SENAI-SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL
CURSO TÉCNICO EM INFORMÁTICA

SISTEMA ORFANATO

QUATRO BARRAS,
2014

BIANCA CECILIO BERNARDI
ETTORE FREITAS TEIXEIRA
PATRICK WILTENBURG MIRANDA

SISTEMA ORFANATO

Trabalho de conclusão de curso apresentado ao curso Técnico em Informática do SENAI - QUATRO BARRAS, como requisito para obtenção de nota, cujo mesmo, aborda o funcionamento de um software no ramo de informática, suas utilizações e documentações técnicas.

Orientador: Prof[a.] Tiago Navarro

QUATRO BARRAS,
2014

Resumo

Ao se desenvolver um software para uma casa assistencial, o principal obstáculo é ter conhecimento sobre o que ela realmente precisa e tentar inovar de alguma forma. Mesmo quando se acha que está desenvolvendo algo original, descobre-se, após uma boa pesquisa, que existe algo semelhante.

O nosso objetivo com a criação deste software é basicamente e simplesmente ajudar essas pessoas que trabalham nessas localidades, que por sua vez fazem o melhor possível para ajudar as pessoas que mais precisam de ajuda, carinho, amor, atenção, que são as crianças.

As ações que são feitas dentro do software foram representadas em diagramas (Classes, Objetos, Sequência e Componentes).

O Diagrama de Classes permite a visualização das classes que irão compor o sistema com seus respectivos métodos e atributos.

O Diagrama de Objetos fornece uma “visão” dos valores armazenados pelos objetos das classes definidas no diagrama de classes.

O Diagrama de Sequência determina a sequência de eventos que ocorre em um determinado projeto, identificando quais métodos devem ser disparados entre os atores e objetos envolvidos e em que ordem.

O Diagrama de Implementação mostra a parte física do sistema, como os computadores, servidores, rede, softwares, etc.

O Diagrama de Componentes determina como tais componentes estarão estruturados e irão interagir para que o sistema funcione de maneira adequada.

Palavras chave: Orfanato, casa assistencial, ajuda, crianças, órfão.

Sumário

Introdução

Como todas as invenções criadas pelo homem, o computador surgiu para facilitar os trabalhos, principalmente o cálculo, a organização e acabar com as pilhas de papel e o desperdício. Em instituições, principalmente nas educacionais, o computador é fundamental para manter um controle de tudo o que acontece no local. Em um orfanato é a mesma coisa.

Porém não adianta ter um ótimo computador sem ter um sistema adequado para a situação. Nem todas as pessoas conseguem inserir dados e um banco de dados de forma manual ou converter arquivos para PDF. Um sistema inteligente atende a todas as necessidades do cliente de forma rápida e segura.

O projeto S.O (Sistema Orfanato) foi criado com o intuito de facilitar as atividades feitas em um orfanato (cadastro de órfãos, pais, funcionários, listagem, entre outros), ajudando a instituição a melhorar e ter um controle maior de quem está vinculado com o orfanato. O sistema será disponibilizado de forma gratuita para instituições carentes.

O S.O usa uma interface JSP/Bootstrap, controlado com JAVA e um banco de dados. Todas as suas ações e classes foram ilustradas utilizando diagramas UML: Diagrama de Classes, Diagrama de Objetos, Diagrama de Sequencias e o Diagrama de Componentes.

1. Tema

Eu serei minha ruina
Se eu me tornar minha obsessão.
Eu esquecerei aqueles que amo
Seu eu não servi-los.
Eu guerrearei com outros
Se me recusar a vê-los.
Portanto eu escolho me afastar
Da minha reflexão,
Confiar não em mim
Mas nos meus irmãos e irmãs,
Proteger sempre o próximo
Até que eu desapareça.
Verônica Roth-Divergente

Ajudar o próximo, confiar nos outros, ser altruísta. Características banalizadas nesse século tecnológico, onde o individualismo e o egoísmo são considerados chaves para alcançar o sucesso. Isso acaba toldando a nossa visão, e acabamos não tendo um senso de altruísmo, deixamos de fazer coisas simples, como doar roupas que não usamos mais, com medo de se passar por ingênuo e ridicularizado.

Na área da informática, na programação principalmente, parece ser difícil achar um meio de ajudar os outros, afinal códigos só funcionam em máquinas. Mas podemos sim ser altruísta usando as habilidades de programação. Podemos criar sistemas que ajudam instituições a controlar suas atividades, tendo maior controle da situação e analisando pontos que podem ser melhorados.

2. Importância

O objetivo do software é fornecer um sistema de gerenciamento para orfanatos. O sistema apresenta diversas funcionalidades, portanto também pode ser utilizado em outras instituições. Possibilitará o controle de cadastro de pais interessados em fazer adoções, dos novos órfãos que chegarem ao orfanato e dos funcionários. Realizará todo o controle de informações das crianças e dos doadores do Orfanato.

O sistema auxiliará em toda a organização do Orfanato, logo é de extrema importância. Além de facilitar no trabalho dos funcionários com a melhor organização

das informações, estará diretamente ligado com a emissão dos formulários e documentos em geral.

Pelo fato de a organização necessitar de vários mecanismos para abranger suas funcionalidades, o software apresentará um emissor de formulários com as informações das crianças e todo o gerenciamento necessário para os estoques de alimentos e medicamentos, sejam esses de doações ou não. Ainda envolvido nas doações haverá o sistema financeiro, ligando o que o orfanato recebe e seus gastos.

3. Área de Atuação

As casas assistências tem como principal objetivo oferecer assistências aos lares menos favorecidos da sociedade, aos lares desestruturados, aos lares que necessitam de algum tipo de ajuda.

O orfanato é um exemplo de casas assistenciais que tem como finalidade atender a crianças e adolescentes tendo até 18(dezoito) anos de ambos os sexos, em regime de abrigo.

> O regime abrigo não é uma internação, não há privação de liberdade. Trata-se de uma medida de apoio residencial, afetivo e provisório até que a criança ou adolescente atendido possa retornar à sua própria família ou ser colocado em família substituta. Ressalte-se que o Estatuto estabelece um prazo de dois dias úteis para que os responsáveis pelos abrigos comuniquem à Justiça os casos de acolhimento de crianças e adolescentes em seus programas sem a prévia medida judicial, encaminhados pelos Conselhos Tutelares, pelas próprias famílias ou outros Órgãos. (MOLAIB, Maria de Fátima Nunes. Crianças e adolescentes em situação de risco e suas relações com a instituição Conselho Tutelar - Página 4/5. **Jus Navigandi**, Teresina, ano 11, n. 1015, 12 abr. 2006. Disponível em: <http://jus.com.br/artigos/8231>. Acesso em: 4 set. 2014.

Por que utilizar um software para um orfanato?

O objetivo do software é resolver um problema ou otimizar um processo.

Em um lugar como este, onde a finalidade é ajudar as pessoas, é imprescindível a existência de um software que auxilie os funcionários a manter a organização.

Tendo isso como base, o objetivo deste software é auxiliar no controle interno desta casa assistencial, facilitando assim organização e tornando mais prazeroso trabalhar como também viver neste local. Ele também irá otimizar os processos de cadastramento tanto de funcionários como os de órfãos e possíveis pais dos mesmos, mostrando os órfãos que foram adotados e por quem foram adotados contendo todas as informações que poderão ser acrescentadas se necessário. Fará também o controle de pagamentos dos funcionários.

Com o avanço da tecnologia não é mais necessário toda a papelada que existia a alguns anos atrás, nós podemos ter tudo o que continha nelas armazenadas em computador.

4. Normas ISO

ISO(International Standard Organization – Organização Internacional de padrões)É uma Organização Internacional de Normalização, com sede em Genebra, na Suíça. Foi criada em 1946 e tem como associados organismos de normalização de cerca de 160 países.

A ISO tem como objetivo criar normas que facilitem o comércio e promovam boas práticas de gestão e o avanço tecnológico.

Assim como ela é uma organização de normas, ela também tem algumas normas ligadas a informática, um exemplo delas é a norma para Sistemas ISO NBR 9241-11, que é focada na usabilidade de computadores, rotinas e requisitos de interação de um sistema.

Usabilidade é a capacidade de um produto ser usado por usuários específicos para alcançar objetivos específicos com eficácia, eficiência e satisfação em um contexto especifico de uso.

> A ABNT NBR ISO 9241-11 define usabilidade e explica como identificar a informação necessária a ser considerada na especificação ou avaliação de usabilidade de um dispositivo de interação visual em termos de medidas de desempenho e satisfação do usuário. Orientação é dada sobre como descrever o contexto de uso do produto (hardware, software ou serviços) e as medidas relevantes de usabilidade de uma maneira explícita. A orientação é dada na forma de princípios e técnicas gerais, em vez da forma de requisitos para usar métodos específicos. Disponível em: < http://www.abntcatalogo.com.br/norma.aspx?ID=86090>. Acesso em: 3 out. 2014.

Ou seja, esta norma define alguns padrões para que sistemas possam incluir à usabilidade nas suas funcionalidades.

Esta norma é focada em três aspectos:

Usuários: Onde é definido os tipos de usuários que utilizarão o software, quais as habilidades e conhecimentos que esse usuário terá de ter e os atributos pessoais que é necessário para o seu uso.

Rotinas: Onde é definido a estrutura, duração e frequência de uso das rotinas no software, também a demanda física e mental que o usuário terá de ter para utilizar o sistema, as dependências das rotinas, os seus resultados e o risco resultante dos seus erros.

Equipamentos: Onde será dada uma descrição básica do sistema e a especificação da máquina para o suportar.

5. Usuários

5.1. Tipos de Usuários

5.1.1. Primário**:**

É o administrador responsável pelas rotinas de cadastro e pelo controle financeiro. Ele tem acesso total ao sistema e pode alterar qualquer informação referente ao sistema.

5.1.2. Secundário:

Sãos os funcionários, que tem acesso limitado ao sistema (cadastro e visualizar adoção), também com login.

5.2. Habilidades e Conhecimentos necessários

5.2.1. Habilidade/ Conhecimento do negócio

5.2.1.1. Primário:

Conhecimento mais profundo do sistema, pois tem maior acesso.

5.2.1.2. Secundário:

Conhecimento básico do sistema, deverá ser alfabetizado e familiarizado com o computador na parte de digitação.

5.2.2. Habilidade/ Conhecimento de Software

5.2.2.1. Primário:

Deve ter amplo conhecimento do software, saber todas as rotinas que esse realiza, para maior aproveitamento do sistema.

5.2.2.2. Secundário:

Conhecimento básico das rotinas o sistema, saber o que o sistema faz, mas somente se aprofundar nas áreas que terá contato.

5.2.3. Experiência na tarefa

5.2.3.1. Primário:

O usuário terá que ter experiência de banco de dados.

5.2.3.2. Secundário:

O usuário deverá ter experiência em realizar cadastros

5.2.4. Experiência Organizacional

5.2.4.1. Primário:

O Usuário deverá ter conhecimento do funcionamento do Orfanato para gerir com qualidade as funcionalidades do Sistema

5.2.4.2. Secundário:

Não há necessidade de experiência por parte do usuário secundário.

5.2.5. Nível de treinamento

5.2.5.1. Primário:

O usuário necessita de um treinamento mínimo, pois precisa conhecer as funcionalidades que irá utilizar para gerir o sistema.

5.2.5.2. Secundário:

Não necessita de treinamento.

5.2.6. Habilidade de entrada de dados no sistema

5.2.6.1. Primário:

O usuário deve ter um conhecimento elevado sobre os dados para não gerar furo no banco ou registrar informações incorretas.

5.2.6.2. Secundário:

Deve ter a capacidade de preencher corretamente os cadastros para efetuar o mesmo.

5.2.7. Qualificações:

Nenhum usuário necessita de qualificação.

5.2.8. Habilidade de Programação

Nenhum Usuário necessita de habilidade em programação.

5.2.9. Conhecimento geral do Sistema

5.2.9.1. Primário:

O usuário deve ter conhecimento do banco de dados do sistema e das pessoas que serão cadastradas.

5.2.9.2. Secundário:

Não necessita ter conhecimento do sistema, só precisa saber efetuar cadastros.

5.3. Atributos Pessoais

5.3.1. Idade:

Acima de 18 anos. Normalmente pessoas com mais experiência.

5.3.2. Gênero

Casa Assistencial.

5.3.3. Capacidades Físicas/Limitações

O usuário não pode ser portador de deficiência mental, visual (sem a capacidade de visualizar a tela de um computador), deficiência física (sem a capacidade de pressionar as teclas de um computador).

5.3.4. Habilidade Intelectual

O usuário deve ter habilidade e praticidade em utilizar o computador.

5.3.5. Atitude / Autodidata

O usuário necessita ser proativo, alguém que tenha força de vontade e sede em adquirir conhecimento.

6. Rotinas

6.1. Estrutura das rotinas/ duração /Frequência de uso

- Cadastrar Crianças:

  Coloca as crianças do estabelecimento no sistema, podendo ter um controle do orfanato.

- Frequência: Média.

- Cadastrar Pais:

  Cadastra interessados em adoções, para agilizar o processo de adoção e ter um conhecimento maior sobre os candidatos.

- Frequência: Média

- Cadastrar Funcionários:

  Cadastra todos os envolvidos do local, tendo controle de salário e de carga horária.

- Frequência: Alta

- Cadastrar Cargo:

  Cadastra todos os cargos do orfanato.

- Frequência: Baixa

- Login/Logout:

  Os usuários fazem o login para ter acesso ao sistema, evitando que outras pessoas não envolvidas visualizem/mudem informações.

- Frequência: Alta.

- Pesquisar:

  Permite que os usuários tenham acesso rápido a informações cadastradas no sistema.

- ➢ Frequência: Alta.

- ➢ Visualizar Adoções:

  Permite que os usuários visualizem todas as adoções feitas, podendo ter uma base de informações para referências futuras.

- ➢ Frequência: Alta.

6.2 Demanda Física e Mental

Habilidades físicas e mentais não serão cobradas em níveis altos, pois o sistema é de fácil utilização.

6.3 Dependências das Rotinas

A única dependência será a de cadastros, pois para um pai adotar uma criança, ambos deverão ter cadastro no sistema.

6.4 Resultado das rotinas:

Mensagem de sucesso ou erro e retorno ao menu principal.

6.5 Risco Resultante de erro:

- ➢ Cadastrar Crianças:

  Cadastro não realizado com sucesso.

- ➢ Cadastrar Pais:

  Cadastro não realizado com sucesso.

- ➢ Cadastrar Funcionários:

  Cadastro não realizado com sucesso.

- ➢ Login/Logout:

  Sem acesso ao sistema.

- ➢ Pesquisar:

Informação não encontrada (dados digitados de forma incorreta ou erro do sistema)

➢ Visualizar Adoções:

Adoção não encontrada, sem cadastro.

7. Equipamentos

7.1. Descrição básica

Será usado para melhorar a organização das funções no orfanato, facilitando o trabalho dos funcionários e melhorando o desempenho.

7.1.1. Identificação do sistema

Sistema Orfanato.

7.1.2. Descrição do sistema

Sistema para gerenciamento de pessoas internas e relacionadas ao orfanato, além de gerenciar o trabalho dos funcionários e doações externas.

7.1.3. Área de aplicação

O sistema poderá ser utilizado no próprio orfanato como seu destino principal, além de servir de base para sistemas de outras instituições.

7.1.4. Funções principais

As funções principais são: cadastro de pais, cadastro de crianças, gerenciamento de doações, emissor de formulários, visualizar adoções, login/logout, pesquisar.

8. Especificação

8.1. Hardware

Qualquer máquina com uma configuração intermediaria.

Processador: velocidade de processamento no mínimo de 2GHz.

Hard Disk: 320GB.

Memória RAM: 2GB.

Sistema Operacional: Windows 7, Vista, 8, 8.1.

9. Descrição da UML

UML é um acrônimo para a expressão Unified Modeling Language. Pela definição de seu nome, vemos que UML é uma linguagem que define uma série de artefatos que nos ajuda na tarefa de modelar e documentar os sistemas orientados a objetos que desenvolvemos. Essa linguagem se tornou nos últimos anos um padrão de modelagem de software, ou seja, ela não é uma linguagem de programação.

A UML surgiu da união de três metodologias de modelagem, o método Booch, o método OMT e o método OOSE e também contou com um amplo apoio da Rational Software. Foi lançado pela primeira vez em 1996, a partir disto, várias empresas começaram a contribuir com o projeto.

Em 1997, ela foi adotada pela OMG como linguagem padrão de modelagem.

O principal objetivo da UML é apresentar de forma simples e prática todas as camadas de um sistema, por meio de uma representação visual estruturada e simplificada, onde os diagramas do sistema se completam.

Os diagramas da UML são divididos em dois tipos:

9.1. Diagramas estruturais:

Os diagramas estruturais são utilizados para visualizar, especificar, construir e documentar os aspectos estáticos de um sistema, ou seja, eles mostram a estrutura do sistema. São eles os diagramas de Classe, Objetos, Componentes, Implantação, Pacotes e Estrutura.

9.2. Diagramas Comportamentais

Os diagramas comportamentais são utilizados para visualizar, especificar, construir e documentar os aspectos dinâmicos de um sistema, ou seja, eles mostram os comportamentos do sistema. São eles os diagramas de Casos de Uso, de Sequência, de Colaboração, de Estados e de Atividades.

10. Diagramas

10.1. Diagrama de Classes

É um dos diagramas mais importantes e mais utilizados da UML. Permite a visualização das classes que compõem o sistema e todos os seus relacionamentos e características.

Ele mostra uma visão estática sobre a organização das classes, definindo também a estrutura lógica das mesmas. Funciona ainda como uma base para que a maioria dos outros diagramas da UML possam ser construídos.

O diagrama é formado por suas classes e dos relacionamentos entre elas. Alguns métodos de desenvolvimento de software recomendam que o diagrama seja utilizado ainda na fase de análise, gerando um modelo conceitual com as informações necessárias ao software. Esse modelo representara, com as classes e seus atributos, o que o software necessitara, sem modelar características como os métodos que as classes poderão conter nessa etapa.

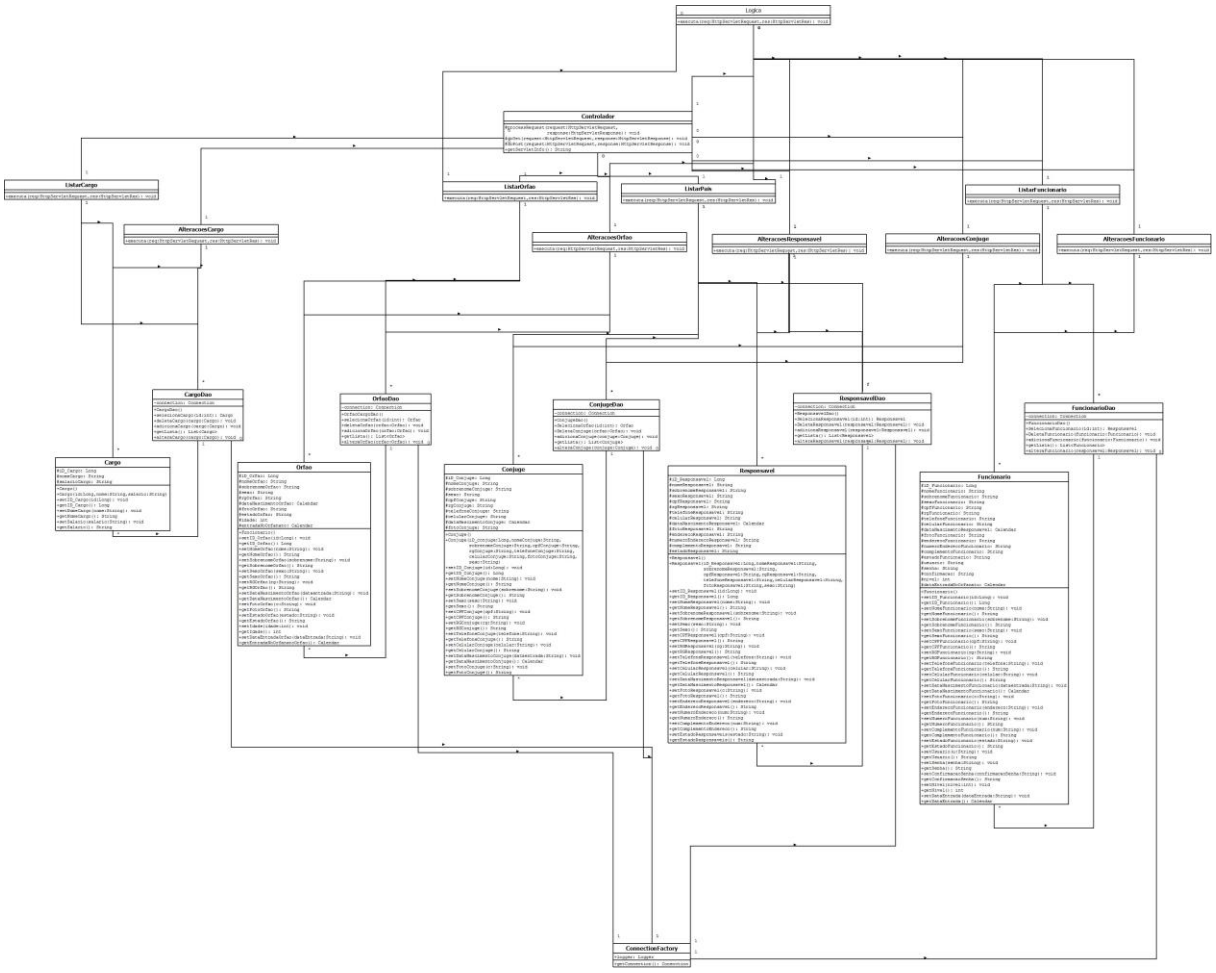
As classes possuem atributos que armazenam os dados dos objetos da classe, além de métodos, que são as funções que uma instancia da classe pode executar. Os atributos apresentam valores variados de uma instancia pra outra, possibilitando que os objetos sejam identificados individualmente, ao passo que os métodos são idênticos para todas as instancias de uma classe especifica.

Por mais que os métodos sejam declarados no diagrama de classes ele não se preocupa em definir as etapas que serão percorridas por esses métodos quando forem chamados, sendo esta uma função dada a outro diagrama, como o de atividades.

Multiplicidade	Significado
0..1	No mínimo zero(nenhum) e no máximo um. Indica que os objetos das classes associadas não precisam obrigatoriamente estar relacionados, mas se houver relacionamento indica que apenas uma instancia da classe relaciona-se com as instancias da outra classe(ou da outra extremidade da associação, se esta for unária).
1..1	Um e somente um. Indica que apenas um objeto da classe relaciona-se com os objetos da outra classe.
0..*	No mínimo nenhum e no máximo muitos. Indica que pode ou não haver instancias da classe participando do relacionamento.
*	Muitos. Indica que muitos objetos da classe estão envolvidos na associação.
1..*	No mínimo um e no máximo muitos. Indica que há pelo menos um objeto envolvido no relacionamento,

	podendo haver muitos objetos envolvidos.
3..5	No mínimo três e no máximo cinco. Estabelece que existem pelo menos três instâncias envolvidas no relacionamento e que podem ser quatro ou cinco as instâncias envolvidas, mas não mais que isso.

O diagrama de classes do sistema, mostra com perfeição as classes que se interagem dentro do software, como vocês podem ver abaixo:



10.2. Diagrama de Objetos

O objetivo do diagrama de objetos é de proporcionar uma “visão” dos valores armazenados pelos objetos das classes definidas no diagrama de classes em um determinado momento da execução de um determinado processo do sistema. Sendo assim, por mais que o diagrama de classes seja estático, é possível criar diagramas de objetos, em que as possíveis situações pelas quais os objetos das classes passarão possam ser simuladas.

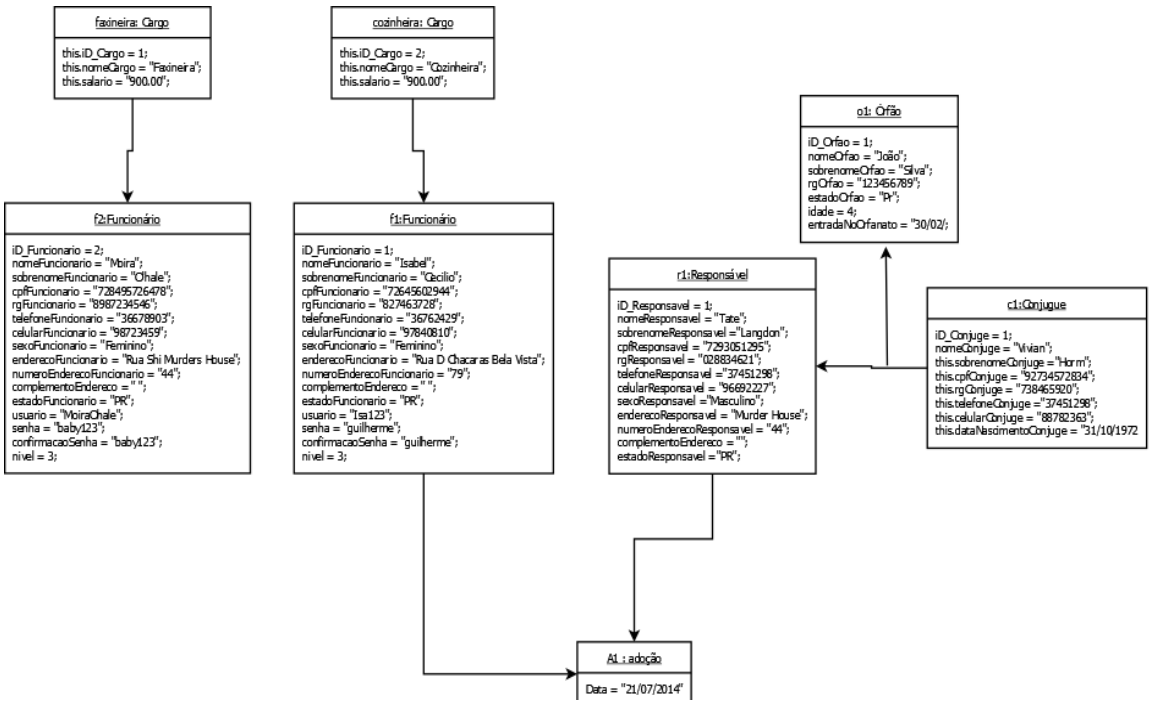
Um componente objeto é semelhante a um componente classe, porém os objetos não apresentam métodos, somente atributos, e esses armazenam os valores possuídos pelos objetos em uma determinada situação. O nome dos objetos está contido, como nas classes, na primeira divisão do retângulo que representa os objetos e pode ser apresentado de três formas:

Em um formato mais complexo, com o nome do objeto, com todas as letras minúsculas, seguido do símbolo de dois pontos (:) e o nome da classe a qual objeto pertence, com as letras iniciais maiúsculas.

O nome do objeto omitido, mas mantendo o símbolo de dois pontos e o nome da classe.

Somente o nome do objeto, sem dois pontos.

O Diagrama de Objetos do nosso Sistema, exemplifica de uma maneira bem simples quais vão ser as interações que irão ocorrer em nosso software.



10.3. Diagrama de Sequência

O diagrama de sequência determina a sequência de eventos que ocorrem em um determinado processo, além de identificar quais métodos devem ser disparados entre os atores e objetos envolvidos e em que ordem. Baseia-se no diagrama de casos de uso, havendo normalmente um diagrama

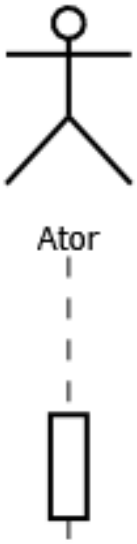
de sequência para cada caso de uso, uma vez que um caso de uso, em geral, refere-se a um processo disparado por um ator. Assim, um diagrama de sequência também permite documentar um caso de uso.

Ele depende também do diagrama de classes, visto que as classes dos objetos declarados no diagrama estão descritas nele, bem com os métodos disparados entre os objetos. No entanto o diagrama de sequência é uma excelente forma de validar o diagrama de classes, pois ao modela-lo muitas vezes percebem-se falhas e a necessidade de se declarar novos métodos que não haviam sido imaginados antes.

Os atores modelados neste diagrama são instancias dos atores declarados no diagrama de casos de uso e possuem a mesma representação, diferenciando-se por possuírem uma linha variada de vida.

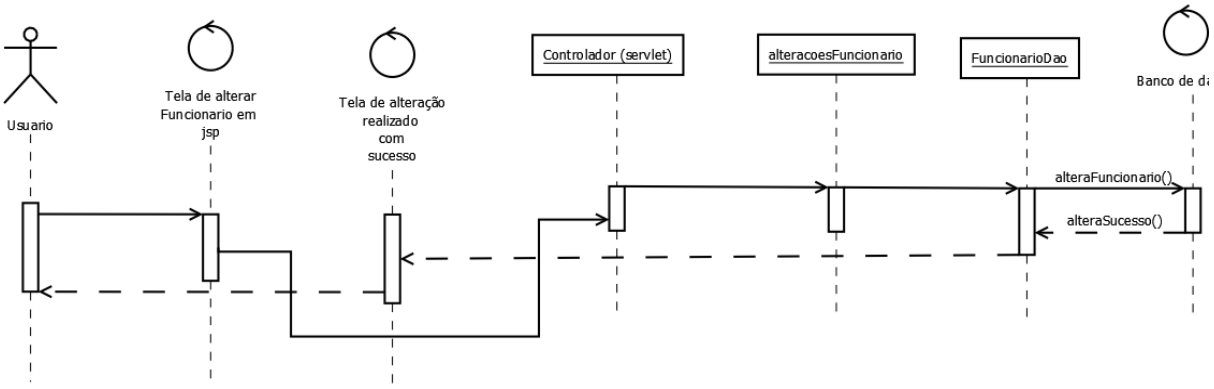
Os objetos, como chamados no diagrama, representam as instâncias das classes envolvidas no processo ilustrado pelo diagrama de sequência. São objetos que participam de uma interação durante um determinado tempo, não necessariamente desde o início do processo, e podem ser destruídos ao longo deste.

A chamada linha de vida representará o tempo em que um objeto existe durante um processo. Essas são representadas por linhas verticais que partem do objeto, e são interrompidas com um “X” quando o objeto é destruído.

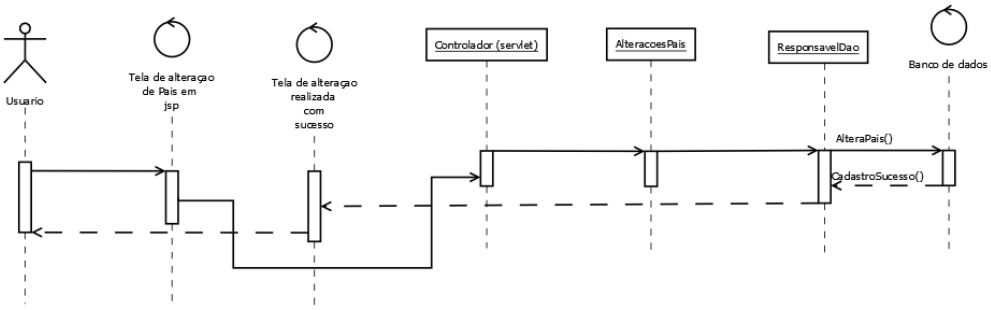


Abaixo estão os diagramas de Sequência do Sistema por nós elaborado.

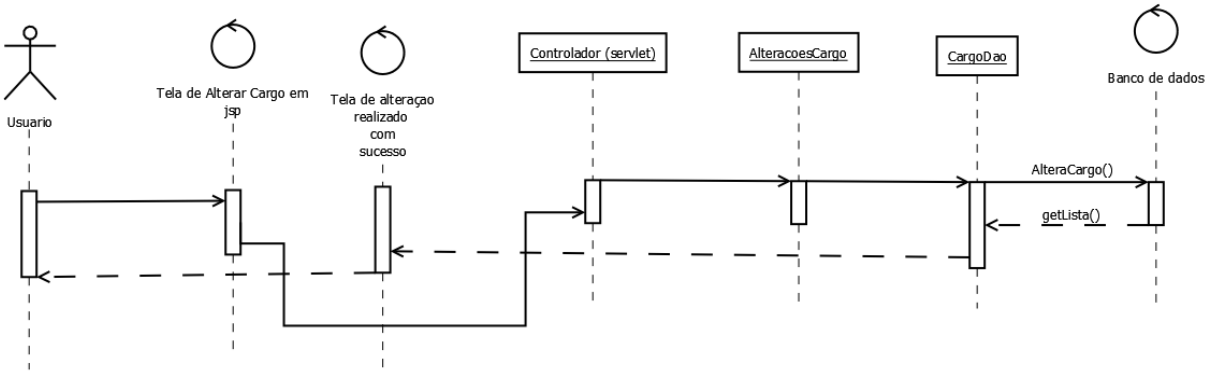
Funcionalidade: Alterar Funcionario



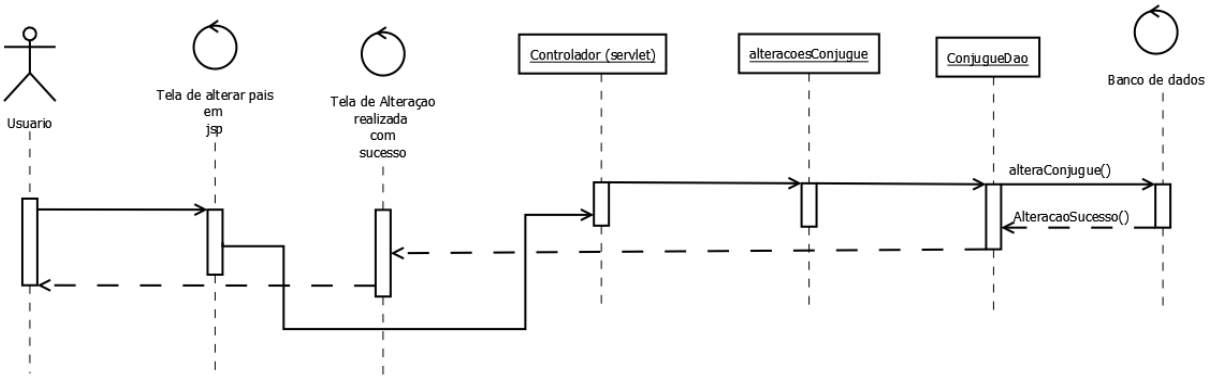
Funcionalidade: Alterar Pais



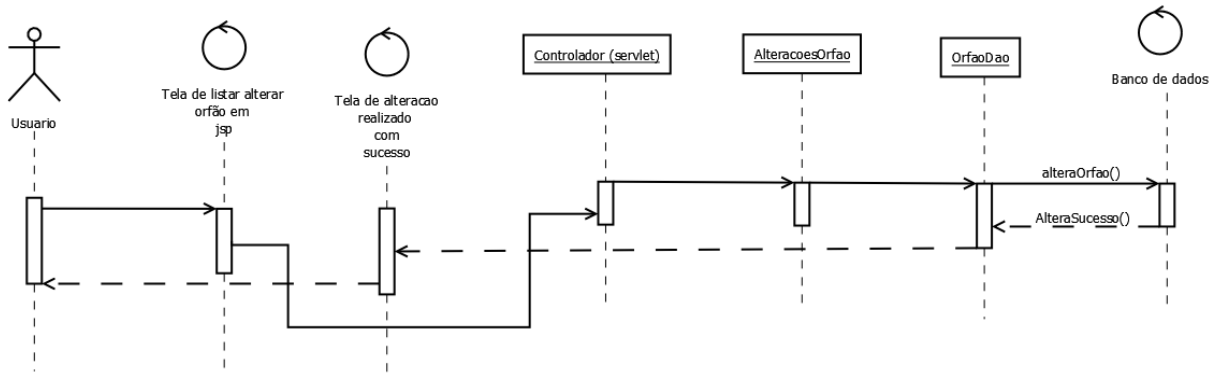
Funcionalidade: Alterar Cargo



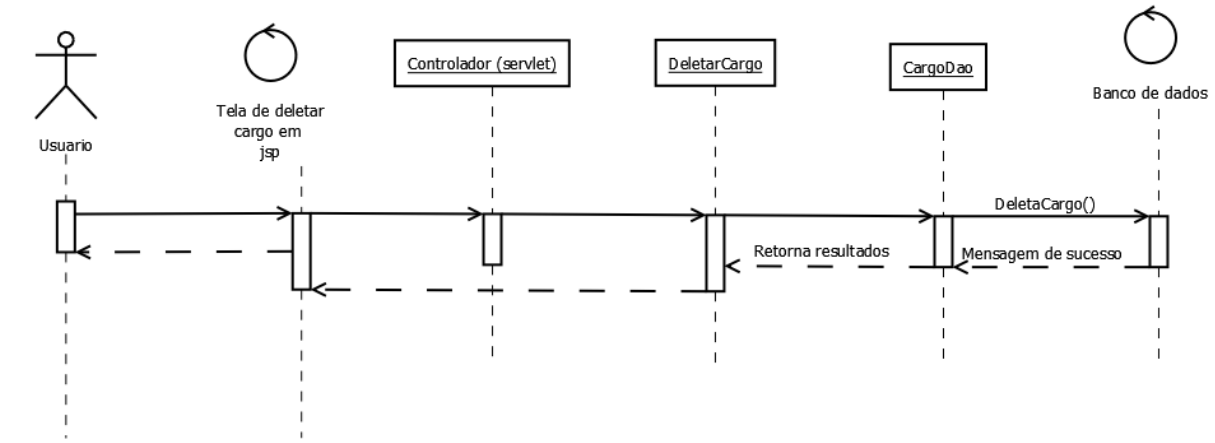
Funcionalidade: Alterar Conjugue



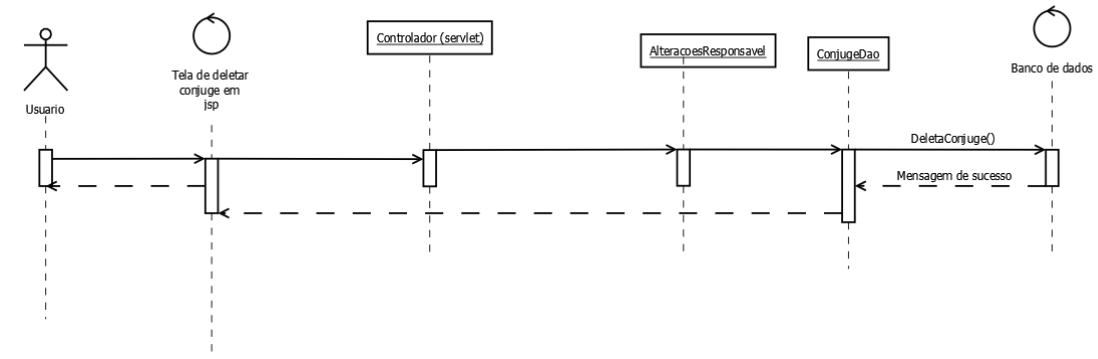
Funcionalidade: Alterar Orfão



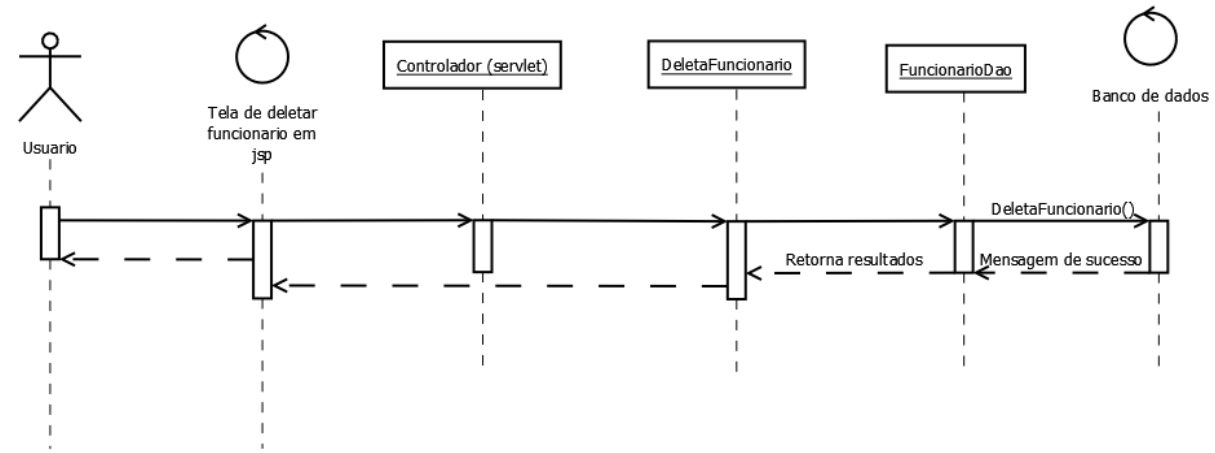
Funcionalidade: Deletar Cargo



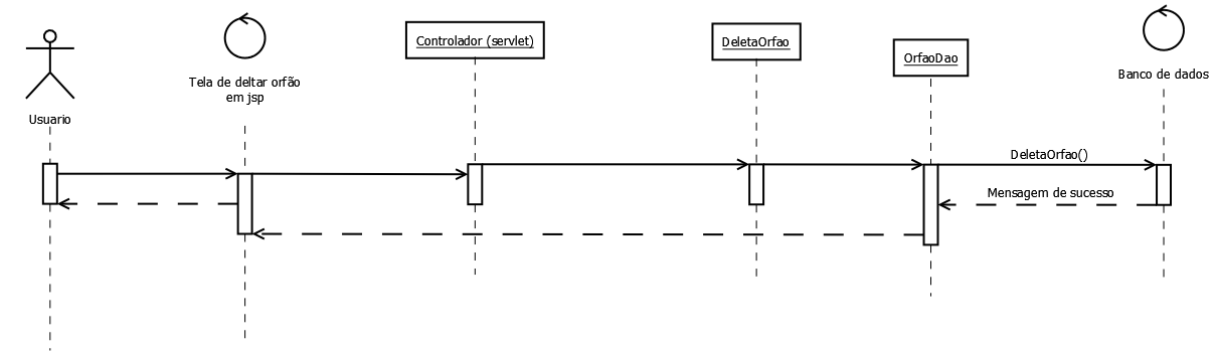
Funcionalidade: Deletar Conjuge



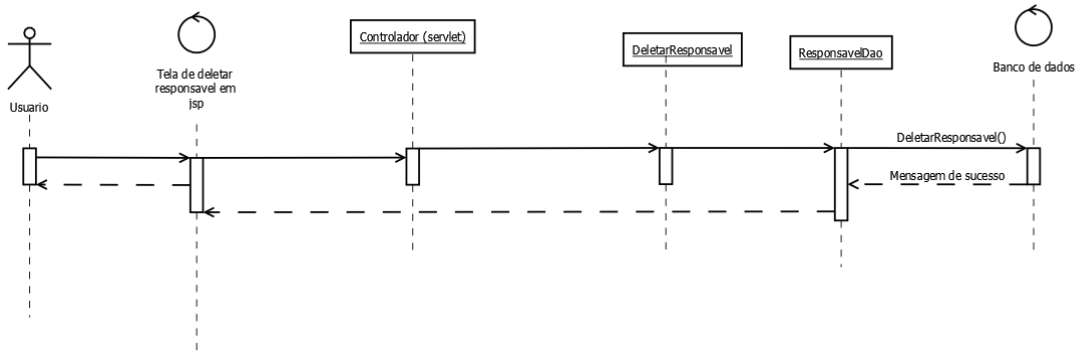
Funcionalidade: Deletar Funcionario



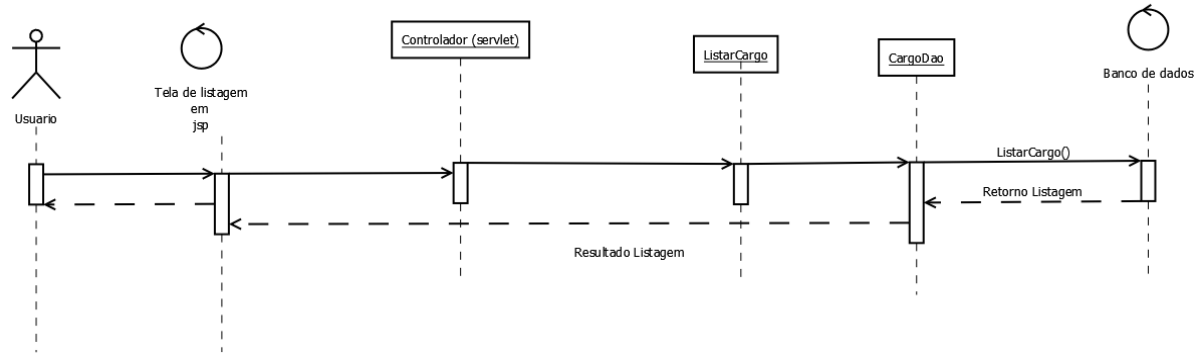
Funcionalidade: Deletar Orfao



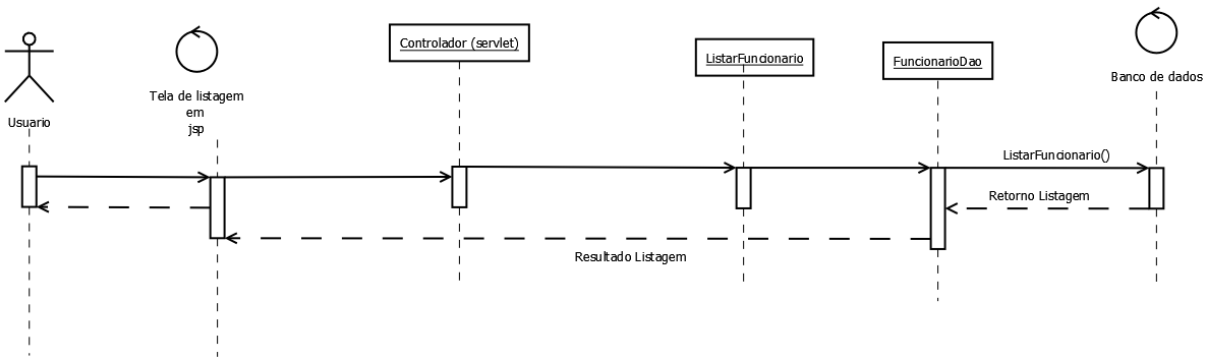
Funcionalidade: Deletar Responsavel

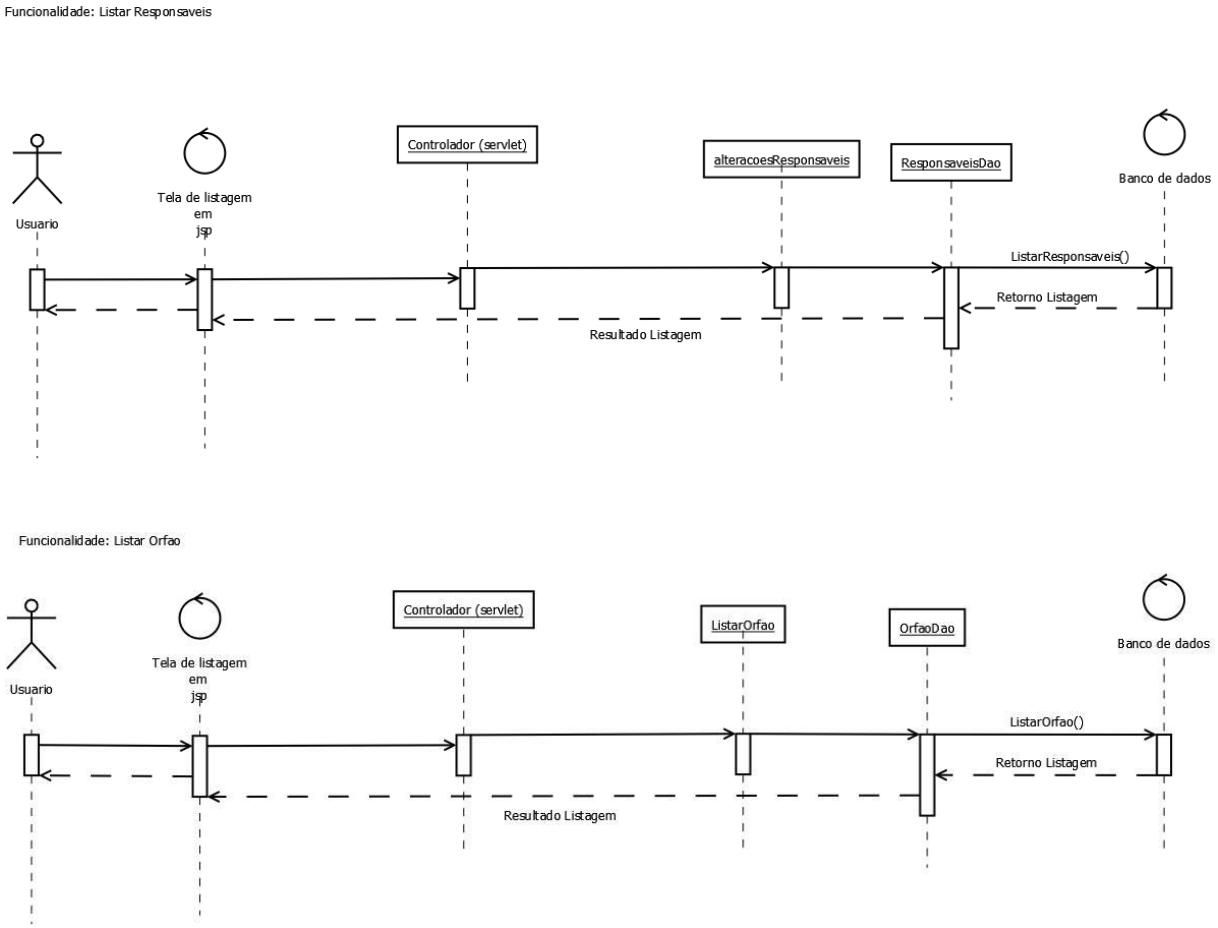


Funcionalidade: Listar Cargo



Funcionalidade: Listar Funcionarios





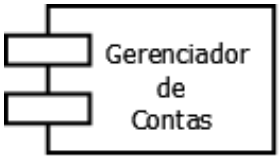
10.4. Diagrama de Componentes

O diagrama de componentes tem a função de identificar todos os componentes que fazer parte de um sistema, um subsistema ou mesmo os componentes ou classes internas de um único componente. Um componente pode ser lógico (um componente de negócio ou processo) ou físico como arquivos contendo código-fonte, arquivos de ajuda, bibliotecas, arquivos executáveis etc.

Ele pode ser usado como uma forma de documentar como os arquivos físicos estão sendo estruturados em um sistema, possibilitando que o mesmo seja melhor compreendido, facilitando também a reutilização de código.

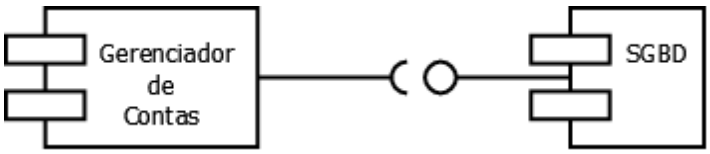
Componente

Um componente é considerado uma unidade autônoma dentro de um sistema. Ele pode conter interfaces fornecidas e requeridas, tendo seus interiores transparentes e inacessíveis por outro meio que não seja fornecido por suas interfaces.



Por possa ser dependente de outros elementos em termos de interfaces requeridas, o componente é encapsulado, assim suas dependências são projetadas de modo que ele possa ser tratado tão independentemente quanto possível.

Uma das principais formas de representar comunicação entre os componentes era realizada por meio do relacionamento de dependência. Porém passou-se a utilizar mais frequentemente interfaces fornecidas e requeridas.



No caso os dois componentes se relacionam. O componente Gerenciador de Contas tem uma interface requerida com o componente SGBD**,** e este, uma interface fornecida com o componente Gerenciador de Contas.

Para o melhor entendimento do diagrama, abaixo estão listados os diagramas que utilizamos para fazer o sistema.

Diagrama Conjuge

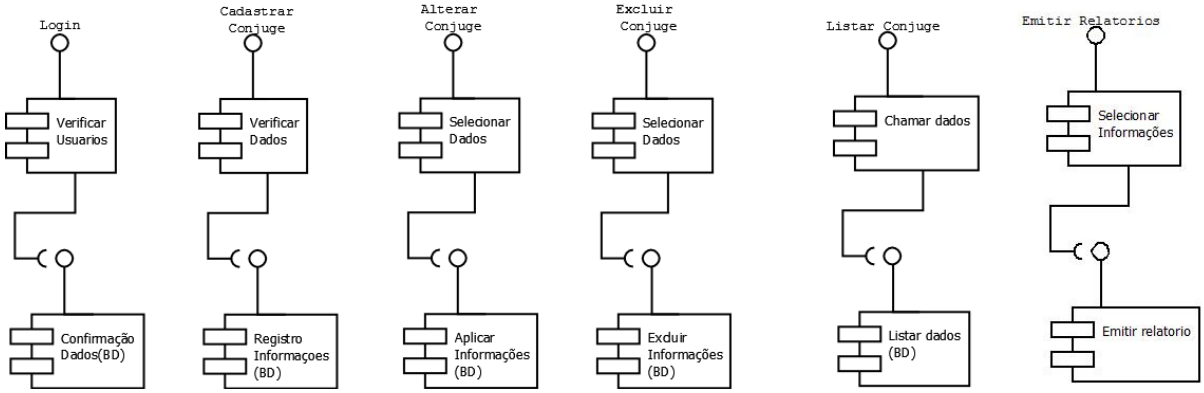


Diagrama Funcionarios

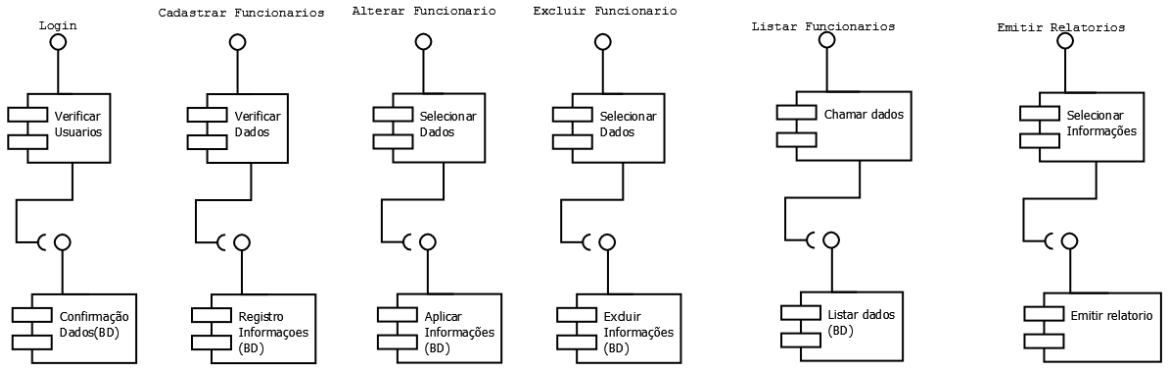


Diagrama Orfaos

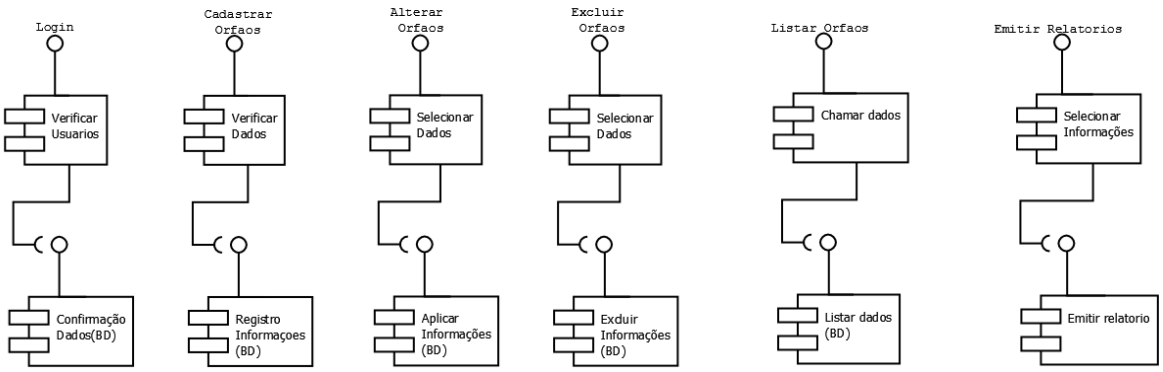


Diagrama Pais

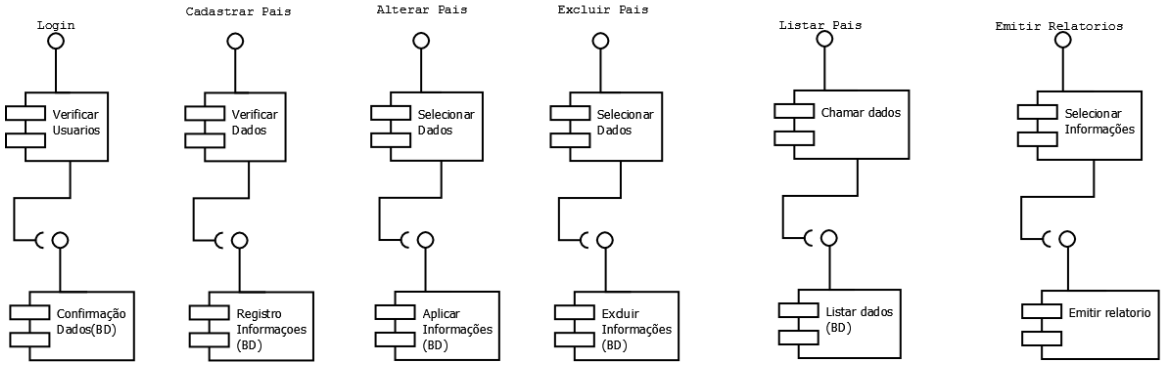
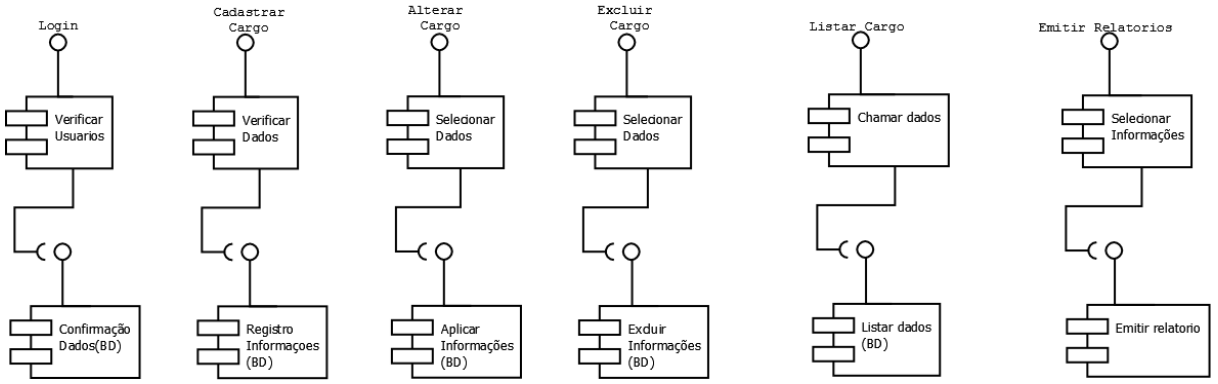


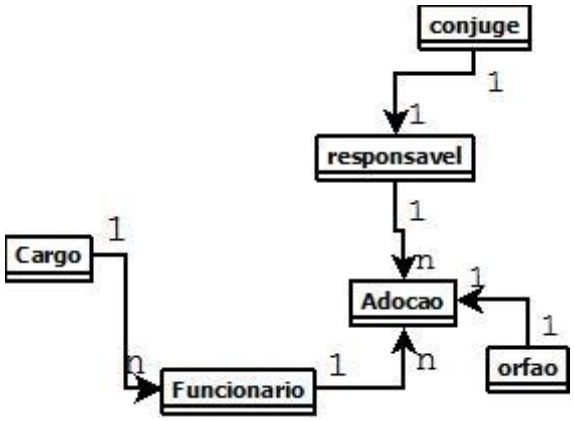
Diagrama de Cargos



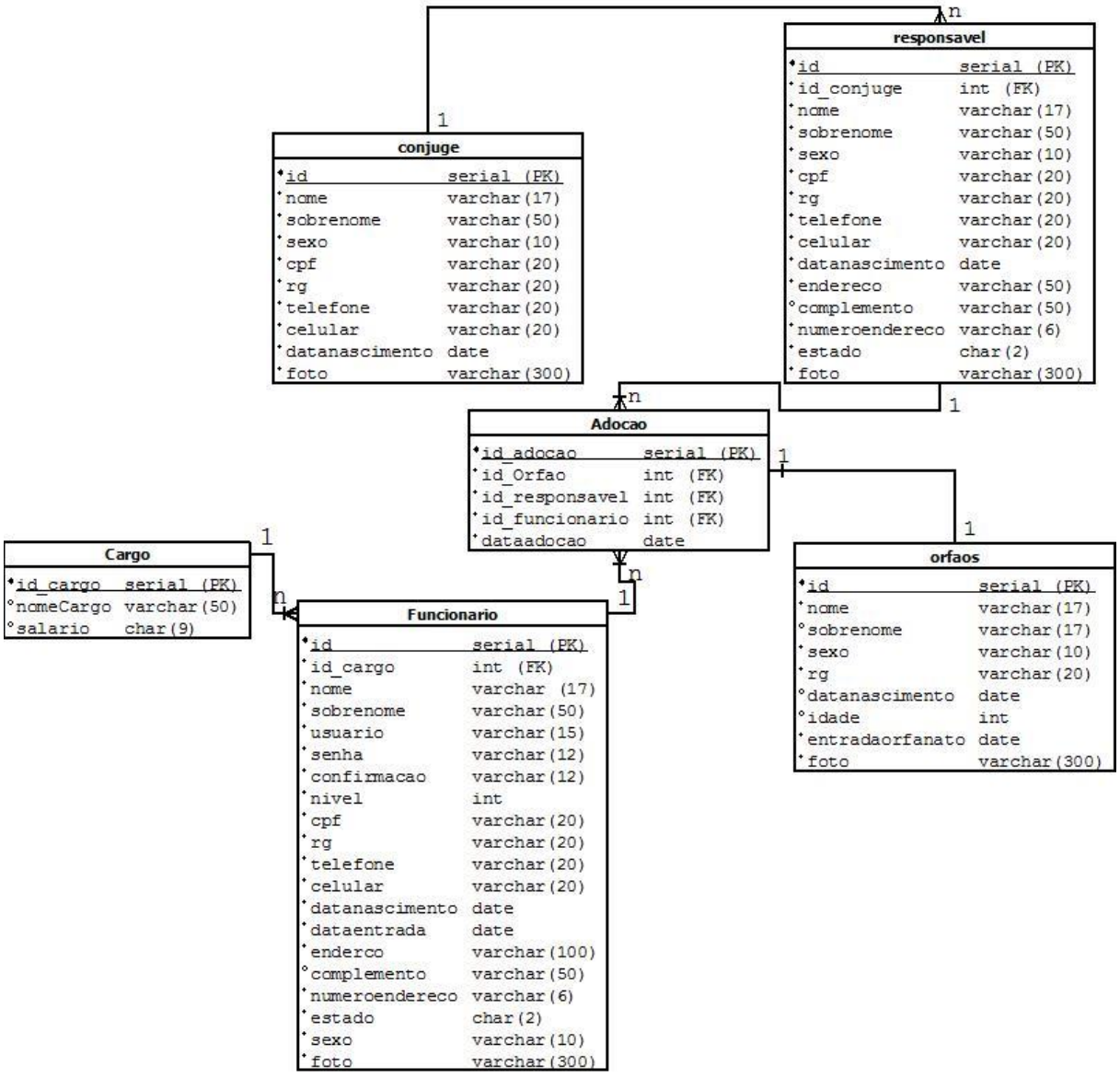
11. Banco de dados

Como o próprio nome já diz, é onde estão armazenados todos os dados do software, nos trabalhamos com o banco de dados relacional, que tem esse nome por que as suas tabelas fazem ligação uma com a outra, tornando o armazenamento mais organizado. Existem vários tipos de linguagens para se trabalhar com banco de dados, nós utilizamos o PostgreSQL. No banco de dados existem 3 modelos:

Modelo Conceitual: Mostra apenas um conceito do seu banco de dados, tendo apenas os nomes da tabela e suas relações



Modelo Logico: Já é um pouco mais completo que o conceitual, mostrando os atributos que a tabela deverá ter, seus relacionamentos



Modelo Físico: É quando você cria realmente o seu banco de Dados, com os códigos.

```
create table orfaos(
id serial primary key,
nome varchar(17) not null,
sobrenome varchar(50) not null,
sexo varchar (10) not null,
rg varchar (20) not null,
datanascimento varchar(11),
idade int not null,
entradaorfanato varchar(11),
foto varchar(300) not null
);

create table adocao(
id serial primary key,
id_orfao int not null,
id_responsavel int not null,
id_funcionario int not null,
dataAdocao varchar(11) not null,
observacao varchar(500),
foreign key (id_orfao) references orfaos,
foreign key (id_responsavel) references responsavel,
foreign key (id_funcionario) references funcionarios
);

create table funcionarios(
id serial primary key,
id_cargo int not null,
nome varchar(17) not null,
sobrenome varchar(50) not null,
usuario varchar(15) not null,
senha varchar(12) not null,
confirmacao varchar(12) not null,
nivel int not null,
cpf varchar (20) not null,
rg varchar (20) not null,
telefone varchar(20) not null,
celular varchar(20) not null,
datanascimento varchar(11) not null,
dataentrada varchar(11) not null,
endereco varchar(100) not null,
complemento varchar (50),
numeroendereco varchar (6) not null,
estado char(2) not null,
sexo varchar (10) not null,
foreign key (id_cargo) references cargo
);

create table Cargo(
id_cargo serial primary key,
nomeCargo varchar(50) not null,
salario varchar(9) not null
);
```

```
create table conjuge(
id serial primary key,
nome varchar(17) not null,
sobrenome varchar(50) not null,
sexo varchar (10) not null,
cpf varchar (20) not null,
rg varchar (20) not null,
telefone varchar(20) not null,
celular varchar(20) not null,
datanascimento varchar(11) not null,
foto varchar(300) not null
);

create table responsavel(
id serial primary key,
id_conjuge int not null,
nome varchar(17) not null,
sobrenome varchar(50) not null,
sexo varchar (10) not null,
cpf varchar (20) not null,
rg varchar (20) not null,
telefone varchar(20) not null,
celular varchar(20) not null,
datanascimento varchar(11) not null,
endereco varchar(50) not null,
complemento varchar (50),
numeroendereco varchar (6) not null,
estado char(2) not null,
foto varchar(300) not null,
foreign key(id_conjuge) references conjuge
);
```

Conclusão

Normalmente quando usufruímos de algum sistema na internet que não nos agrada, não o recomendamos para ninguém, porém com o desenvolvimento deste software podemos sentir na pele o quão complicado, estressante e difícil é desenvolver um sistema, por mais simples que o mesmo seja. Isso nos leva a repensar o que queremos para o nosso futuro, pois a profissão de programador é muito estressante e ao mesmo tempo muito prazerosa. Algumas pessoas quando necessitam de um software e vão fazer o orçamento, reclamam pelo preço, mas depois da produção desse sistema, nós temos uma melhor compreensão para este preço, por que provavelmente quem fez o sistema, teve muita dificuldade para conclui – lo.

Nas etapas de construção do sistema, descobrirmos ser extremamente bipolares, por que sempre quando ocorria um erro que não conseguíamos achar a causa, nos estressávamos e tínhamos vontade de desistir de tudo, porém a partir do momento em que achávamos a causa e a concertava, essa era a profissão dos sonhos.

Para fazer ele se tornar real, pegamos todos os nossos conhecimentos, tudo o que foi passado, para fazer ele, aprendemos muito, cada ação nova que queríamos que ele executasse, foi feito pesquisa para com que ela se tornasse possível, fazendo o trabalho se tornar muito mais agradável, pois sabemos que estamos adquirindo conhecimento que levaremos para o resto da vida.

Foi muito gratificante construirmos esse sistema, pois após concordarmos em o construí-lo, passamos a pensar como ele seria, fizemos planos, e ele está pronto, algo que nasceu em nossa mente e hoje já pode ser utilizado. Fez com que todo o estresse passado valesse a pena.

Referências Bibliográficas

UML e DIAGRAMA DE CASOS DE USO : Disponível em:
http://www.devmedia.com.br/o-que-e-uml-e-diagramas-de-caso-de-uso-introducao-pratica-a-uml/23408
Acesso: 08/06/2014

Guedes, GTA. UML 2 Guia Prático. São Paulo: Novatec, 2014

Diagrama de atividade : Disponível em:
http://www.purainfo.com.br/artigos/uml-diagrama-de-atividades/
Acesso: 20/02/2014

DIAGRAMA DE IMPLEMENTAÇÃO: Disponível em:
http://www-01.ibm.com/support/knowledgecenter/SS8PJ7_8.5.1/com.ibm.xtools.modeler.doc/topics/cdepd.html?lang=pt-br
Acesso: 09/06/2014

DIAGRAMA ESTRUTURAIS: Disponível em:
http://micreiros.com/uml-e-os-diagramas-estruturais/
Acesso: 02/10/2014

UML: Disponível em:
http://www.infoescola.com/engenharia-de-software/uml/
Acesso :02/10/2014

CONSELHO TUTELAR: Disponível em:
http://jus.com.br/artigos/8231/criancas-e-adolescentes-em-situacao-de-risco-e-suas-relacoes-com-a-instituicao-conselho-tutelar/4
Acesso 04/09/2014 às 09:51

EXEMPLO DE UM ORFANTO: Disponível em:
http://www.orfanatorenascer.com.br/

Acesso 04/09/2014 às 10:09

ABNT NBR ISO 9241-11: Disponivel em:
http://www.abntcatalogo.com.br/norma.aspx?ID=86090
Acesso 04/09/2014