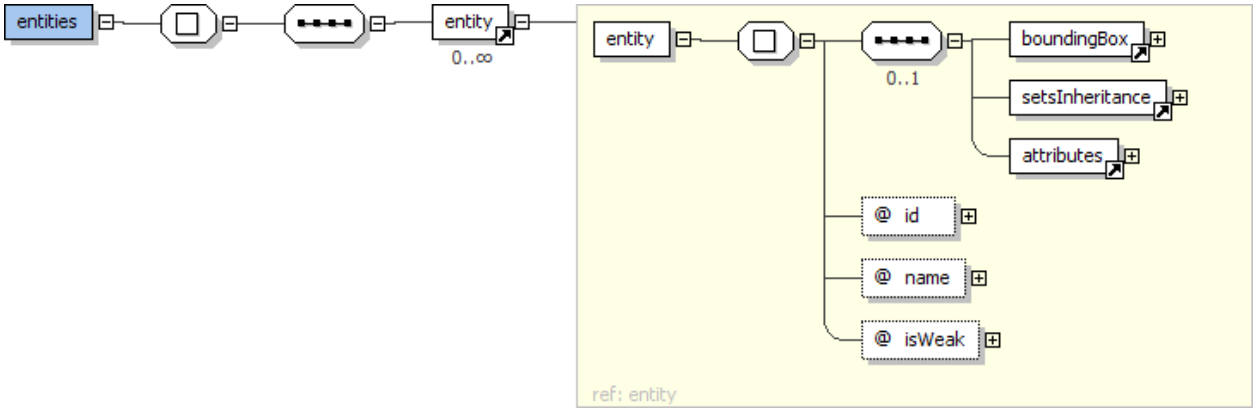
[XML, 2008] XML - EXTENSIBLE MARKUP LANGUAGE. World Wide Web Consortium. Especificação de XML. Disponível em <http://www.w3.org/XML/>. Acesso em: 10 out. 2008.

Apêndice A - Persistência

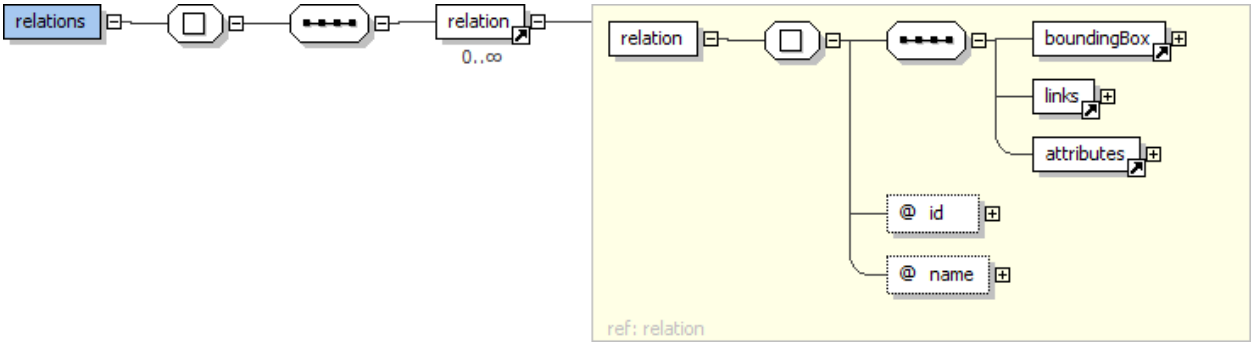
- **XML Schema do arquivo XML manipulado pela ferramenta ERE-CASE**



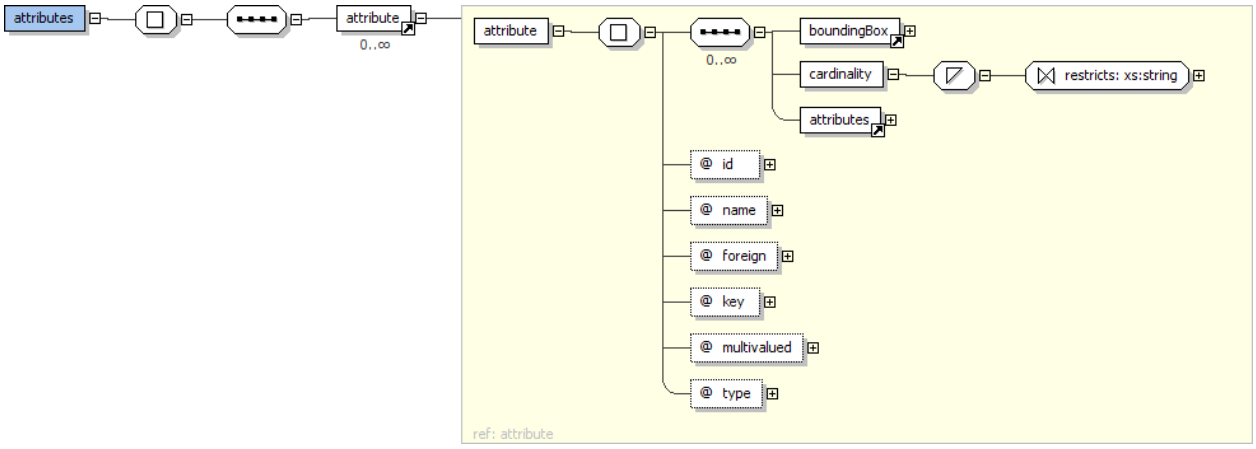
XML- Document



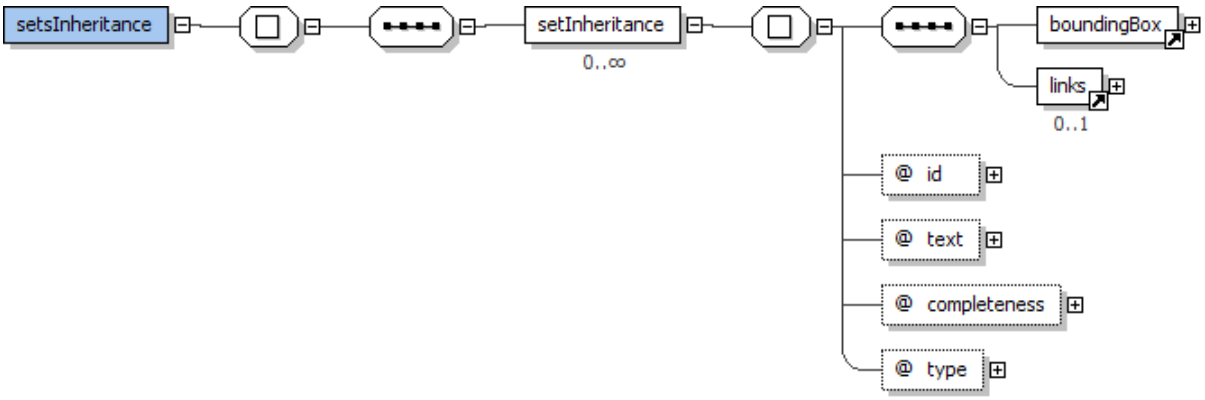
Entidades



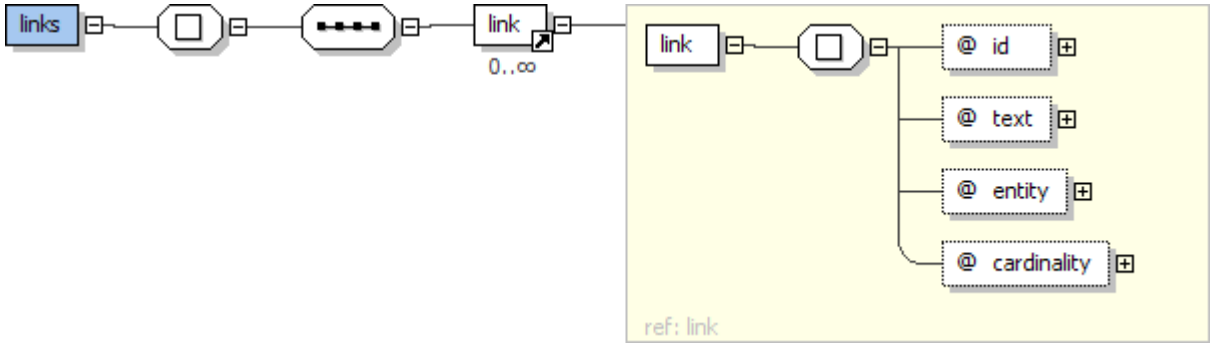
Relações



Atributos



Heranças



Conexões entre entidades, relações e heranças

Data e Assinaturas

Recife, 28 de novembro de 2008.

_____

Fernando da Fonseca Souza

(Orientador)

_____

Robson do Nascimento Fidalgo

(Co-Orientador)

_____

Renan Pereira Gouveia de Lima

(Aluno)