Super Trunfo

Relatório

Breno Candido Conrado
Curso de Engenharia Mecatrônica
Universidade Federal de Santa Catarina
Joinville, Brasil
breno.caandido@gmail.com

***Resumo*—Este relatório é uma documentação do projeto final referente à disciplina Programação III do curso de Engenharia Mecatrônica.**

***Palavras-chave*—engenharia; programação; documentação; projeto**

I. INTRODUÇÃO

Como projeto final da disciplina Programação III, foi feito um jogo de cartas utilizando a linguagem de programação C++ e o software Qt para a interface gráfica de usuário (GUI).

O propósito do projeto foi, no início, criar uma estrutura (framework) para que todos os participantes da disciplina pudessem ter em comum, e desenvolver seus jogos à partir desta base. Este relatório se aprofundará na documentação do jogo Super Trunfo.

II. O JOGO

A. Regras do Jogo

O Super Trunfo é um jogo de cartas que consiste em tomar cartas de outros jogadores para si mesmo a fim de ganhar através de competições entre os atributos de cada carta.

Cada conjunto possui um tema principal, ex carros, motos, animais, heróis de quadrinhos. Toda carta possui algo ou alguém à representando, além sua lista de atributos. A figura 1 é um exemplo de Super Trunfo de equipes de Fórmula 1.

Fig. 1. Exemplo de carta presente no Super Trunfo de equipes de Fórmula 1[1].

O jogo suporta de dois a dez jogadores. Cada rodada inicia com o primeiro jogador escolhendo qual atributo deseja comparar com seus adversários. Após isso, o jogador com o maior valor no atributo escolhido vence a rodada e poderá escolher a próxima característica desejada.[1]

[1] Disponível em <http://blogdocapelli.grandepremio.uol.com.br/category/jogos/>

No caso de empate, os jogadores que empataram devem competir novamente, competindo pelas cartas da rodada em andamento, junto com as da rodada anterior.

Todo baralho possui uma carta Super Trunfo, a qual possui os maiores valores em todas as características. Esta pode ser vencida apenas por cartas xA (presente no canto superior direito).

B. Implementação do Framework

Primeiramente, foi feito um diagrama UML (Unified Modelling Language) para representar como seria a estrutura geral do projeto, mostrando classes e as relações entre e elas. O modelo final pode ser visto na figura 2. Observe as relações de agregação entre as classes, principalmente entre a classe Jogo e o restante.

Após isso, foram implementadas as classes começando pelas classes que não dependem de outras para existir, como a classe Carta e, em seguida, Baralho e Jogador.

Para que o framework aceite cartas diferentes, criadas fazendo uso de polimorfismo, o restante das classes acima desta foram implementadas em forma de template. Entretanto, a construção básica continua usando as classes base, caso não haja necessidade de derivar. Este conceito foi aplicado também para Jogador, permitindo aceitar diferentes jogadores derivados.

No fim, a classe Jogo é responsável pela criação do restante das classes, com exceção de Regra, a qual é passada como parâmetro do construtor de jogo.

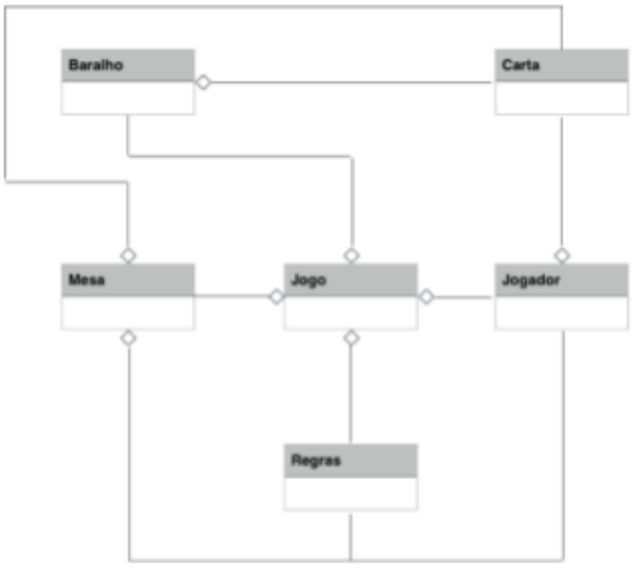

Fig. 2. Diagrama UML utilizado

C. Desenvolvendo o Jogo

Com o framework feito, foi iniciado o desenvolvimento do Super Trunfo. Como decisão de projeto, o jogo suporta apenas dois jogadores, sendo um usuário e o computador.

Sabendo que o maior diferencial deste jogo em relação ao restante é fato de sua carta não ser aquela vista e baralhos convencionais, o primeiro passo foi derivar a classe Carta, base, para uma mais específica: Carta_trunfo. Esta nova classe permite que haja atributos relacionados a cada objeto da classe Carta_trunfo, os quais são definidos de acordo com com seu número e letra (chamada de naipe, devido ao modo como foi projetado o framework). A lista de atributos está definida em um std::vector privado, sendo acessada dentro do construtor de Carta_trunfo. A figura 3 mostra a declaração da classe.

Resolvida essa parte, foi derivada a classe Jogo para poder implementar métodos específicos ao jogo Super Trunfo, além de sobrescrever outros, deixando-os adequados ao contexto no qual são utilizados. Os métodos principais dessa classe são o receber_jogada, realizar_jogada e jogar; estes ditam o progresso do jogo, sendo o método, recebendo entradas do usuário e realizando-as de acordo com a regra. A figura 4 mostra como foi organizada a classe; uma grande parte são métodos privados que auxiliam na execução do programa.

```
class Carta_trunfo : public p3::Carta {
public:

    Carta_trunfo(int numero, Carta::Naipe naipe, bool frente = false);

    int participacao_gps();
    int titulos_mundiais_construtores();
    int vitorias();
    int pole_positions();
    int gps_com_podios();
    int index();

private:

    int _participacao_gps = 0;
    int _titulos_mundiais_construtores = 0;
    int _vitorias = 0;
    int _pole_positions = 0;
    int _gps_com_podios = 0;

    using Atributos = std::vector<int>;
    std::vector< Atributos > lista_atributos {
```

Fig. 3. Classe Carta_trunfo

```
class Jogo_trunfo : public p3::JogoBasico<Carta_trunfo>{
public:
    Jogo_trunfo(p3::Regra *regra, std::vector<std::string> nomes, Trunfo
    ~Jogo_trunfo();
    void jogar();
    Jogada recebe_jogada();
    void realiza_jogada(Jogada jogada);
    void determina_acao_jogador_vencedor(const size_t jogador_vencedor);
    bool fim_jogada() override;
    int ultimo_jogador_vencedor();
    void imprime_numero_cartas();

private:
    Carta_trunfo carta_super_trunfo;
    size_t _ultimo_jogador_vencedor = 2;

    void verifica_jogadores_derrotados() override;
    void move_carta_jogador_vencedor(const size_t jogador_vencedor);
    void move_carta_empate();
    int checa_super_trunfo();
    size_t jogador_oponente();
    void imprime_atributo_escolhido(Jogada jogada);
    int carta_jogador_0_index();
    int carta_jogador_1_index();

    TrunfoUI *_ui;
```

Fig. 4. Classe Jogo_trunfo

O restante das classes não necessitou mudanças, entretanto foram criadas Jogada e InteligenciaArtifical. Jogada possui um enum class para listar os possíveis escolhas a serem feitas durante o jogo, referindo-se aos atributos da carta. InteligenciaArtificial inclui um simples método de geração de números aleatórios para quando for a vez do computador jogar.

O fluxo básico de execução consiste em:

1. Recebe atributo escolhido pelo jogador atual;
2. Compara as cartas dos jogadores;
3. Move as cartas ao vencedor ou monte de empate;
4. Faz as alterações necessárias em Jogo_trunfo;

D. Interface Gráfica

A interface gráfica foi feita fazendo uso do software de desenvolvimento Qt, junto a seu IDE (Integrated Development Environment) QtCreator.

A janela principal, TrunfoUI, derivada de QMainWindow, contém toda a parte gráfica do jogo, sendo ela separada em vários QWidgets, os quais permitem maior controle sobre o posicionamento e geometria das diversas partes incluídas na janela principal. Entradas do usuário se dão na forma de botões (QPushButton) e comunicação através de caixas de diálogo e textos na tela.

A classe TrunfoUI comporta todos os métodos utilizados dentro da execução jogo, sendo as diversas outras classes criadas para facilitar a divisão em telas e se comunicar com o usuário através de caixas de diálogo. A figura 5 mostra a tela inicial do jogo.

Fig. 5. Tela inicial

Cada bloco da interface é um widget; como, por exemplo a parte central, formada por título, um espaçador para separar o texto e os botões. Esta separação também permite esconder, mostrar, ativar e desativar blocos quando necessário, além de poder conectar vários QWidget a outro de maior escala, controlando todos ao mesmo tempo.

A tela de jogo (Fig. 6) é formada pelos botões, os textos que mostram ao usuário o estado atual do jogo e, mais importante, as cartas atuais. O modo como foi feita a transição de uma tela para outro foi escondendo o QWidget da tela inicial e mostrando o da tela de jogo. Durante o jogo, os botões são habilitados apenas quando for a vez do usuário de jogar e a carta do computador é mostrada apenas no fim de cada rodada.

Fig. 6. Tela de jogo

III. Opinião Geral

Terminado o projeto, observa-se o progresso feito à partir do início, desenvolvendo a framework em grupo, até o fim, aplicando tudo que foi construído para a criação de cada jogo.

Finalmente, o aprendizado de noções de desenvolvimento coletivo e o aprimoramento das habilidades de programação, além de poder estender o conhecimento de outros softwares que virão a ser de uso futuramente trouxe benefícios incomparáveis.

Referências

[1] Gametrack, Regras do jogo Super Trunfo. Disponível em <http://www.gametrack.com.br/jogos/cancan/instrucoes/supertrunfo.asp>. Acessado em 28 de junho de 2017.

[2] The Qt Company, Qt Documentation. Disponível em <http://doc.qt.io/>. Acessado em 28 de junho de 2017.