Universidade de São Paulo - USP

Instituto de Ciências Matemáticas e de Computação

SSC0124 - Análise e Projeto Orientados a Objetos

Projeto 2: Modelo Conceitual, Sequência do Sistema e Contrato de Operação
Sistema de Biblioteca

9167910 Carlos Alberto Schneider Junior
8936926 Frederico de Azevedo Marques
8937420 Lucas Kassouf Crocomo
8936756 Roberto Pommella Alegro
8066395 Rodrigo das Neves Bernardi

São Carlos
2015

Sumário

Introdução

Neste projeto estaremos aplicando conhecimentos aprendidos em sala de aula para construir alguns Diagramas UML.

Um desses diagramas é o Modelo Conceitual, que irá mostrar as relações entre os conceitos presentes no sistema; apesar de se parecer muito com um diagrama de classes, é importante frisar que nem todos os conceitos mostrados ali irão se tornar classes no final do desenvolvimento (alguns, ao contrário, poderão se tornar atributos de outra classe).

Para auxiliar a construção deste Modelo Conceitual foi feita a Tabela 1 (Candidatos a Conceitos). Nela, listamos inicialmente todos os substantivos e verbos que poderiam ser importantes para o sistema, em seguida analisamos para separar os que eram de fato relevantes, e marcamos os que foram escolhidos como Candidato final, incluindo uma breve explicação do seu papel no sistema.

Depois, criamos os Diagramas de Sequência do Sistema (DSS) para os casos de uso escolhidos no Projeto 1. Note que tentamos criar os métodos de forma independente da interface, ou seja, os métodos descritos sempre irão acessar a base de dados de alguma forma; evitamos abstrações como 'Exibe tela de Cadastro' ou 'Exibe o Menu Principal'.

Por fim, foram criados quatro Contratos de Operação para quatro operações presentes nos DSS. Foi escolhida uma operação de cada tipo para fazer o contrato (Incluir, Excluir, Buscar e Alterar).

Foi utilizada linguagem PlantUML[1] para descrever os diagramas, e a *engine* Graphviz[2] para renderizar as imagens (exceto os DSS que é renderizado pelo próprio PlantUML).

PlantUML é uma linguagem para descrição de diagramas UML. Sua sintaxe é simples, porém bem completa, sendo capaz de descrever todos os diagramas da UML, de forma bem transparente até mesmo para quem tem apenas o código para consultar.

Para auxiliar na visualização das relações construídas com o PlantUML foi usada a Graphviz, que aceita várias linguagens e foi construída com o objetivo de renderizar Grafos, e como alguns diagramas da UML se assemelham muito a um Grafo, ela serviu perfeitamente para renderizar os diagramas descritos pelo PlantUML (que são esses presentes neste projeto).

Acesse os links no rodapé para mais informações sobre o PlantUML e o Graphviz, como manual de referência da linguagem, e comandos para gerar as imagens.

[1] http://plantuml.com/
[2] https://github.com/ellson/graphviz

Modelo Conceitual

Tabela 1: Candidatos a Conceitos

Conceito	Eliminado	Candidato Final	Explicação
acessar	X		
alteração	X		
ano	X		
aquisição		X	Define a entrada de novos livros no sistema
atributo	X		
atualização	X		
autor		X	Define os dados do autor
biblioteca	X		
cancelada	X		
capacidade	X		
CIC	X		
código	X		
concretizado	X		
controle	X		
data	X		
devolução		X	Define os dados da devolução
dias	X		
disponível	X		
doação		X	Define outra fonte de entrada de livros no sistema
doador		X	Define os dados do doador
edição	X		
editora		X	Define os dados da editora
emissão	X		
emprestado	X		
empréstimo		X	Gerencia os livros disponíveis e a política de empréstimo
endereço	X		
estado	X		
exclusão	X		
exemplar	X		
formação profissional	X		
fornecedor		X	Define os dados de fornecedores
funcionário		X	Define aquele que trabalha na biblioteca
hora	X		
idade	X		

impressora		X	Responsável por imprimir os relatórios e consultas
informação	x		
inserção	X		
livro		X	Guarda os dados do livro
local de publicação		X	Local onde o livro foi publicado
mensagem de erro	X		
multa		X	Valor a ser pago pelo usuário inadimplente
nome	X		
nome do representante	X		
numero	X		
numero de paginas	X		
pendente	X		
portabilidade	X		
prazo	X		
preço	X		
profissão	X		
quantidade	X		
razão social	X		
registro	X		
relatorio		X	Resumo de informações requisitadas
renovação	X		
renovado	X		
reserva		X	Define a politica e armazena os dados de reservas
reservado	X		
respeito	X		
RG	X		
salario	X		
sexo	X		
sistema	X		
sobrenome	X		
status	X		
supervisor		X	Define aquele que gerencia os funcionários
tela		X	Responsável por exibir relatórios e consultas
telefone	X		
tempo	X		
terminais		X	Pontos de acesso ao sistema
título	X		
usuário		X	Define aquele que utiliza o sistema
valor	X		
volume	X		

Modelo Conceitual

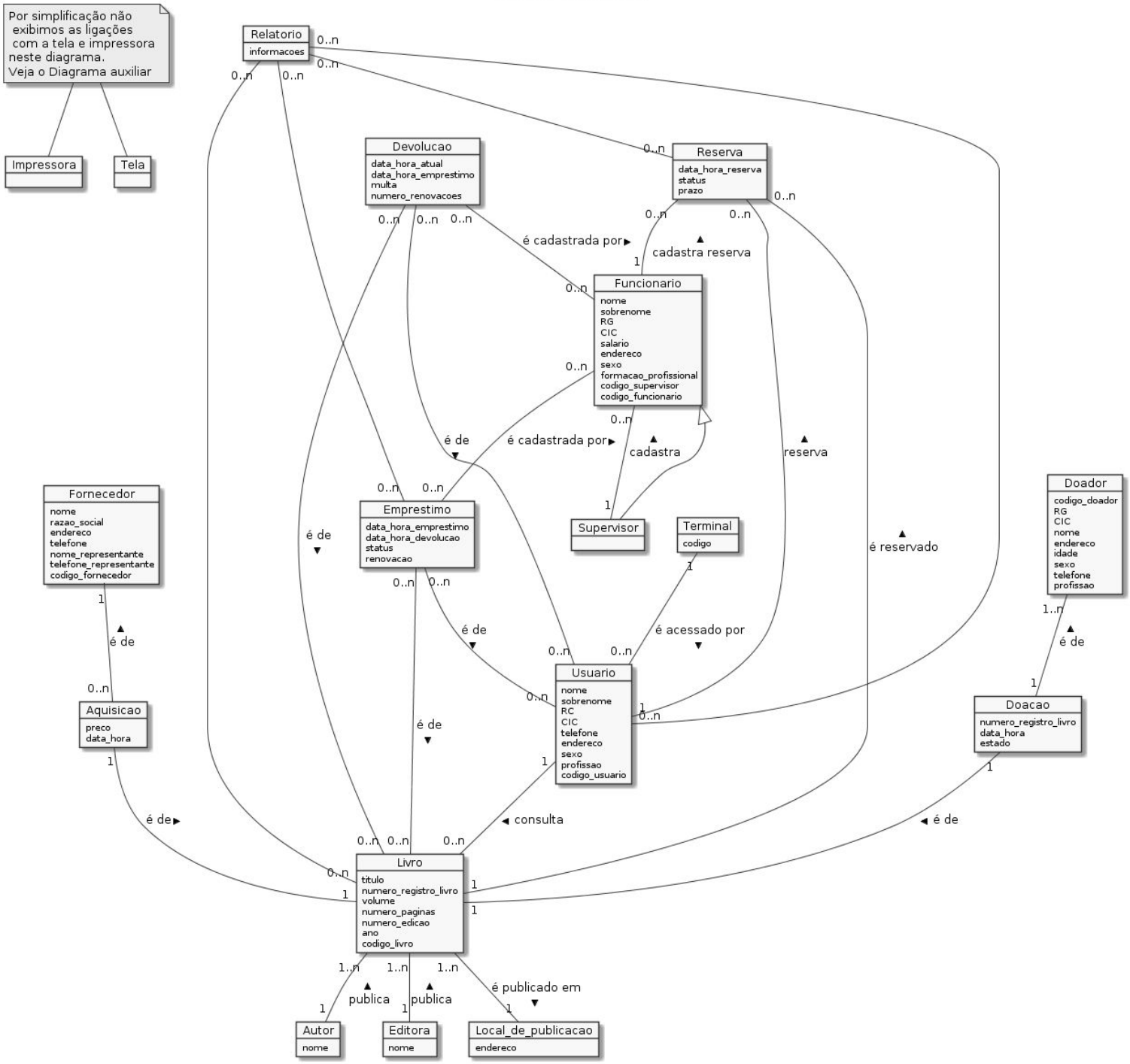
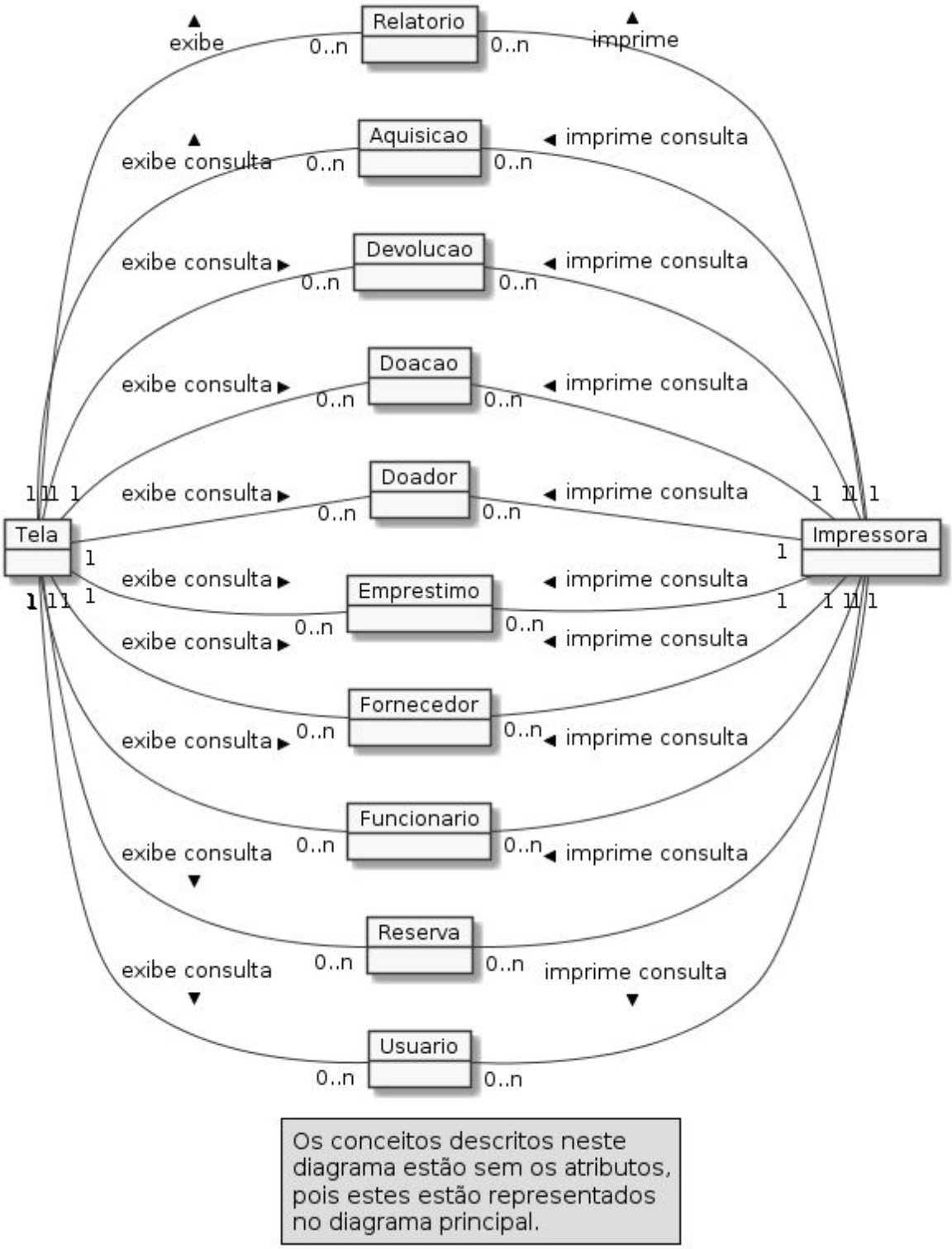


Diagrama Auxiliar do Modelo Conceitual



Diagramas de Sequência do Sistema

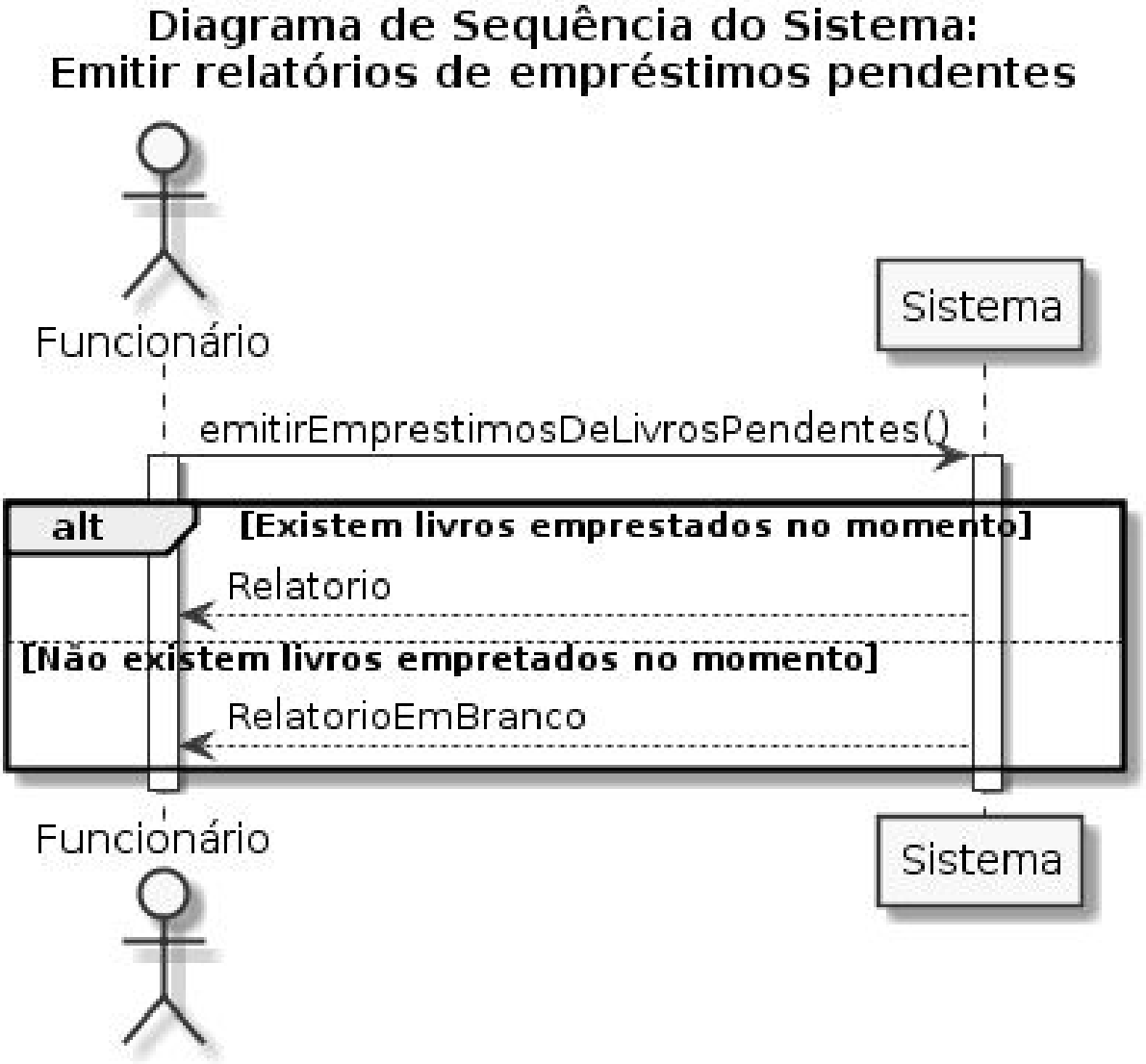


Diagrama de Sequência do Sistema: Incluir Livro

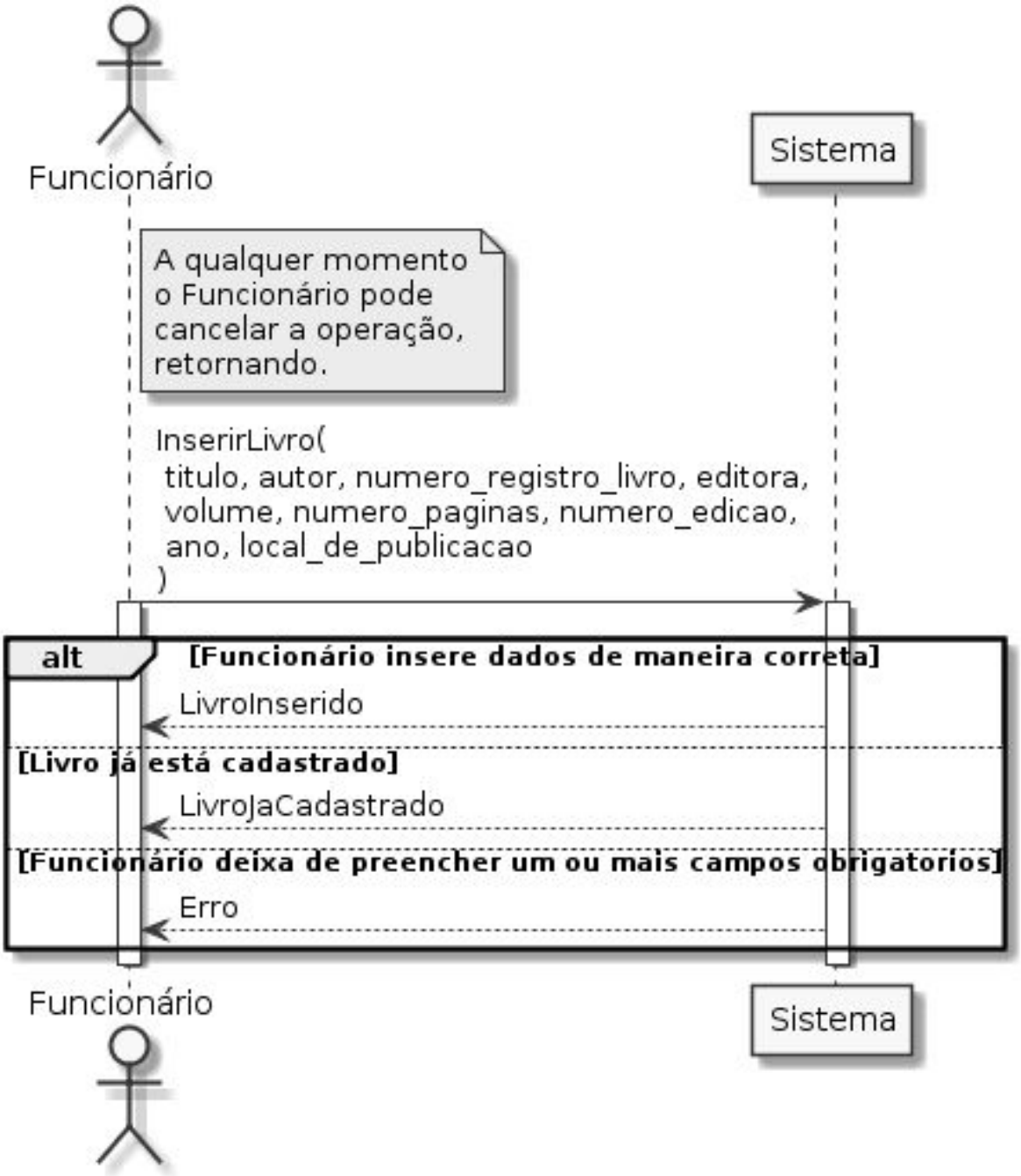


Diagrama de Sequência do Sistema: Consultar Usuário

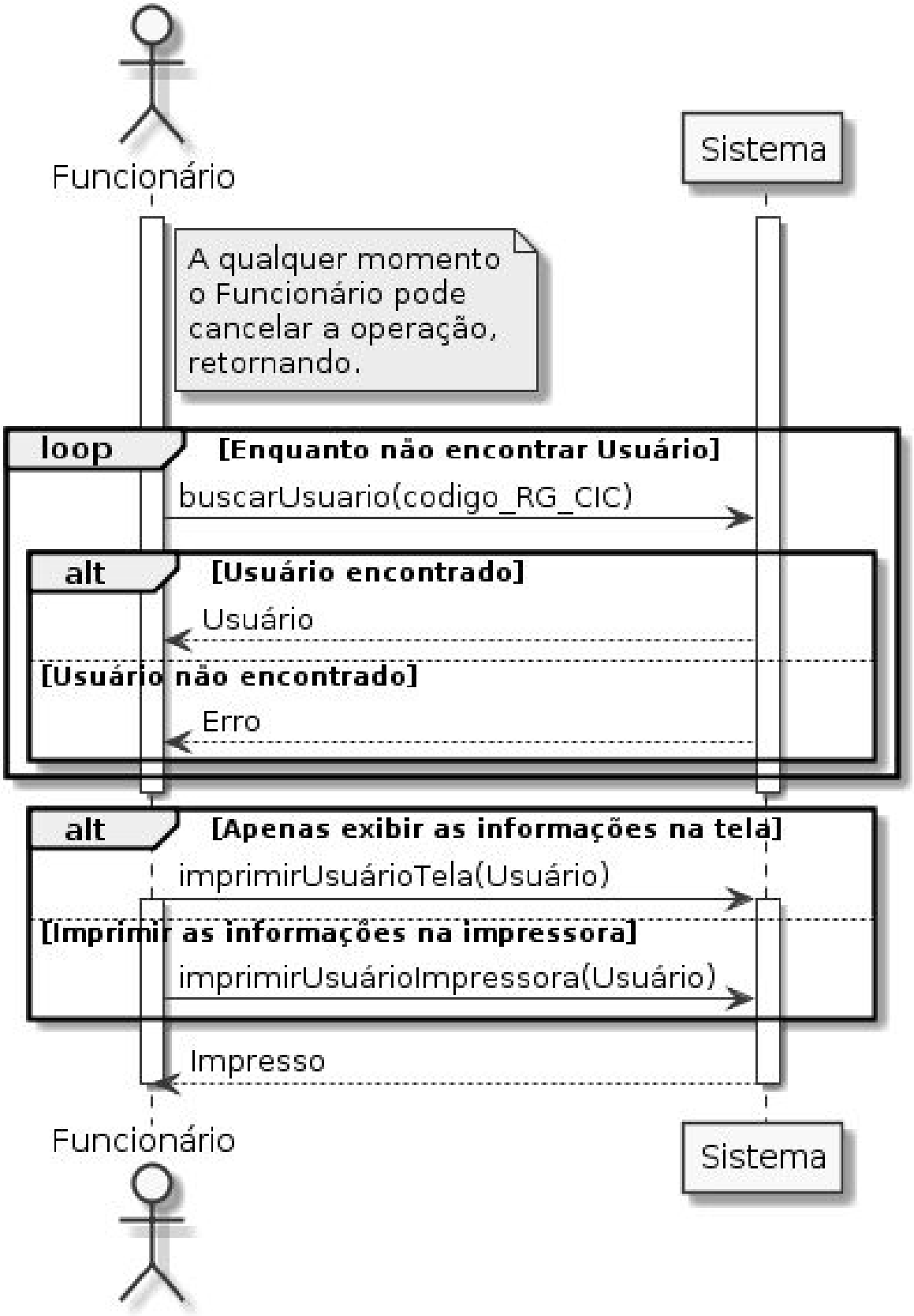


Diagrama de Sequência do Sistema: Alterar Funcionário

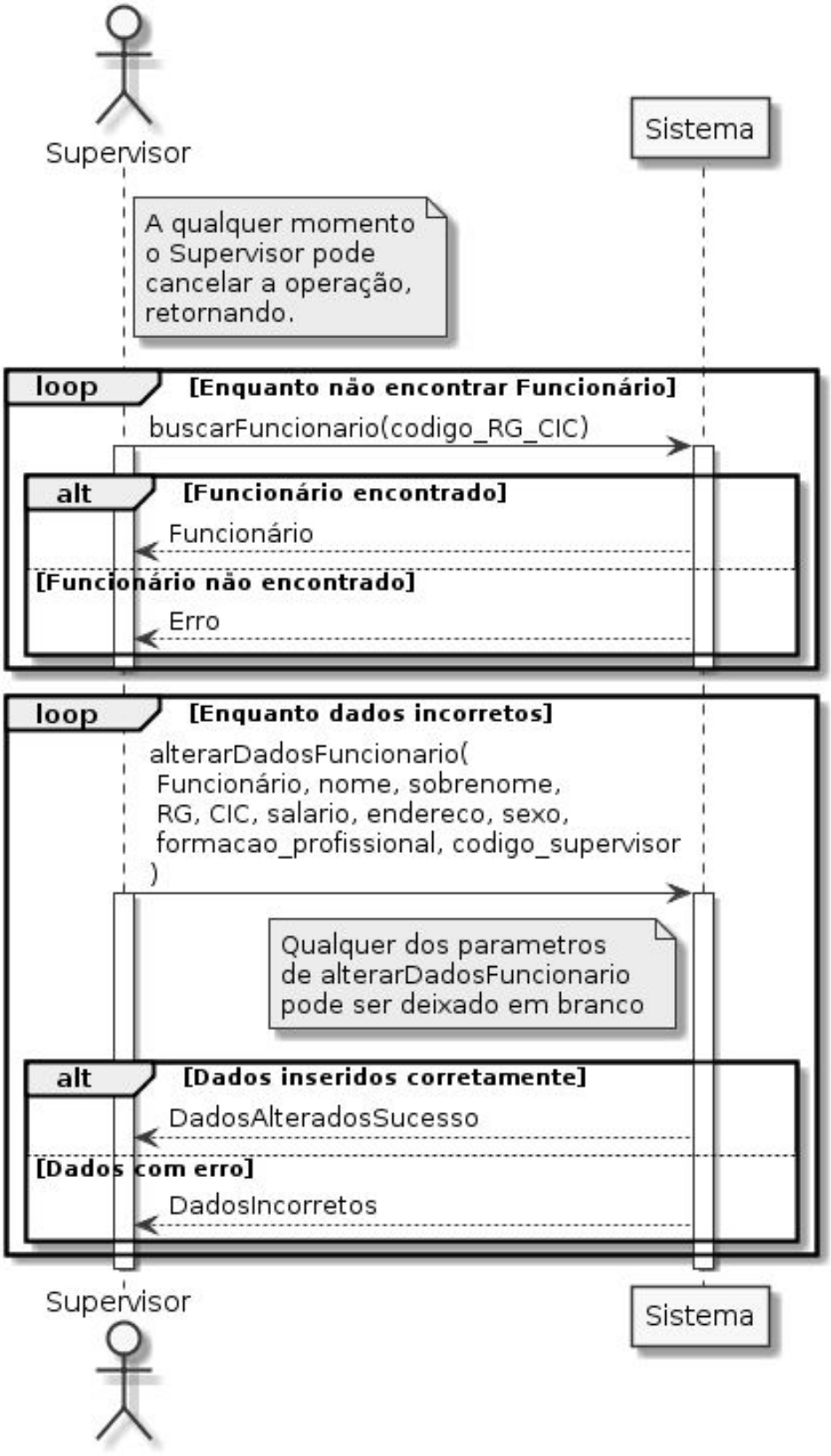
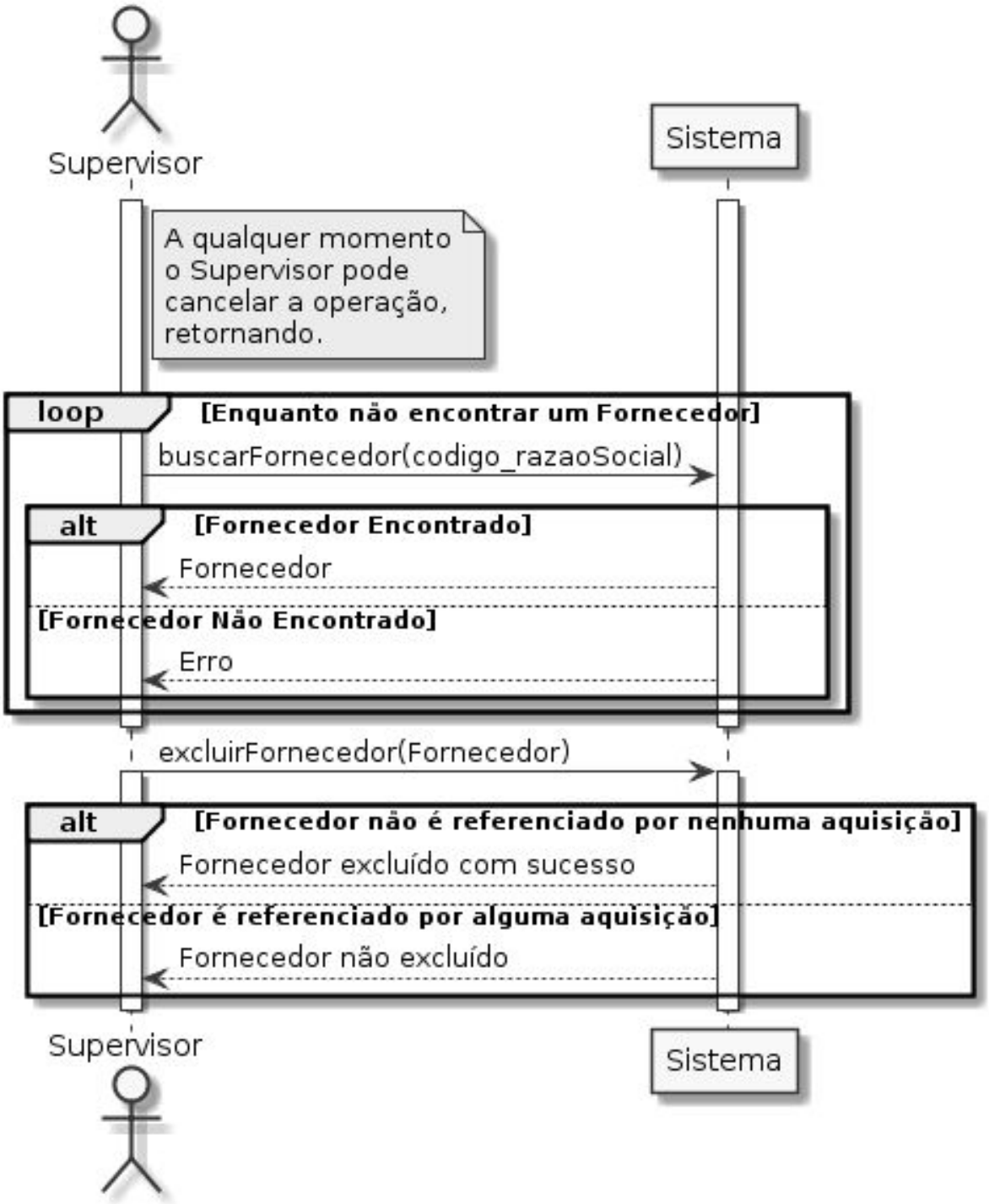


Diagrama de Sequência do Sistema
Excluir Fornecedor



Contratos de Operação

Nome da operação	`InserirLivro`
Parâmetros de entrada	`titulo` - Título do Livro `autor` - Autor do Livro `numero_registro_livro` - Número de Registro (ISBN) do Livro `editora` - Editora que publicou o Livro `volume` - Volume do Livro `numero_paginas` - Número de Páginas do Livro `numero_edicao` - Edição do Livro `ano` - Ano de publicação do Livro `local_de_publicacao` - Local de Publicação do Livro
Referências Cruzadas	Caso de Uso: ‘`Incluir Livro`’
Pré-condições	Existe uma Aquisição de um Fornecedor ou Doação de um Doador referente ao livro `` `l` `` que se quer inserir.

Pós-condições	É criado um Livro `` `l` ``.
	`` `l.titulo = titulo` ``
	`` `l.autor = autor` ``
	`` `l.numero_registro_livro = numero_registro_livro` ``
	`` `l.editora = editora` ``
	`` `l.numero_paginas = numero_paginas` ``
	`` `l.numero_edicao = numero_edicao` ``
	`` `l.anos = anos` ``
	`` `l.codigo_livro = codigo_livro` ``
	Caso não exista um Autor com nome `` `l.autor` `` no sistema, é criado o Autor `` `a` `` e `` `a.nome = l.autor` ``.
	Caso não exista uma Editora com nome `` `l.editora` `` no sistema, é criada a Editora `` `e` `` e `` `e.nome = l.editora` ``.
	Caso não exista um Local_de_publicacao com nome `` `l.local_de_publicacao` `` no sistema, é criado o Local_de_publicacao `` `loc` `` e `` `loc.nome = l.editora` ``.

Nome da operação	`alterarDadosFuncionario`
Parâmetros de entrada	`Funcionário` - Um funcionário existente `nome` - O novo nome do funcionário `sobrenome` - O novo sobrenome do funcionário `RG` - O novo RG do funcionário `CIC` - O novo CIC do funcionário `salario` - O novo salário do funcionário `endereco` - O novo endereço do funcionário `sexo` - O novo sexo do funcionário `formacao_profissional` - A nova formação profissional do funcionário `codigo_supervisor` - O código do novo supervisor do funcionário NOTE: Qualquer um dos parâmetros pode estar em branco
Referências cruzadas	Caso de uso: ‘`Alterar Funcionário`’
Pré-condições	Existe um Funcionario `` `func` `` e deseja-se alterar alguma(s) informações dele. Essas informações são conhecidas.

Pós-condições	se os dados são validos:
	`` `func.nome = nome` ``
	`` `func.sobrenome = sobrenome` ``
	`` `func.RG = RG` ``
	`` `func.CIC = CIC` ``
	`` `func.salario = salario` ``
	`` `func.endereco = endereco` ``
	`` `func.sexo = sexo` ``
	`` `func.formacao_profissional = formacao_profissional` ``
	`` `func.codigo_supervisor = codigo_supervisor` ``
	se os dados são inválidos:
	Nenhuma alteração é salva no `` `func` `` e é retornado um erro, mostrando que algum dado estava incorreto.

Nome da operação	buscarUsuario
Parâmetros de entrada	codigo_RG_CIC - O código, RG ou CIC do usuário a ser buscado
Referências cruzadas	Caso de uso: ‘Consultar Usuário’
Pré-condições	
Pós-condições	É retornado um usuário `u` tal que `u.codigo_usuario == codigo_RG_CIC` ou `u.RG == codigo_RG_CIC` ou `u.CIC == codigo_RG_CIC` ou um erro, caso nenhum usuário seja encontrado.

Nome da operação	excluirFornecedor
Parâmetros de entrada	Fornecedor - Um fornecedor existente a ser excluído
Referências cruzadas	Caso de uso: ‘Excluir Fornecedor’
Pré-condições	Existe um Fornecedor `f` que se quer excluir.
Pós-condições	O Fornecedor `f` é excluído caso `f.aquisicoes.size == 0`.

Observações

1. Para melhor visualização e simplificação da modelagem, as relações de tela e impressora com os demais componentes não foram dispostos no “Modelo Conceitual”. Tais ligações são encontradas no “Diagrama Auxiliar do Modelo Conceitual”.

2. Ao iniciar a modelagem, o grupo abordou o trabalho com uma prerrogativa inicial de que o programa seria executado em dependência de uma interface especifica. Contudo, esse conceito foi revisado e a modelagem atual se encontra independente de uma interface específica e, por tanto, modularizada.

Conclusão

Um dos objetivos da modelagem de software é permitir correções e alterações pertinentes no programa final, antes que este seja criado, evitando assim reimplementações desnecessárias, como foi o caso neste exercício.

Iniciamos a modelagem com uma determinada abordagem[3], analisamos e entendemos que o programa poderia ser aprimorado através de uma abordagem diferente. Se o programa estivesse em sua fase de desenvolvimento, seria necessário o depreendimento de recursos adicionais para a modificação realizada. Contudo, por se tratar apenas da modelagem, o programa pode ser facilmente modificado, sem a necessidade de mais recursos.

Percebe-se assim, o papel fundamental da modelagem de software para a criação de uma solução final e a sua devida importância. Aqui se tratava apenas de um exercício, no entanto se a mesma situação ocorresse em um ambiente competitivo onde tempo e dinheiro são os recursos primários, seria notória a diferença entre aplicar ou não a molagem – tempo e dinheiro não desperdiçados.

[3] Observação 2, pág. 14