Planos de Arquitetura, Gerência de Configuração e de Implantação

Informações sobre o documento

Sumário do documento

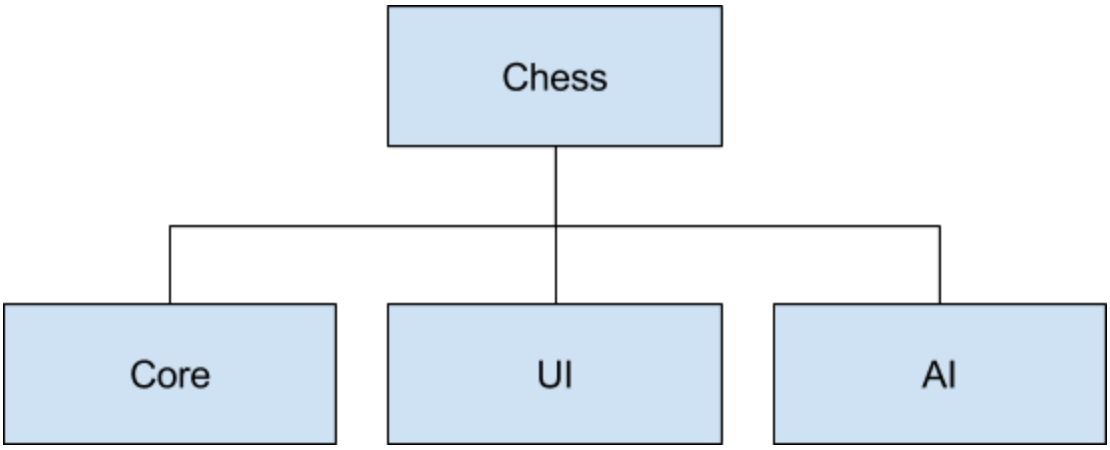
Chess	*Nome*
Nome do componente	*Planos de Arquitetura, Gerência de Configuração e de Implantação*
Autores do documento	*Beatriz Loza, Igor Martire*
Data	*01/05/2017*
Número de páginas	*3*

Sumário de alterações do Documento

Versão	Data	Descrição	Responsável
0.1	*01/05/2017*	*Primeiro esboço do documento criado*	*Igor Martire*
0.2	*DD/MM/AAAA*	*<Documento revisado e editado>*	*Nome*
1.0	*DD/MM/AAAA*	*<Aprovado>*	*Nome*

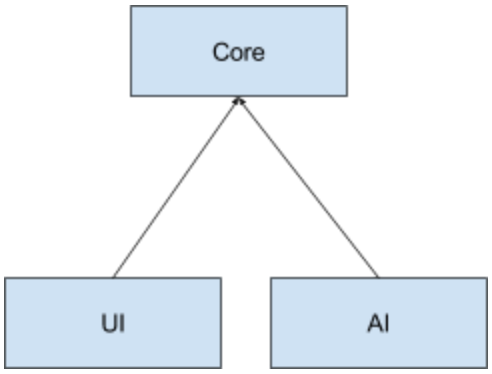
1 Plano de Arquitetura

1.1 Visão estrutural



A arquitetura do sistema é organizada em três pacotes, cujas integrações compõem o sistema como um todo. O pacote Core contém todas as funcionalidades relacionadas à lógica do jogo de xadrez: movimentação de peças, identificação de xeque e xeque-mate, armazenamento do estado do tabuleiro e do jogo. O pacote de UI (user interface) por sua vez traz uma representação gráfica do jogo que acontece no pacote Core. Por fim, o pacote de AI engloba a parte de inteligência artificial que é responsável por determinar a movimentação do jogador-máquina quando em seu turno.

1.2 Visão de dependências



Em termos de dependências, temos que as funcionalidades do Core são isoladas das funcionalidades das demais partes do sistema. Já os pacotes de UI e AI dependem do pacote Core para que possam funcionar.

1.3 Arquitetura técnica

Já na parte técnica, temos que o software desenvolvido deve ser executável em plataformas Windows e Mac com a versão 3.6 do Python instalada, bem como o pacote de depedência PyGame em sua versão 1.9.3.

2 Plano de Gerência de Configuração

A equipe utilizará a ferramenta git para gerência de configuração e controle de versionamento. Como repositório compartilhado, será utilizado o GitHub. Como ferramenta de integração contínua será utilizado o Travis CI com a seguinte configuração:

```
language: python
python:
  - "3.6"
install:
  - pip install -r requirements.txt
script:
  - pep8 chess/ tests/
  - python -m pytest tests/
```

Como modelo de ramificação será utilizado o modelo proposto por Vincent Driessen (disponível em: http://nvie.com/posts/a-successful-git-branching-model/). Estendemos o modelo para a criação de ramos docs/<issue> para os ramos que venham a adicionar documentações

do projeto, tendo em vista que as mesmas são armazenadas no mesmo repositório onde o código se encontra.

Como estrutura de diretórios e arquivos, o padrão adotado é:

├── **CONTRIBUTING.md** - documento com guidelines de contribuição com o projeto
├── **LICENSE** - licença de distribuição do software
├── **README.md** - descrição do projeto
├── **chess** - pacote raiz do sistema
│ ├── **ai** - pacote de inteligência artificial
│ ├── **core** - pacote com as funcionalidades fundamentais do jogo de xadrez
│ └── **ui** - pacote de interface gráfica com o usuário
├── **docs** - diretório para os arquivos de documentação do projeto
├── **requirements.txt** - arquivo usado para listar as dependências do projeto
├── **tests** - diretório para os códigos de teste automatizados

3 Plano de Implantação

A instalação do software deve ser fácil para os usuários finais. Assim, a implantação será através da execução de um arquivo .exe na plataforma Windows e da execução de um arquivo .app na plataforma Mac.

Além disso, haverá a opção de executar o código em qualquer sistema que possua o Python na versão 3.6 através da obtenção do pacote pelo PyPI e instalação feita pelo arquivo setup.py conforme instruções contidas no arquivo README que estará junto ao pacote.