Atividade Acadêmica de Computação III
Sistema de Apoio ao Gerenciamento de Museus (SAGM)

Andressa Oliveira[1], **Mayara Marques**[2]

[1]Universidade Federal Rural do Rio de Janeiro (UFRRJ)
R. Governador Roberto Silveira S/N – Nova Iguaçu – Rio de Janeiro – RJ – Brasil

[2]Departamento de Ciência da Computação

dessaschweitzer@gmail.com, mmrosatab@hotmail.com

Resumo. *Para a construção de uma aplicação Web voltada para auxiliar a gestão de museus, de seus acervos e obras, foram utilizados padrões arquiteturais e testes automatizados vistos na disciplina Computação III. A principal característica é a divisão em 3 camadas, tendo como padrão arquitetural da camada de domínio o modelo de domínio simples e, para a camada de dados,portão de acesso aos dados em linha.*

1. Contexto da aplicação

O minimundo proposto relata a necessidade de um sistema responsável por realizar diversas tarefas pertinentes à criação e administração de museus. Para tal, foi utilizada a arquitetura em 3 camadas. O objetivo de aplicar a arquitetura em 3 camadas é isolar a lógica do domínio das atividades realizadas na camada de dados como, por exemplo, consultas, e das atividades realizadas na camada de apresentação. Esse isolamento é capaz de prover melhor legibilidade e manutenção do software, uma vez que isolando camada de domínio e camada de dados, é possível realizar refatoração sem graves impactos.

Em cada camada, foi definido um tipo de padrão a ser aplicado; tendo como propósito final a junção de ambos os padrões com as adaptações necessárias para funcionar de forma adequada. Dentro do contexto explicitado, os casos de uso implementados foram **ECU2)** Criar Usuário, **ECU5)** Criar Gestor, **ECU12)** Solicitar criação do Museu e **ECU13)** Criar Museu. As seções seguintes definem em etapas como foi feita a construção da aplicação.

2. Modelagem de dados

2.1. Modelo conceitual

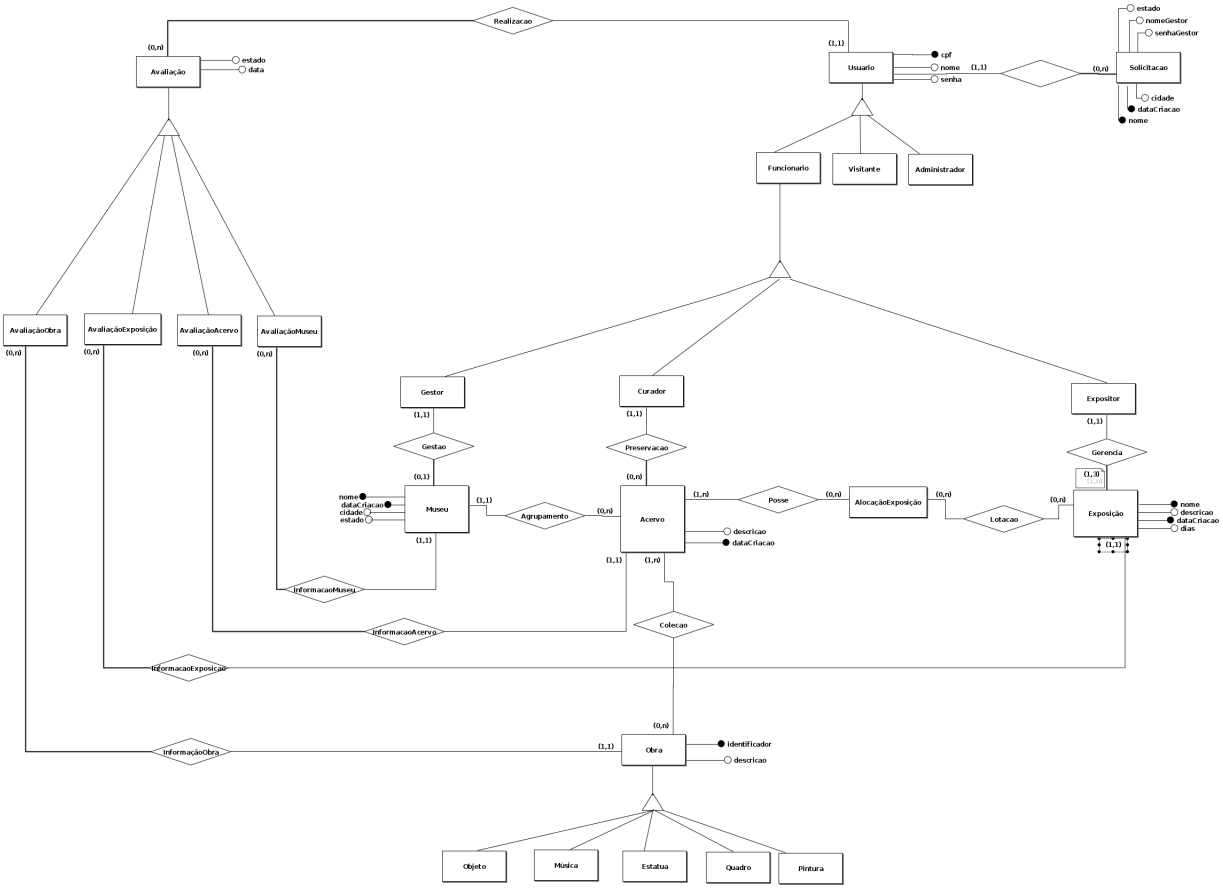

Figura 1. Modelo conceitual

2.2. Modelo lógico

Algumas decisões tiveram que ser tomadas para o modelo lógico em função do domínio da aplicação. Para implementar a especialização caracterizada pelas classes **Gestor e Administrador**, que estendem **Usuário**, únicas necessárias para o desenvolvimento dos casos de uso definidos, foi criada uma única tabela: **Usuário**; contendo todos os dados comuns a essas entidades. A esta tabela também foi adicionada uma coluna chamada **tipo** que serve como uma representação dos diferentes tipos de usuários que estão presentes no domínio da aplicação.

TABELAS

Usuario (cpf, nome, senha, tipo)
Avaliacao (estado, data, tipo)
Obra (identificador, descricao, tipo)
Museu (nome, dataCriacao, cidade, estado)

Acervo (dataCriacao, descricao)
Exposicao (nome, dataCriacao, descricao, dias)

RELACIONAMENTOS

Obra (identificador, descricao, tipo)
Avaliacao (cpf, estado, data)
cpf referencia Usuario

Usuario (cpf, nome, senha, tipo, nome, dataCriacao, cidade, estado)
(**Fusão de Tabelas:** Gestor → Museu)

Acervo (dataCriacao, descricao, cpf)
(**Adição de colunas:** Usuario → Acervo)
cpf referencia Usuario

Preservacao (cpf, dataCriacao)
(**Tabela própria:** Curador → Acervo)
cpf referencia Usuario
dataCriacao referencia Acervo

Exposicao (nome, dataCriacao, descricao, dias, cpf)
cpf referencia Usuario

AcervoExposicao (nome, dataCriacao, dataCriacao)
nome referencia Exposicao
dataCriacao referencia Exposicao
dataCriacao referencia Acervo

Colecao (identificador, dataCriacao)

Avaliacao (cpf, estado, data, tipo, nomeMu, dataCriacaoMu, dataCriacaoAc, nomeExp, dataCriacaoExp, identificadorObra)
(**Adição de colunas:** AvaliacaoMuseu → Museu, AvaliacaoAcervo → Acervo. . . etc)
cpf referencia Usuario
nomeMu referencia Museu
dataCriacaoMu referencia Museu
dataCriacaoAc referencia Acervo
nomeExp referencia Exposicao
dataCriacaoExp referencia Exposicao
IdentificadorObra referencia Obra

2.3. Normalização

2.3.1. 1FN

Aplicando a 1FN ao modelo, a tabela
Avaliacao (cpf, estado, data, tipo)
cpf referencia Usuario

AvaliacaoMuseu (cpf, nomeMu, dataCriacaoMu)
cpf referencia Usuario
nomeMu referencia Usuario
dataCriacaoMu referencia Usuario

AvaliacaoAcervo (cpf,dataCriacaoAc)
cpf referencia Usuario
dataCriacaoAc referencia Acervo

AvaliacaoExp (cpf, nomeExp, dataCriacaoExp)
cpf referencia Usuario
nomeExp referencia Exposicao
dataCriacaoExp referencia Exposicao

AvaliacaoObra (cpf, identificadorObra)
cpf referencia Usuario
IdentificadorObra referencia Obra

2.3.2. 2FN e 3FN

Após a análise foi concluído que o modelo já se encontrava na 2FN e 3FN.

2.4. Modelo físico

3. Implementação

As seções abaixo apresentam as particularidades de cada padrão dentro do contexto da aplicação, bem como a forma com que foi realizada a sua combinação.

3.1. Modelo de Domínio Simples

Em (FOWLER, 2002), diz-se que o *Modelo de Domínio Simples* é caracterizado por ser um modelo onde os objetos do domínio incorporam tanto o comportamento quanto os dados.

3.2. Portão de Acesso aos Dados

No *Portão de Acesso aos Dados*, padrão de arquitetura utilizado na camada de dados, existe uma entidade responsável por fazer buscas por elementos do banco de dados e uma responsável por realizar alterações diretamente nesse banco, como atualizar, inserir

e deletar. Ambas são denominadas na literatura como *finder* e *gateway*. Segundo Fowler (FOWLER, 2002), elas estão relacionadas entre si e essa relação será apresentada na seção 5.

4. Diagramas de Sequência

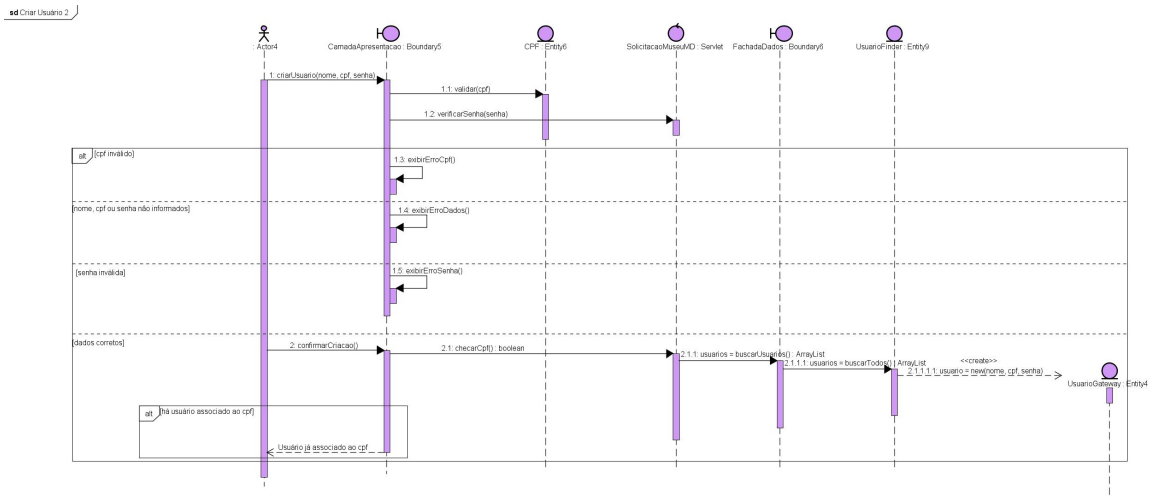

Figura 2. Diagrama do caso de uso Criar Usuário

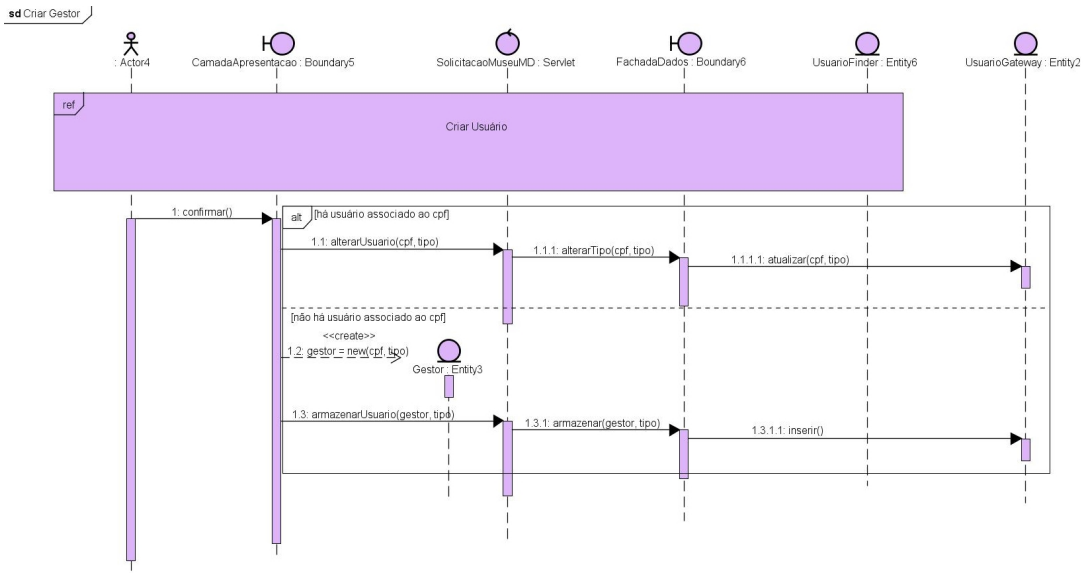

Figura 3. Diagrama do caso de uso Criar Gestor

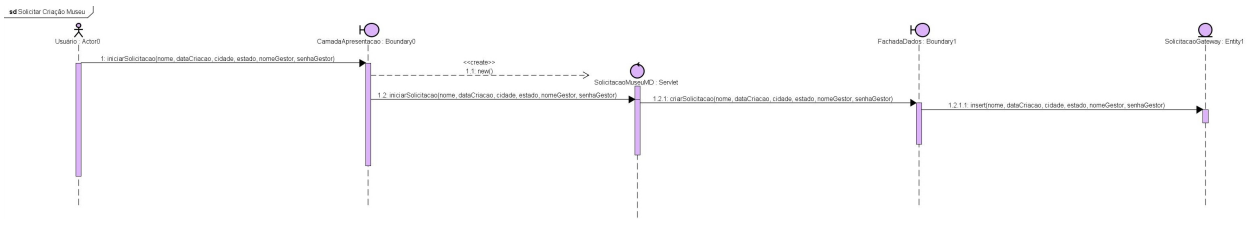

Figura 4. Diagrama do caso de uso Solicitar criação museu

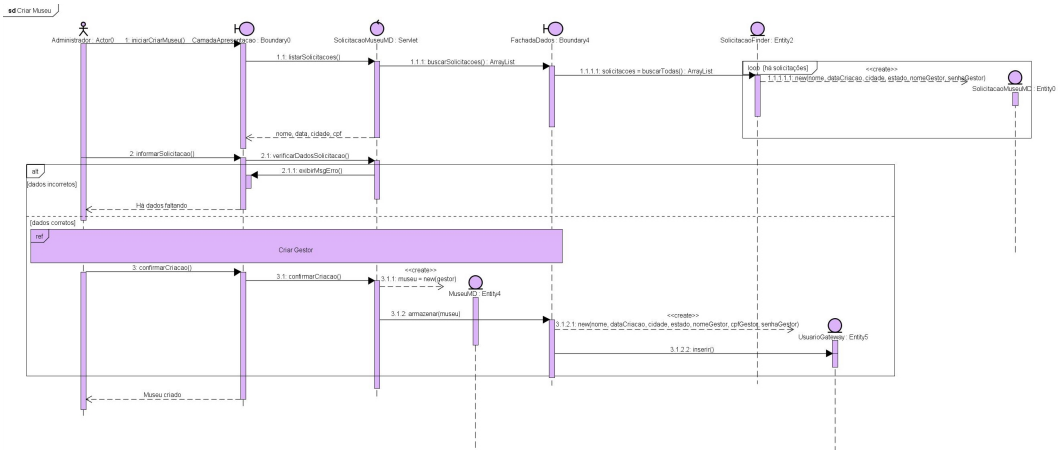

Figura 5. Diagrama do caso de uso Criar Museu

5. Diagrama de classes final

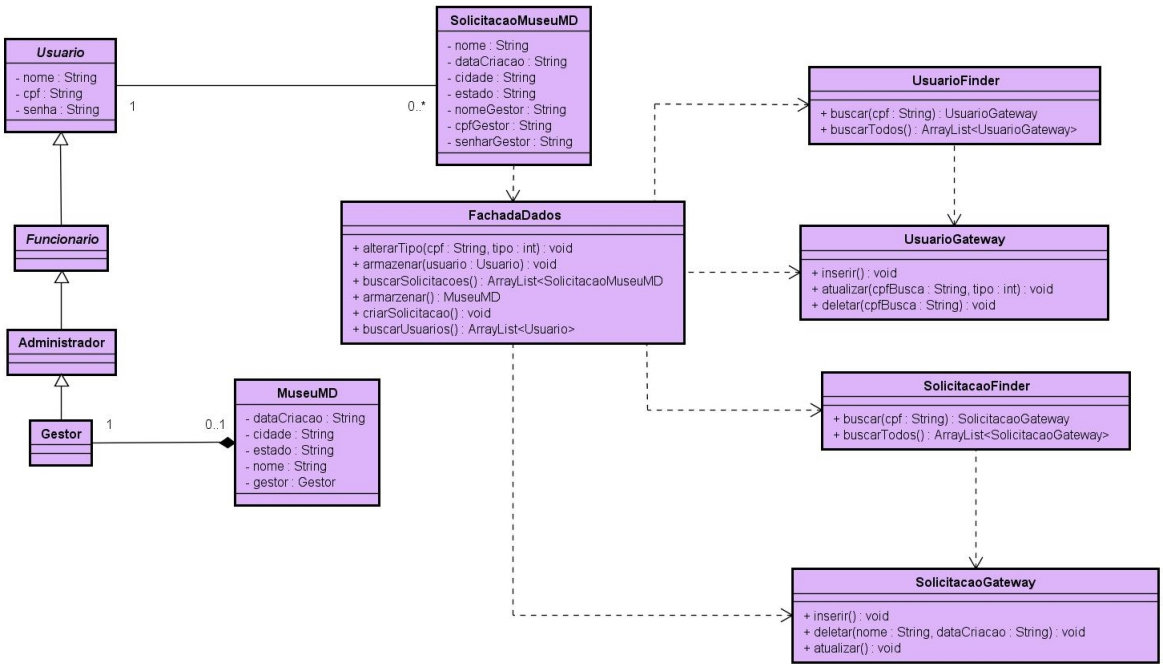

Figura 6. Diagrama de Classes Final

6. Testes

6.1. Testes Funcionais

Os testes funcionais foram realizados sem o auxílio de ferramentas de automatização.

- Criar Usuário - Caso de teste <(nome, cpf, senha), resultado>
  - **CT01** <("NomeValido","CpfValido","SenhaValida"), válido>
  - **CT02** <(,,), inválido - gerar notificação "preenchimentos dos campos nome, cpf e senha é obrigatório»

- **CT03** <("NomeValido","CpfInvalido","SenhaValida"), inválido - gerar notificação "cpf inválido»
- **CT04** <("NomeValido","CpfValido","SenhaValida"), inválido - gerar notificação "cpf já está associado a um usuário»
- **CT05** <("NomeValido","CpfInvalido","SenhaValida"), inválido - gerar notificação "cpf é inválido»
- **CT06** <("NomeValido","CpfValido","SenhaInvalida"), inválido - gerar notificação "senha deve ter seis dígitos»

Referências

FOWLER, M. *Patterns of Enterprise Application Architecture*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2002. ISBN 0321127420.