Trabalho de Engenharia Reversa

Sistema de Livraria

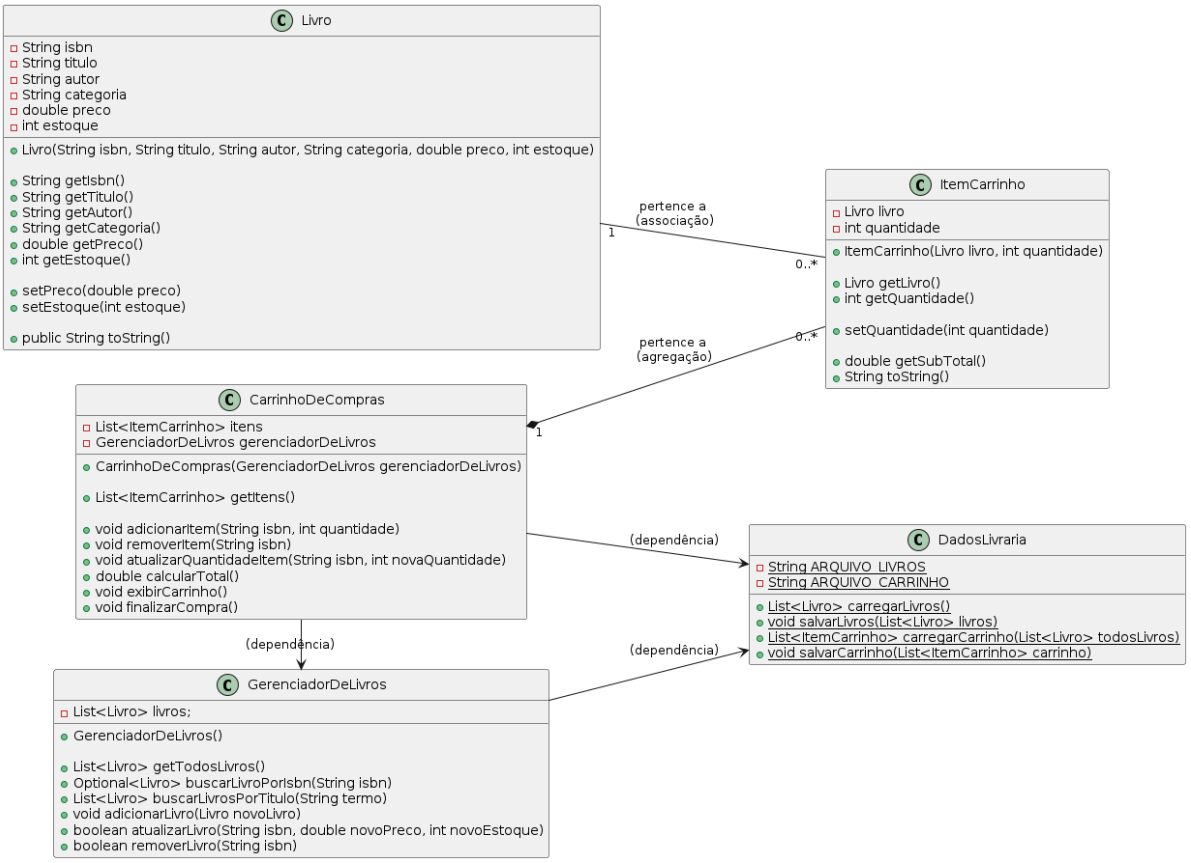
Rafael Gama

Diagrama de Casos de Uso

Código PlantUml do diagrama de casos de uso

```
@startuml useCases
left to right direction
actor Administrador
actor Cliente

rectangle "Sistema Livraria" {
 (Listar livros disponíveis) as showBooks
 (Adicionar livro) as addBook
 (Atualizar preço e estoque de livro) as updateBook
 (Remover livro) as deleteBook

 (Exibir Carrinho) as showCart
 (Adicionar item ao Carrinho) as addToCart
 (Atualizar quantidade do item) as updateItem
 (Remover item do Carrinho) as removeFromCart
 (Finalizar compra) as endPurchase
 'caso não visível ao usuário, mas usado internamente
 (Calcular total) as calculateTotal

 Administrador -- showBooks
 Administrador -- addBook
 Administrador -- updateBook
 Administrador -- deleteBook

 Cliente -- showCart
 Cliente -- addToCart
 Cliente -- updateItem
 Cliente -- removeFromCart
 Cliente -- endPurchase

 showCart -- calculateTotal : <<include>>
}
@enduml
```

Diagrama de Classe



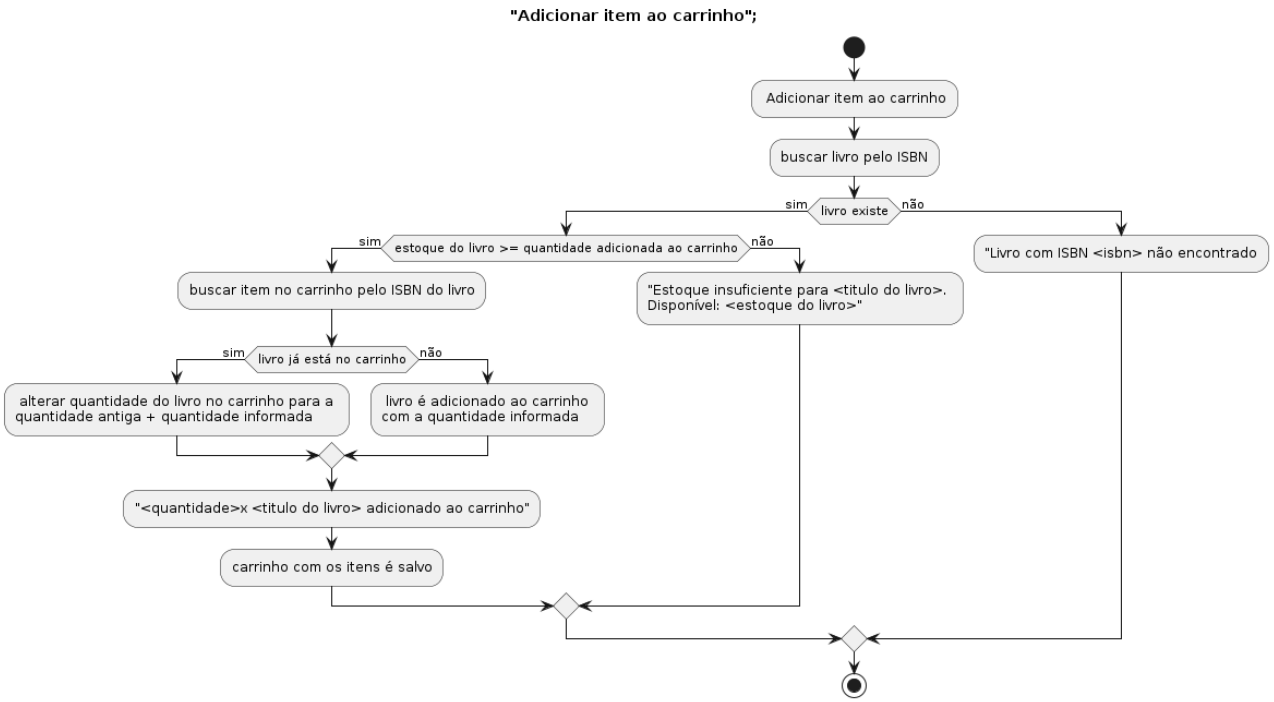
Código PlantUml do diagrama de classe:

```
@startuml classes
left to right direction
class Livro {
 - String isbn
 - String titulo
 - String autor
 - String categoria
 - double preco
 - int estoque
  + Livro(String isbn, String titulo, String autor, String categoria, double preco, int
estoque)
  + String getIsbn()
 + String getTitulo()
 + String getAutor()
 + String getCategoria()
 + double getPreco()
 + int getEstoque()

 + setPreco(double preco)
 + setEstoque(int estoque)

 + public String toString()
}
```

```
class ItemCarrinho {
 - Livro livro
 - int quantidade
  + ItemCarrinho(Livro livro, int quantidade)
  + Livro getLivro()
 + int getQuantidade()

 + setQuantidade(int quantidade)
  + double getSubTotal()
 + String toString()
}

class GerenciadorDeLivros {
 - List<Livro> livros;

 + GerenciadorDeLivros()
 'this.livros = DadosLivraria.carregarLivros(); representar no relacionamento
  + List<Livro> getTodosLivros()
 + Optional<Livro> buscarLivroPorIsbn(String isbn)
 + List<Livro> buscarLivrosPorTitulo(String termo)
 + void adicionarLivro(Livro novoLivro)
 + boolean atualizarLivro(String isbn, double novoPreco, int novoEstoque)
 + boolean removerLivro(String isbn)
}

class CarrinhoDeCompras {
 - List<ItemCarrinho> itens
 - GerenciadorDeLivros gerenciadorDeLivros

 + CarrinhoDeCompras(GerenciadorDeLivros gerenciadorDeLivros)
 'this.itens = DadosLivraria.carregarCarrinho(gerenciadorDeLivros.getTodosLivros());
representar no relacionamento

 + List<ItemCarrinho> getItens()
  + void adicionarItem(String isbn, int quantidade)
 + void removerItem(String isbn)
 + void atualizarQuantidadeItem(String isbn, int novaQuantidade)
 + double calcularTotal()
 + void exibirCarrinho()
 + void finalizarCompra()
}

class DadosLivraria {
 - {static} String ARQUIVO_LIVROS
 - {static} String ARQUIVO_CARRINHO

 + {static} List<Livro> carregarLivros()
 + {static} void salvarLivros(List<Livro> livros)
 + {static} List<ItemCarrinho> carregarCarrinho(List<Livro> todosLivros)
 + {static} void salvarCarrinho(List<ItemCarrinho> carrinho)
}
```

```
'1 Livro existe independente de pertencer a um ItemCarrinho, e pode pertencer a vários
(ou nenhum) ItemCarrinho
Livro "1" -- "0..*" ItemCarrinho : pertence a \n(associação)
'1 CarrinhoDeCompras pode possuir nenhum ou vários ItemCarrinho, que só podem pertencer a
um CarrinhoDeCompras,
'pois cada item tem o seu Livro e sua quantidade, não fazendo sentido que um ItemCarrinho
se repita
CarrinhoDeCompras "1" *-- "0..*" ItemCarrinho : pertence a \n(agregação)
'DadosLivraria é uma dependência do construtor de GerenciadorLivros
GerenciadorDeLivros --> DadosLivraria : (dependência)
'DadosLivraria é uma dependência do construtor de CarrinhoDeCompras
CarrinhoDeCompras --> DadosLivraria : (dependência)
CarrinhoDeCompras -> GerenciadorDeLivros : (dependência)
@enduml
```

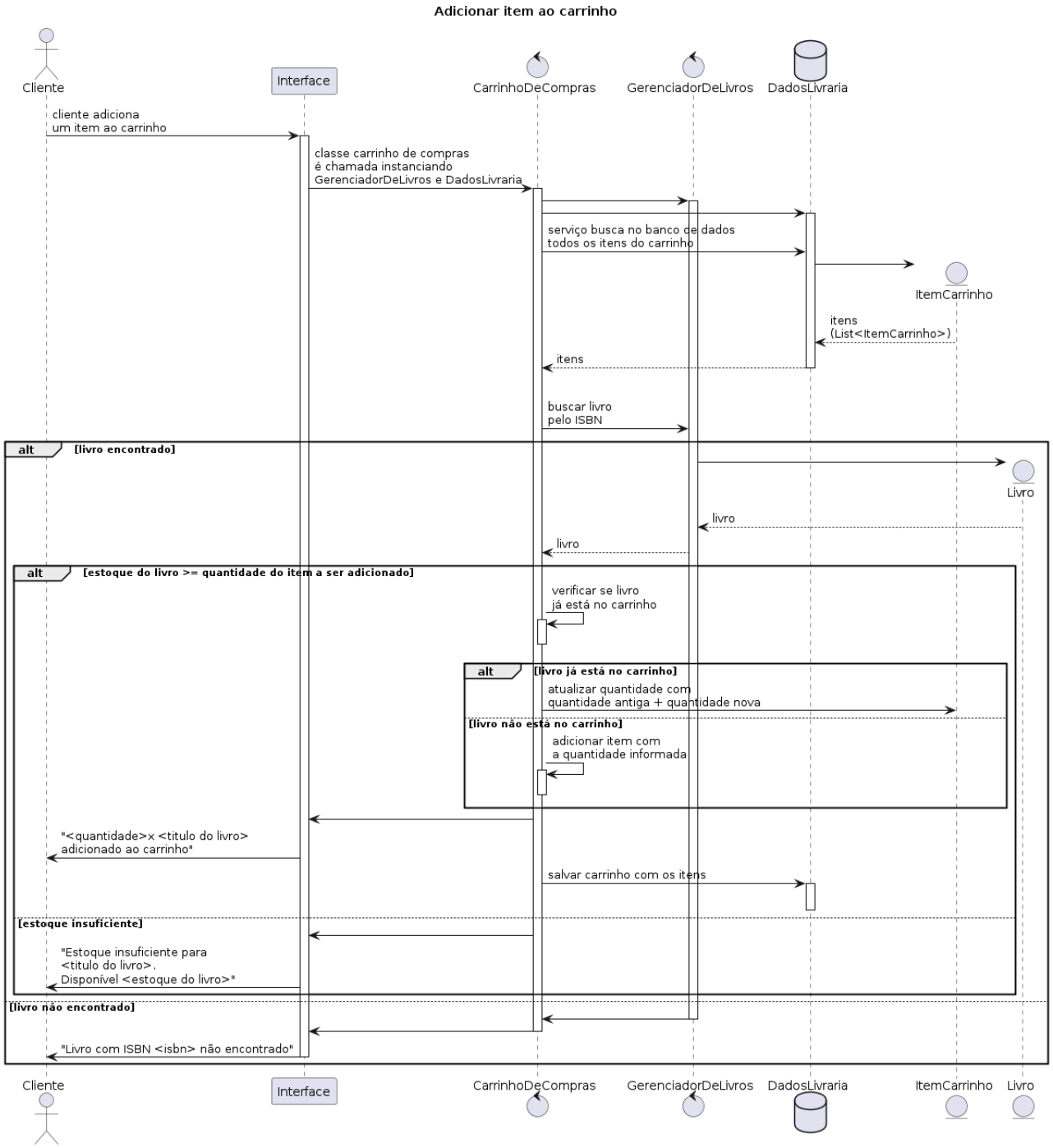
Diagrama de Atividades



Código PlantUml do diagrama de atividades

```
@startuml activity3
title "Adicionar item ao carrinho";
start
: Adicionar item ao carrinho;
:buscar livro pelo ISBN;
if (livro existe) then (sim)
 if (estoque do livro >= quantidade adicionada ao carrinho) then (sim)
   :buscar item no carrinho pelo ISBN do livro;
   if (livro já está no carrinho) then (sim)
     : alterar quantidade do livro no carrinho para a \nquantidade antiga + quantidade
informada;
   else (não)
     : livro é adicionado ao carrinho \ncom a quantidade informada;
   endif
   :"<quantidade>x <titulo do livro> adicionado ao carrinho";
   :carrinho com os itens é salvo;
 else (não)
   :"Estoque insuficiente para <titulo do livro>. \nDisponível: <estoque do livro>";
 endif
else (não);
 :"Livro com ISBN <isbn> não encontrado;
endif
stop
@enduml
```

Diagrama de Sequência



Código PlantUml do diagrama de sequência

```
@startuml sequence2
title "Adicionar item ao carrinho"
actor Cliente
participant Interface
control CarrinhoDeCompras
control GerenciadorDeLivros
database DadosLivraria

Cliente -> Interface : cliente adiciona \num item ao carrinho
'construtor da classe CarrinhoDeCompras
activate Interface
Interface -> CarrinhoDeCompras : classe carrinho de compras \né chamada instanciando
\nGerenciadorDeLivros e DadosLivraria
activate CarrinhoDeCompras
CarrinhoDeCompras -> GerenciadorDeLivros
activate GerenciadorDeLivros
CarrinhoDeCompras -> DadosLivraria
activate DadosLivraria
CarrinhoDeCompras -> DadosLivraria : serviço busca no banco de dados \ntodos os itens do
carrinho
create entity ItemCarrinho
DadosLivraria -> ItemCarrinho
ItemCarrinho --> DadosLivraria : itens \n(List<ItemCarrinho>)
DadosLivraria -->CarrinhoDeCompras : itens
deactivate DadosLivraria
|||
'lógica do método adicionarItem
CarrinhoDeCompras -> GerenciadorDeLivros : buscar livro \npelo ISBN
alt livro encontrado
 create entity Livro
 GerenciadorDeLivros -> Livro
 Livro --> GerenciadorDeLivros : livro
 GerenciadorDeLivros --> CarrinhoDeCompras : livro
   alt estoque do livro >= quantidade do item a ser adicionado
     CarrinhoDeCompras -> CarrinhoDeCompras : verificar se livro \njá está no carrinho
     activate CarrinhoDeCompras
     deactivate CarrinhoDeCompras
     alt livro já está no carrinho
       CarrinhoDeCompras -> ItemCarrinho : atualizar quantidade com \nquantidade antiga +
quantidade nova
     else livro não está no carrinho
       CarrinhoDeCompras -> CarrinhoDeCompras : adicionar item com \na quantidade
informada
       activate CarrinhoDeCompras
       deactivate CarrinhoDeCompras
     end
     CarrinhoDeCompras -> Interface
     Interface -> Cliente : "<quantidade>x <titulo do livro> \nadicionado ao carrinho"
     CarrinhoDeCompras -> DadosLivraria : salvar carrinho com os itens
     activate DadosLivraria
     deactivate DadosLivraria
```

```
   else estoque insuficiente
   CarrinhoDeCompras -> Interface
   Interface -> Cliente : "Estoque insuficiente para \n<titulo do livro>. \nDisponível
<estoque do livro>"
   end
else livro não encontrado
GerenciadorDeLivros -> CarrinhoDeCompras
deactivate GerenciadorDeLivros
CarrinhoDeCompras -> Interface
deactivate CarrinhoDeCompras
Interface -> Cliente : "Livro com ISBN <isbn> não encontrado"
deactivate Interface
end
@enduml
```