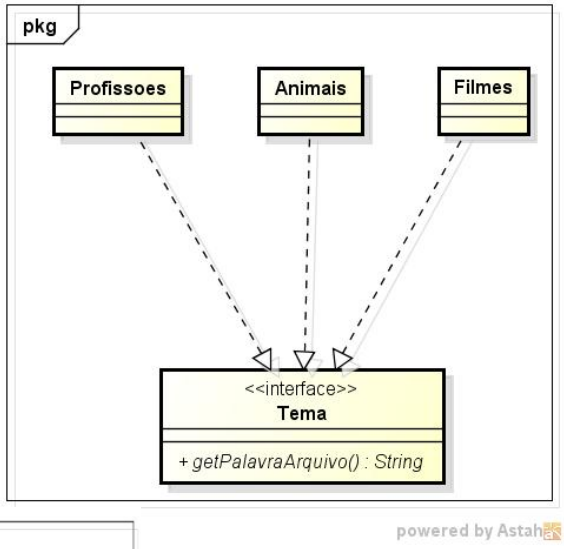
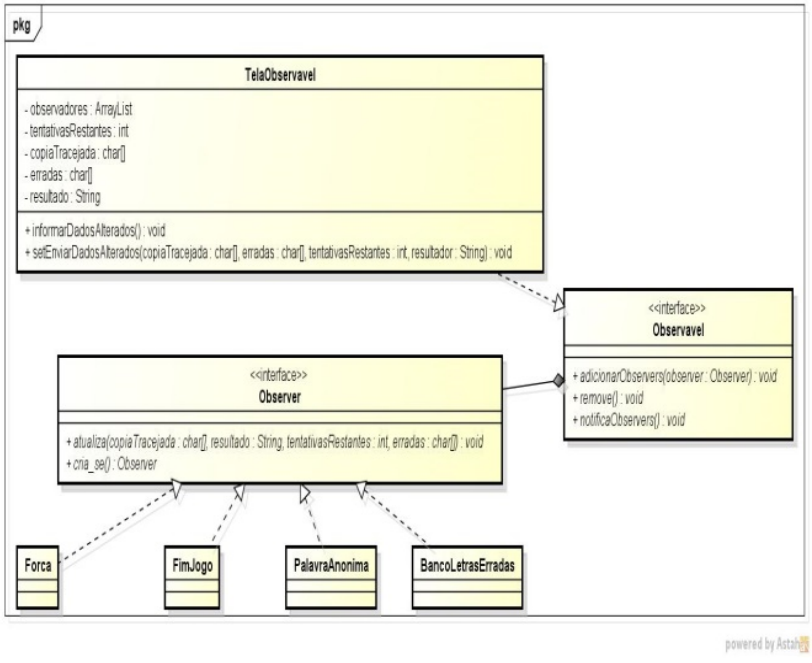
Universidade Federal de Goiás
Goiânia, 20 de junho de 2015.
Alunos: Victor Vidal / Weryques Santos
Disciplina: Projeto de Software
Dr. Professor Cássio Rodrigues

**Relatório**

Este relatório tem o intuito de explicar o Diagrama de classes criado para o jogo da forca e busca uma compreensão a respeito dos padrões de projetos usados e porque foram usados. O diagrama será dividido em partes de modo a facilitar a compreensão do mesmo.

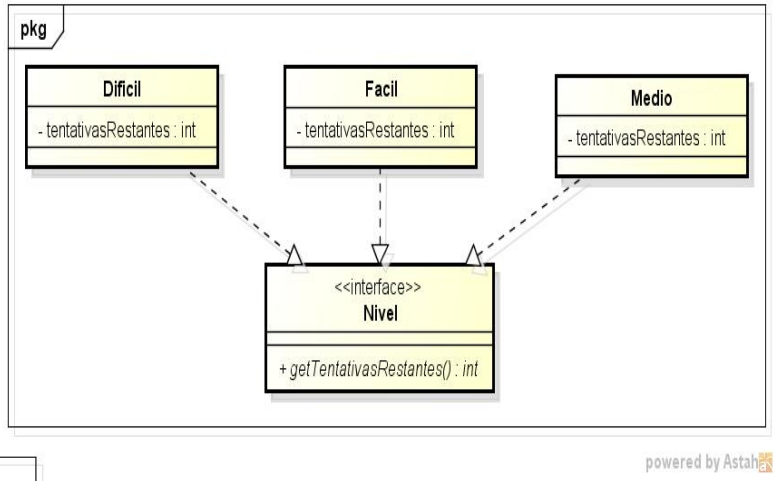
Na imagem 1, é feita a divisão dos possíveis temas a serem escolhidos pelo usuário.

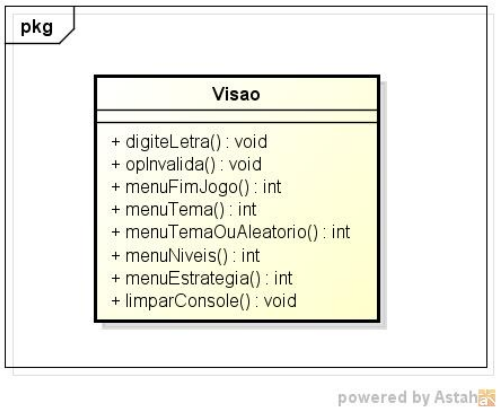


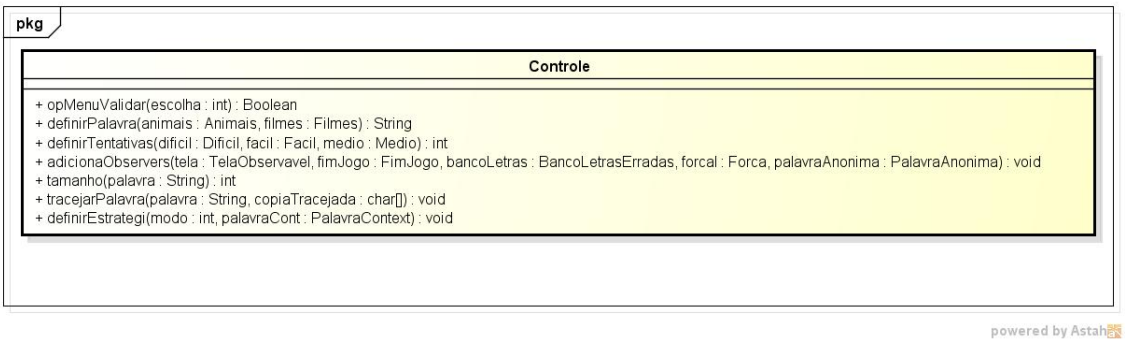

Na imagem de número 2, podemos ver o padrão Observer, um dos padrões que foram utilizados para implementar o projeto. O mesmo avisa ao interessado sempre que há atualizações através do método atualiza. É implementado por forca (número de chances restantes), Fim de jogo (Mensagem de vitória ou derrota), Palavra anônima (Representam as letras acertadas, seria algo como "Banco de acertos"), e por último Banco de letras Erradas, que tem como objetivo informar ao usuário as letras que o mesmo já inseriu e estão incorretas.

A seguir, na imagem 3 temos a escolha dos níveis. Aqui é onde o usuário decide em qual nível vai jogar (Isso influenciará diretamente na quantidade de "chances" o mesmo terá para acertar a palavra, pode ser observado no método implementado dentro das classes , denominado de tentativas restantes).



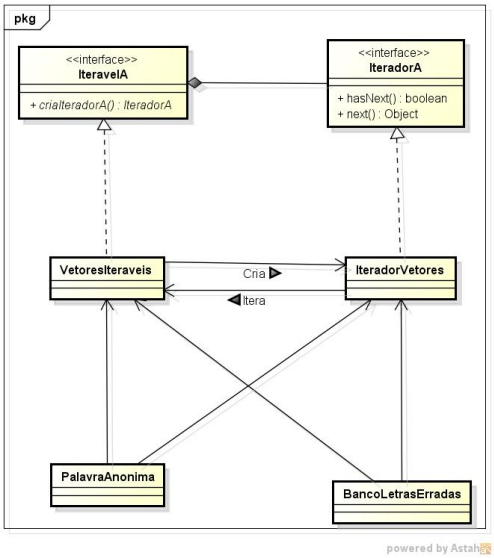



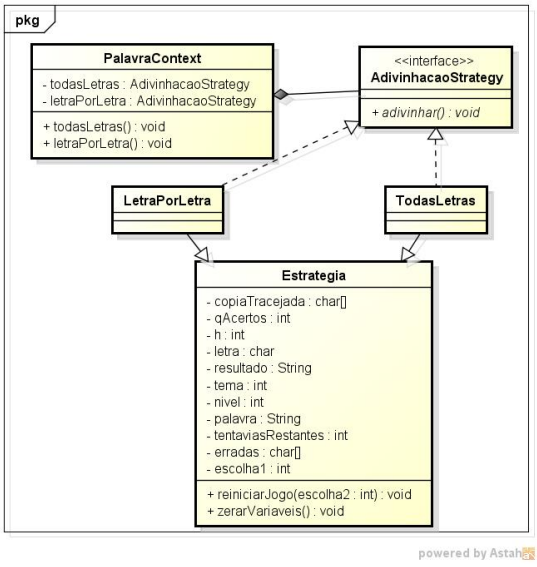

Na quarta imagem temos a classe visão, ela é responsável pela Impressão de todos os menus do jogo, por limpar o console, informar que a opção está incorreta e solicitar a digitação de uma letra para o acerto da palavra, assim como sugerem os métodos implementados.




Na imagem de número 5 podemos ver o método que interage com várias outras classes dentro do código. Temos como alguns métodos o DefinirTentativas que está ligado com a interface nível citada acima, AdicionaObservers que também está ligado com o padrão Observer citado anteriormente dentre outros.

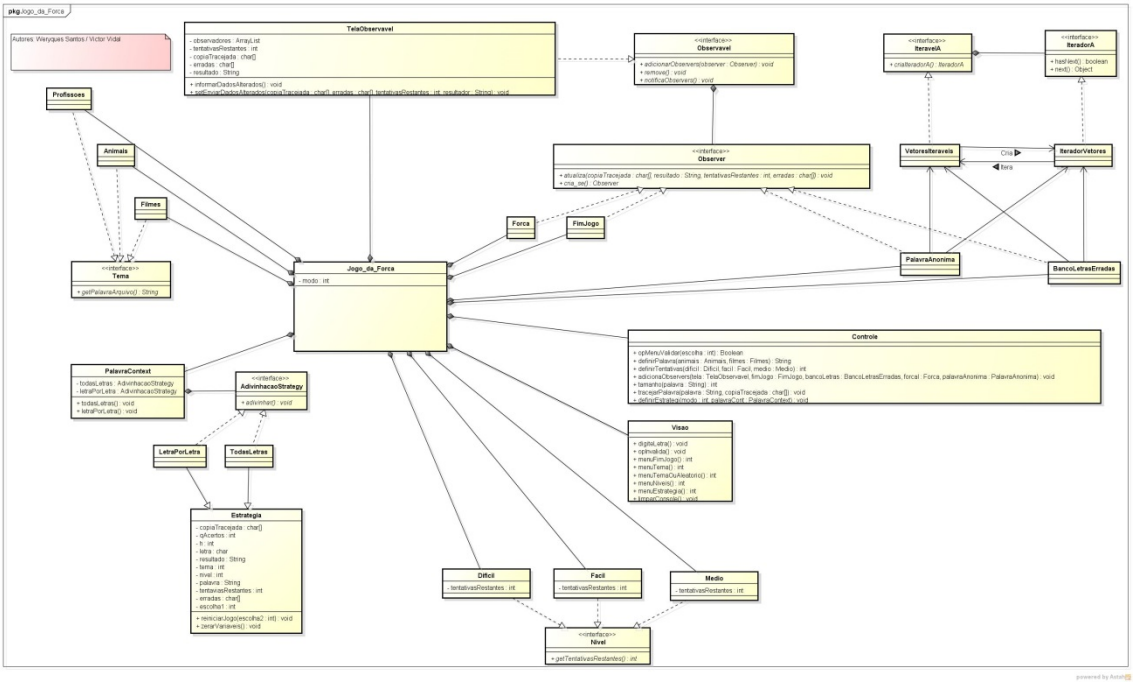
Na sexta imagem temos o padrão Iterator, ele foi utilizado pelos Observers Palavra anônima e Banco de Letras erradas pois o mesmo não deve ter conhecimento de como os Vetores errados e palavra anônima são iterados, eles devem apenas saber o resultado.






Na imagem 7 temos o padrão Strategy onde as classes LetraPorLetra e TodasLetras implementam a interface AdivinhacaoStrategy. O Strategy na classe LetraPorLetra Verifica se a letra digitada existe na palavra e se ainda não foi revelada, caso não tenha sido a variável qacertos é incrementada e o local tracejado recebe a letra, e caso já tenha sido revelada ocorre um break, pois somente a primeira ocorrência da letra é revelada.

Quanto a classe TodasLetras será verificado a quantidade de letras acertadas de modo a dizer ao usuário que o mesmo venceu, verifica a quantidade de erros do jogador pra informa-lo em caso de derrota e verifica se a letra digitada existe na palavra e se ainda não foi revelada.




Por último temos o diagrama final mostrando todas as classes que compõe Jogo_da_Forca.

## Padrões Utilizados

Foram utilizados três padrões de projeto em nosso trabalho: Observer, Strategy e Iterator. Vejamos uma breve explicação de cada um deles e como foram utilizados em nosso trabalho.

Observer

O padrão Observer foi aplicado às classes **TelaObservavel**, **Forca**, **FimJogo, PalavraAnonima**, **BancoLetrasErradas**. Onde **TelaObservavel** é o Observável Concreto e **FimJogo, PalavraAnonima, BancoLetrasErradas, Forca** são os Observadores Concretos, esse implementam além do método padrão da classe Observer (*atualiza()*), o método *cria_se()* que retorna uma instancia da própria classe.

Strategy

O padrão Strategy foi aplicado à classe **PalavraContext** que é o contexto da estratégia, às classes **LetraPorLetra** e **TodasLetra** que implementam a *interface* (Strategy) **AdivinhacaoStrategy** e herdam da classe **Estrategia** métodos e atributos comuns.