INTEGRAÇÃO DE TÉCNICAS DE SISTEMAS DISTRIBUÍDOS APLICADA AO DESENVOLVIMENTO DE UM AMBIENTE PARA O JOGO DE XADREZ

Acadêmico: Antonio Carlos Bambino Filho
Orientador: Prof. Marcel Hugo

SUMÁRIO

- Introdução;
- Objetivos;
- Fundamentação teórica;
- Desenvolvimento;
  - Especificação;
  - Implementação
- Conclusão;
- Extensões;

INTRODUÇÃO

- *Rich Internet Applications* (RIAs);
- Java para aplicações distribuídas;
- Webservices;
- Java Web Start;
- Xadrez;

OBJETIVOS

- Desenvolver um ambiente de jogo via web para que pessoas possam jogar xadrez e jogos derivados deste;
- Explorar as funcionalidades da linguagem Java para implementar as regras;
- Gerenciar os jogos com um webservice Java, utilizando a API JAX;
- Integrar webservice, Java Web Start, servidor web e banco de dados;

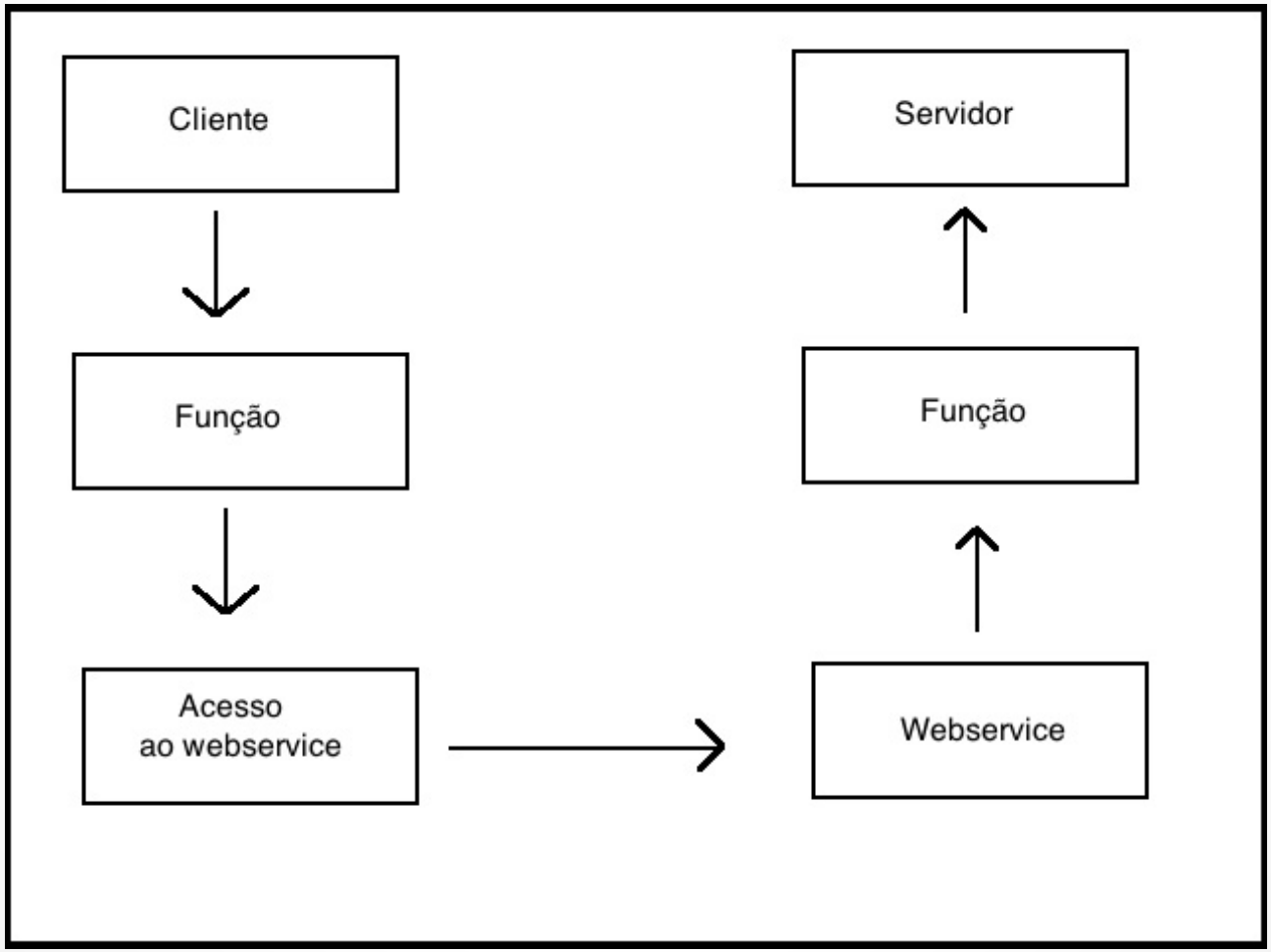
FUNDAMENTAÇÃO TEÓRICA

- Webservices;
  - JAX
- Java Web Start;
- Servidor Web;
  - GlassFish

Webservices

- Service-Oriented Architecture (SOA);
- Remote Procedure Call (RPC);

Webservices



Webservices

- WSDL – Definição em XML das funções;
- SOAP – protocolo para transmissão das mensagens;

Webservices

```
<wsdl:binding name="nmtoken" type="qname">*
    <wsdl:documentation .... />?
    <-- extensibility element --> *
    <wsdl:operation name="nmtoken">*
       <wsdl:documentation .... /> ?
       <-- extensibility element --> *
       <wsdl:input> ?
           <wsdl:documentation .... /> ?
           <-- extensibility element -->
       </wsdl:input>
       <wsdl:output> ?
           <wsdl:documentation .... /> ?
           <-- extensibility element --> *
       </wsdl:output>
       <wsdl:fault name="nmtoken"> *
           <wsdl:documentation .... /> ?
           <-- extensibility element --> *
       </wsdl:fault>
    </wsdl:operation>
</wsdl:binding>

<wsdl:service name="nmtoken"> *
    <wsdl:documentation .... />?
    <wsdl:port name="nmtoken" binding="qname">
       <wsdl:documentation .... /> ?
       <-- extensibility element -->
    </wsdl:port>
    <-- extensibility element -->
</wsdl:service>
```

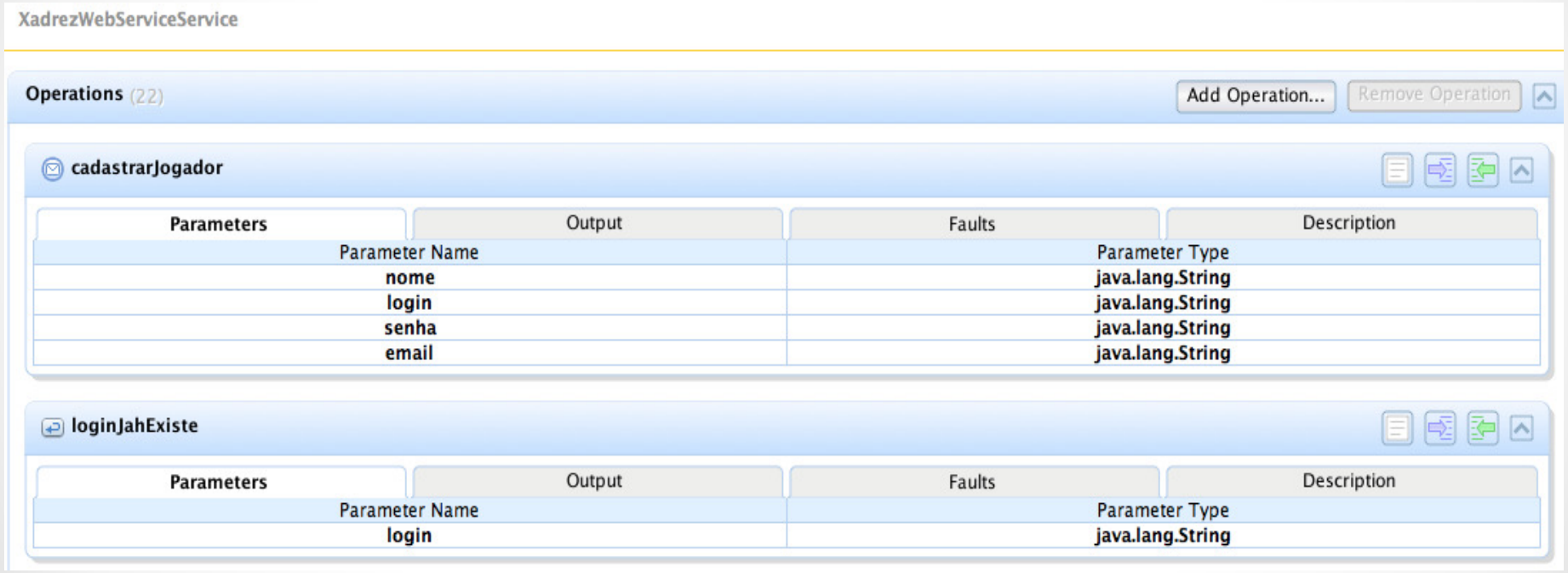
Webservices

```
POST /rpcrouter HTTP/1.1
Host: 127.0.0.1
Content-Type: text/xml; charset=utf-8
Content-Length: 559
SOAPAction: "http://mauricio.com"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
xmlns:xsi="Schema-Instance"
xmlns:xsd="Schema"
xmlns:soap="Envelope">
<soap:Body>
<Converte xmlns="http://conv.com.br">
<Valor>5</Valor>
<De>DEC</De>
<Para>BIN</Para>
</Converte>
</soap:Body>
</soap:Envelope>
```

Webservices - JAX

- Biblioteca que traz velocidade e simplificação;
- Abstração do SOAP e do WSDL;
- Limitação: tipos abstratos de dados;
-

JAX


XadrezWebServiceService
Operations (22)
Add Operation...
Remove Operation
cadastrarJogador
Parameters
Output
Faults
Description
Parameter Name
Parameter Type
nome
java.lang.String
login
java.lang.String
senha
java.lang.String
email
java.lang.String
loginJahExiste
Parameters
Output
Faults
Description
Parameter Name
Parameter Type
login
java.lang.String

Java Web Start

- Interface gráfica;
- Armazenado em memória cache;
- Versão mais atualizada do servidor;
- JNLP – XML verificador responsável;

Java Web Start

```
<?xml version="1.0" encoding="utf-8"?>
<!-- JNLP File for SwingSet2 Demo Application -->
<jnlp
  spec="6.0+"
  codebase="http://my_company.com/jaws/apps"
  href="swingset2.jnlp">
  <information>
    <title>SwingSet2 Demo Application</title>
    <vendor>Sun Microsystems, Inc.</vendor>
    <homepage href="docs/help.html"/>
    <description>SwingSet2 Demo Application</description>
    <description kind="short">A demo of the capabilities
    of the Swing Graphical User Interface.</description>
    <icon href="images/swingset2.jpg"/>
    <icon kind="splash" href="images/splash.gif"/>
    <offline-allowed/>
    <association mime-type="application-x/swingset2-file"  extensions="swingset2"/>
    <shortcut online="false">
      <desktop/>
      <menu submenu="My Corporation Apps"/>
    </shortcut>
  </information>
  <information os="linux">
    <title> SwingSet2 Demo on Linux </title>
    <homepage href="docs/linuxhelp.html">
  </information>
  <security>
      <all-permissions/>
  </security>
  <resources>
    <j2se version="1.6+" java-vm-args="-esa -Xnoclassgc"/>
    <jar href="lib/SwingSet2.jar"/>
  </resources>
  <application-desc main-class="SwingSet2"/>
</jnlp>
```

Servidor Web

Servidor web - GlassFish

- Servidor Oracle;
- Um dos mais utilizados no mercado;
- Compatibilidade com Java Web Start e JAX;

Trabalhos Correlatos

- *Internet Chess Club* (ICC);
- Internet Xadrez Clube (IXC);
- Buho21;

DESENVOLVIMENTO

- Principais requisitos;
- Especificação;
- Implementação;
- Operacionalidade;

Principais Requisitos

- Permitir que um desafio seja nominal ou lançado em uma tabela de desafios;
- Mostrar os jogadores conectados, os desafios lançados e as partidas em andamento;
- Atribuir ou decrementar uma pontuação (rating) para cada jogador;

Principais Requisitos

- Permitir que os jogadores possam jogar, além de xadrez clássico, xadrez randômico, mata-mata e australiano;
- Possibilitar que o jogador salve a partida após o término em formato PGN;

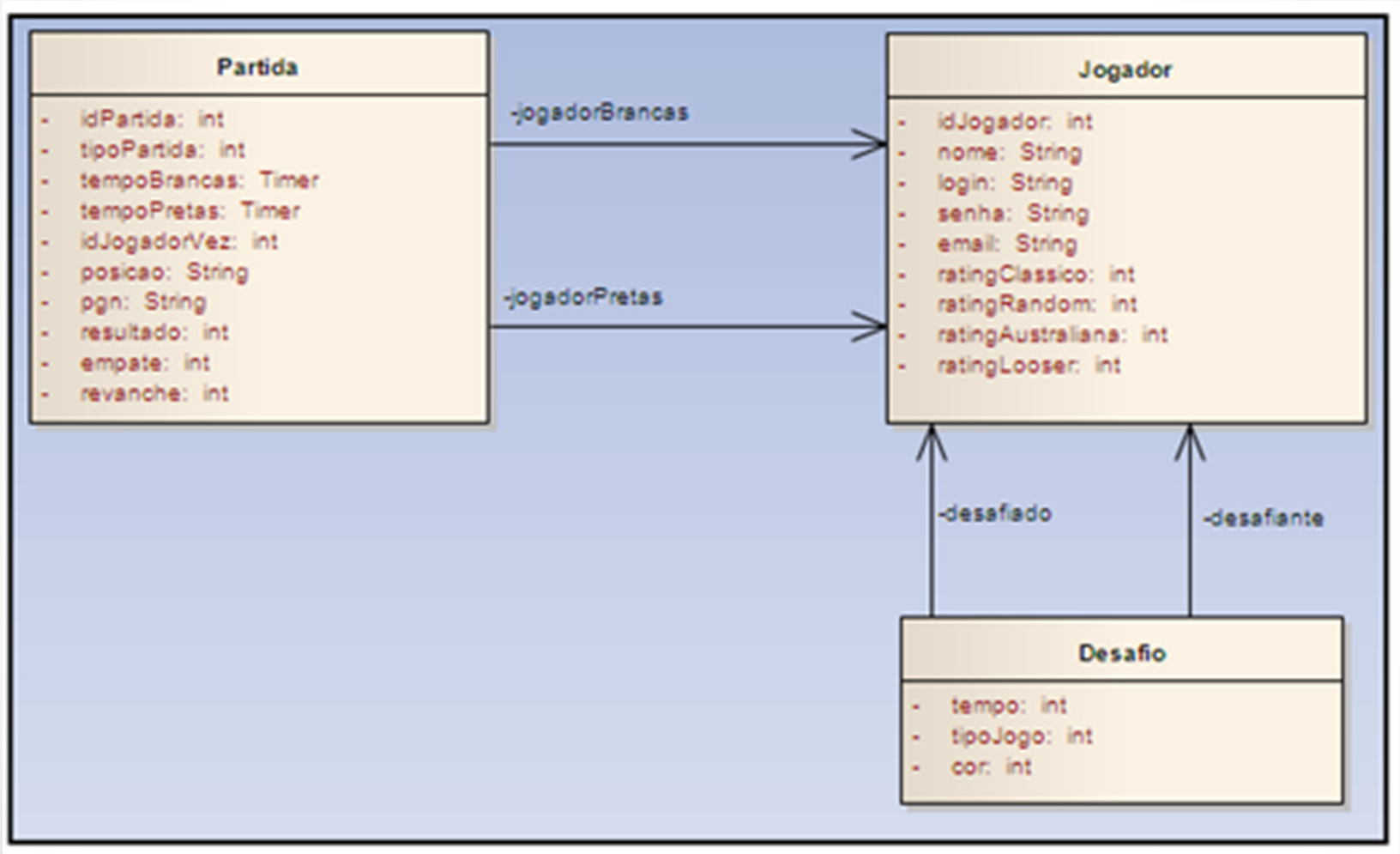
Especificação

- Divisão entre dois projetos: XadrezServidor e XadrezCliente;
- XadrezServidor – métodos que serão acessados remotamente (webservice e Banco de Dados);
- XadrezCliente – validação das jogadas

XadrezServidor

- Classes Partida, Jogador e Desafio funcionando como beans;
- Serialização dos objetos para transmissão;

XadrezServidor



XadrezServidor

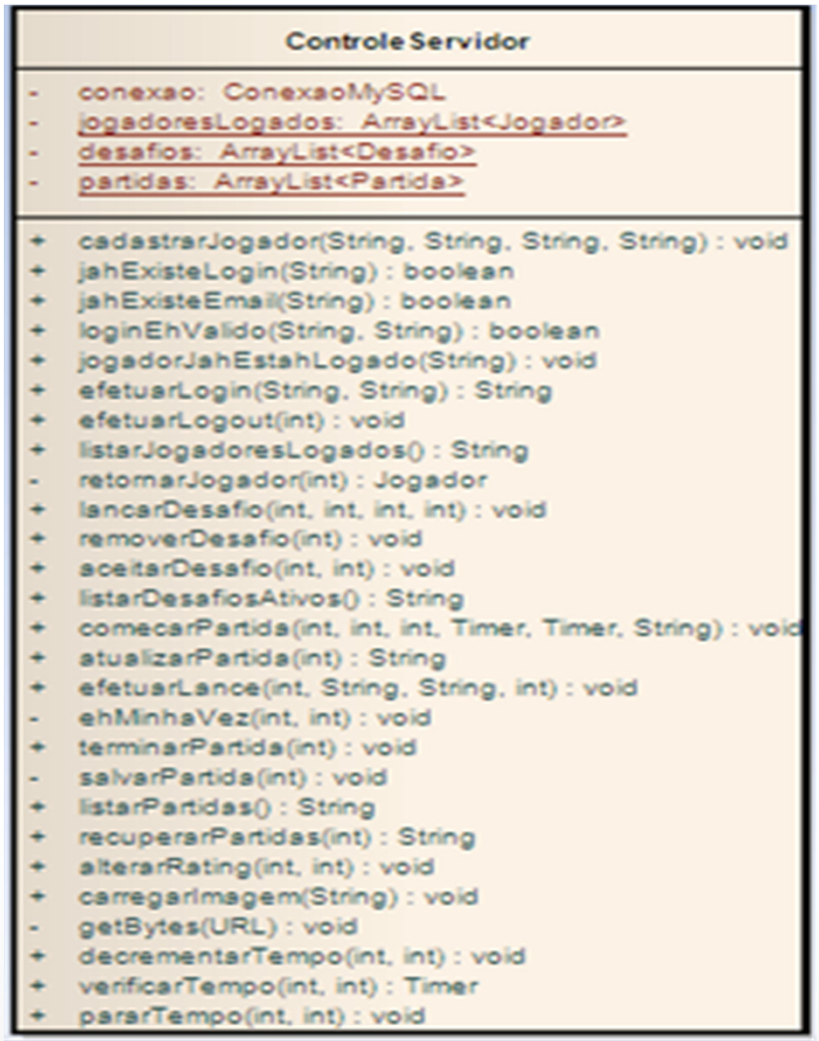
- Classe ConexaoMySQL faz a conexão com o banco de dados;
- Validação de usuário, cadastro e verificação de pontuação;

XadrezServidor

XadrezServidor

- Classe ControleServidor chama os métodos da classe ConexaoMySQL;
- Listas de jogadores, desafios e partidas em memória;
- Retorna Strings no lugar de objetos, através da biblioteca XStream;

XadrezServidor



XadrezServidor

- Classe XadrezWebService desenvolvida através do JAX;
- Possui todos os métodos públicos da classe ControleServidor;
- Arquivo XadrezWebServiceService.wsdl é gerado após deploy da aplicação;

XadrezServidor

```
<message name="jahExisteEmail">
<part name="parameters" element="tns:jahExisteEmail" />
</message>
<message name="jahExisteEmailResponse">
<part name="parameters" element="tns:jahExisteEmailResponse" />
</message>
<message name="loginEhValido">
<part name="parameters" element="tns:loginEhValido" />
</message>
<message name="loginEhValidoResponse">
<part name="parameters" element="tns:loginEhValidoResponse" />
</message>
```

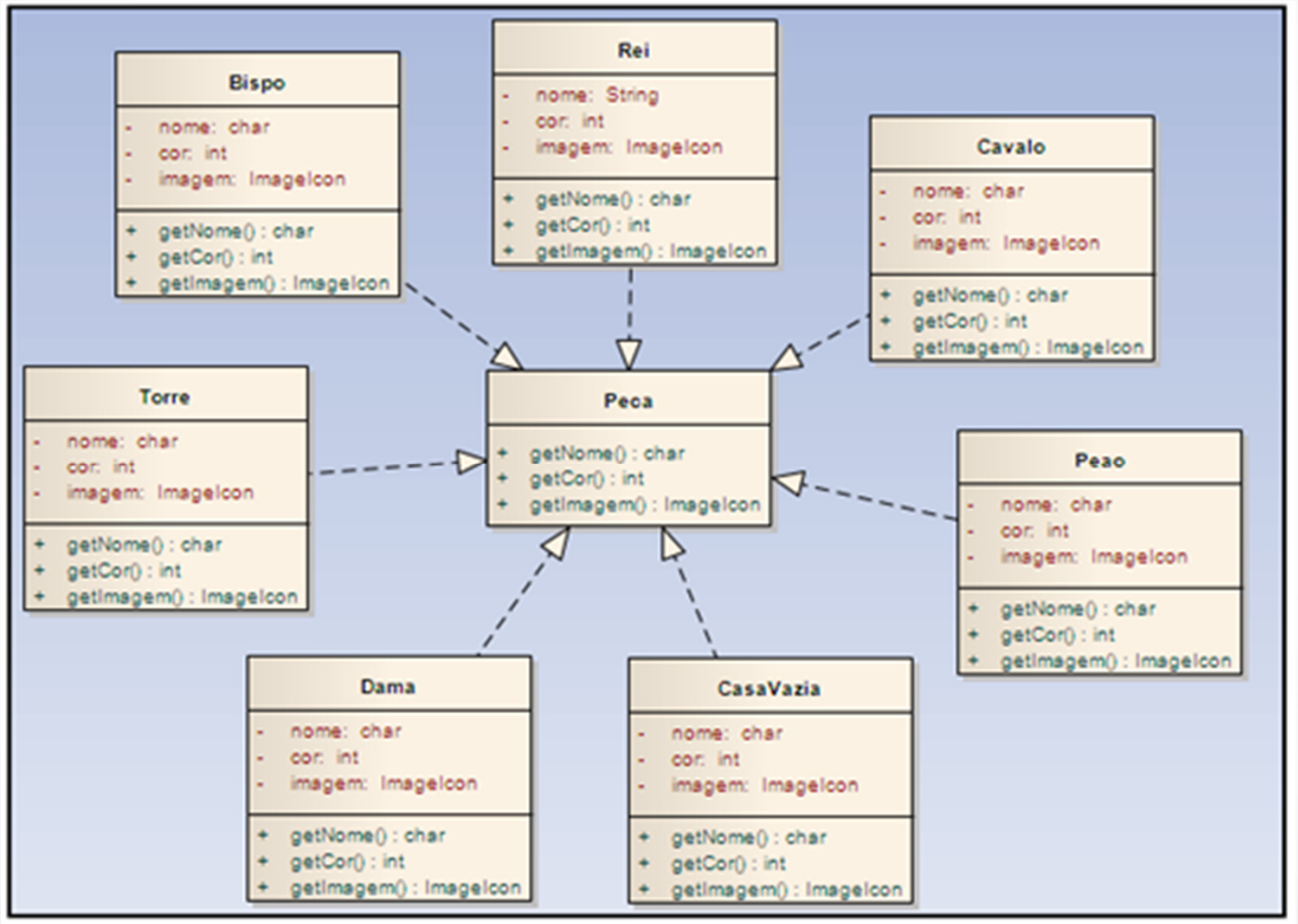
XadrezCliente

- Classes LoginGUI, LoginDialog, PrincipalGUI, DesafiosDialog e PartidaGUI responsáveis pelas telas;
- Classe LoginException para personalizar o tratamento de exceções;
- Classes Partida, Desafio e Jogador também funcionando como beans;

XadrezCliente

- Pacote pecas guarda uma classe para cada tipo de peça do tabuleiro;
- No objeto, o nome da peça é sua inicial – se for maiúscula é branca, senão, preta;
- Atributo imagem guarda a imagem buscada no servidor, transmitida como um array de bytes;

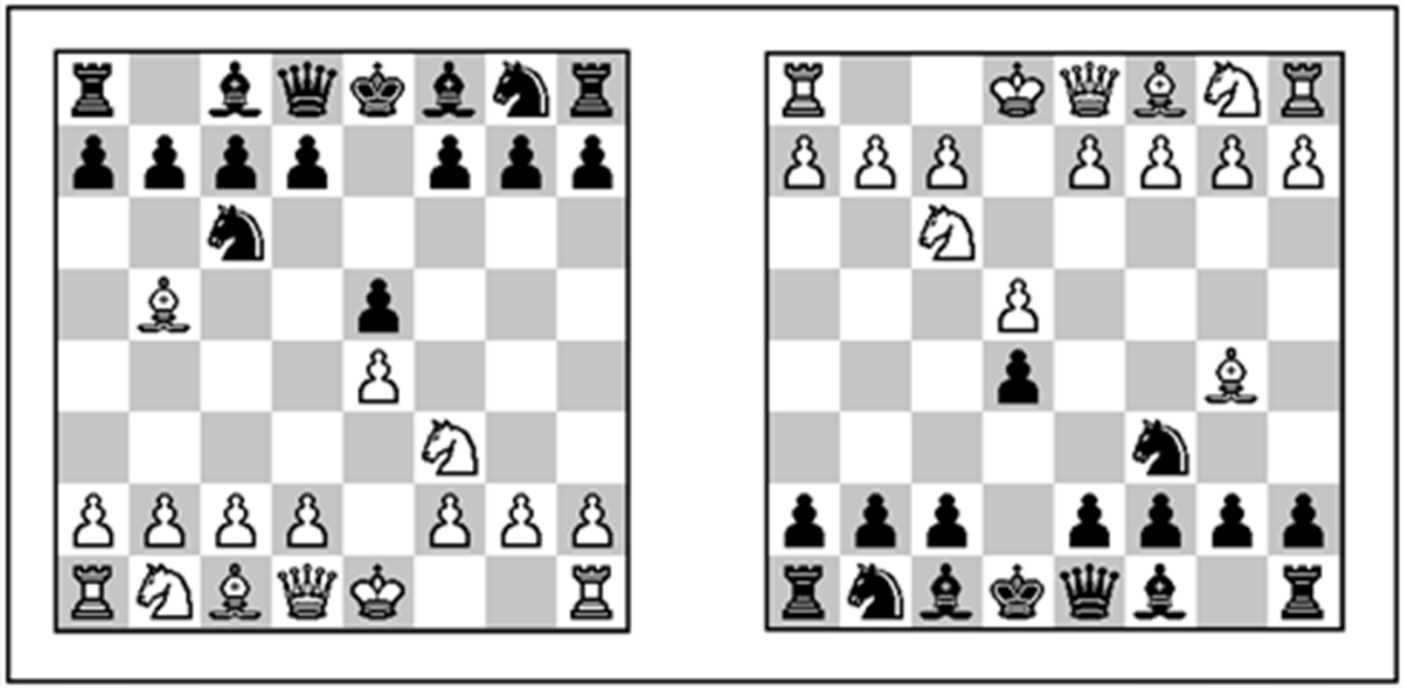
XadrezCliente



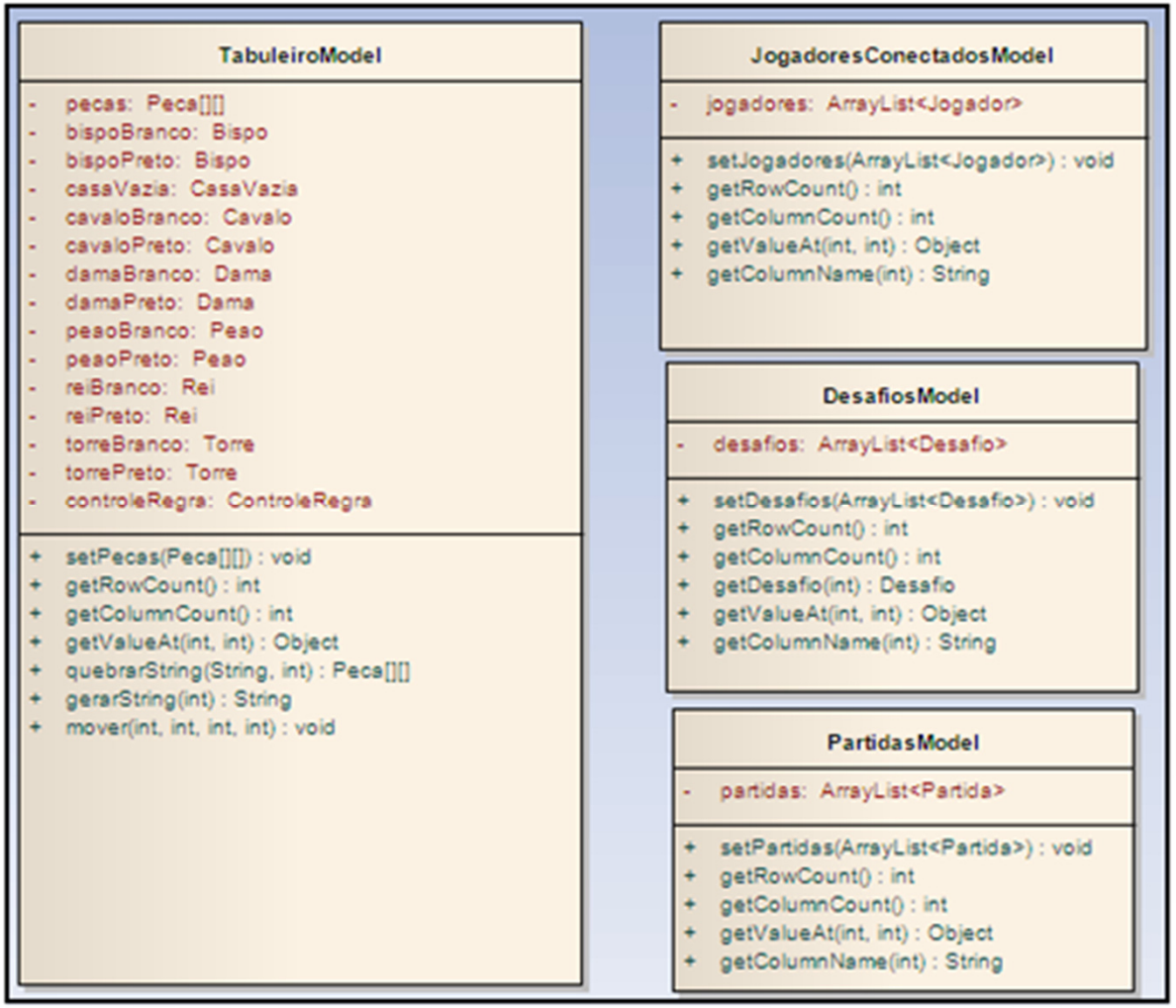
XadrezCliente

- Classes TabuleiroModel, JogadoresConectadosModel, DesafiosModel e PartidasModel gerenciam os conteúdos das JTable que serão utilizadas;
- TabuleiroModel - métodos quebrarString e gerarString;
- Array de peças no TabuleiroModel;

XadrezCliente



XadrezCliente



XadrezCliente

- Classes DesafiosThread, PartidasThread, TabuleiroThread e JogadoresThread são os threads responsáveis pelo loop de requisições ao servidor;
- Cada thread possui uma referência para um modelo de tabela que será renderizado;

XadrezCliente

- Pacote webservice – classes geradas pelo JAX;
- Cada operação no webservice vira uma classe;
- Caso tenha retorno, a função gera também uma classe Response;

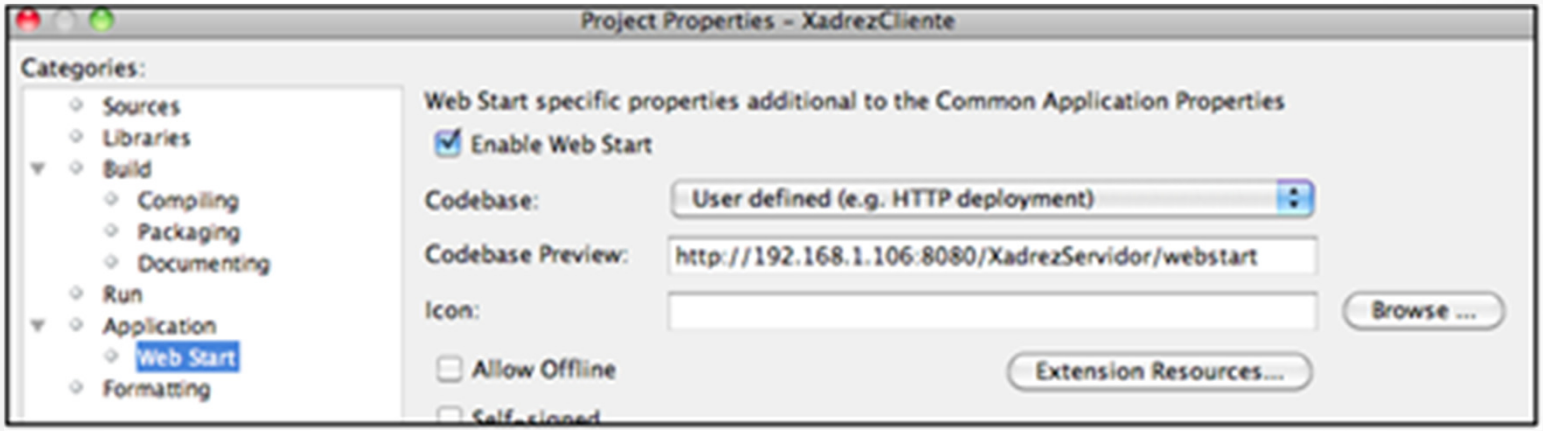
XadrezCliente

```
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "listarDesafiosAtivos")
public class ListarDesafiosAtivos {


}
```

```
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "listarDesafiosAtivosResponse", propOrder = {
    "_return"
})
public class ListarDesafiosAtivosResponse {

    @XmlElement(name = "return")
    protected String _return;

    /**
     * Gets the value of the return property.
     *
     * @return
     *     possible object is
     *     {@link String }
     *
     */
    public String getReturn() {
        return _return;
    }

    /**
     * Sets the value of the return property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     *
     */
    public void setReturn(String value) {
        this._return = value;
    }

}
```

IMPLEMENTAÇÃO

- Preparação do ambiente:
  - Habilitar o Java Web Start no XadrezCliente;
  - Apontar a pasta onde o arquivo JNLP ficará armazenado;
  - Inicializar serviço do MySQL;

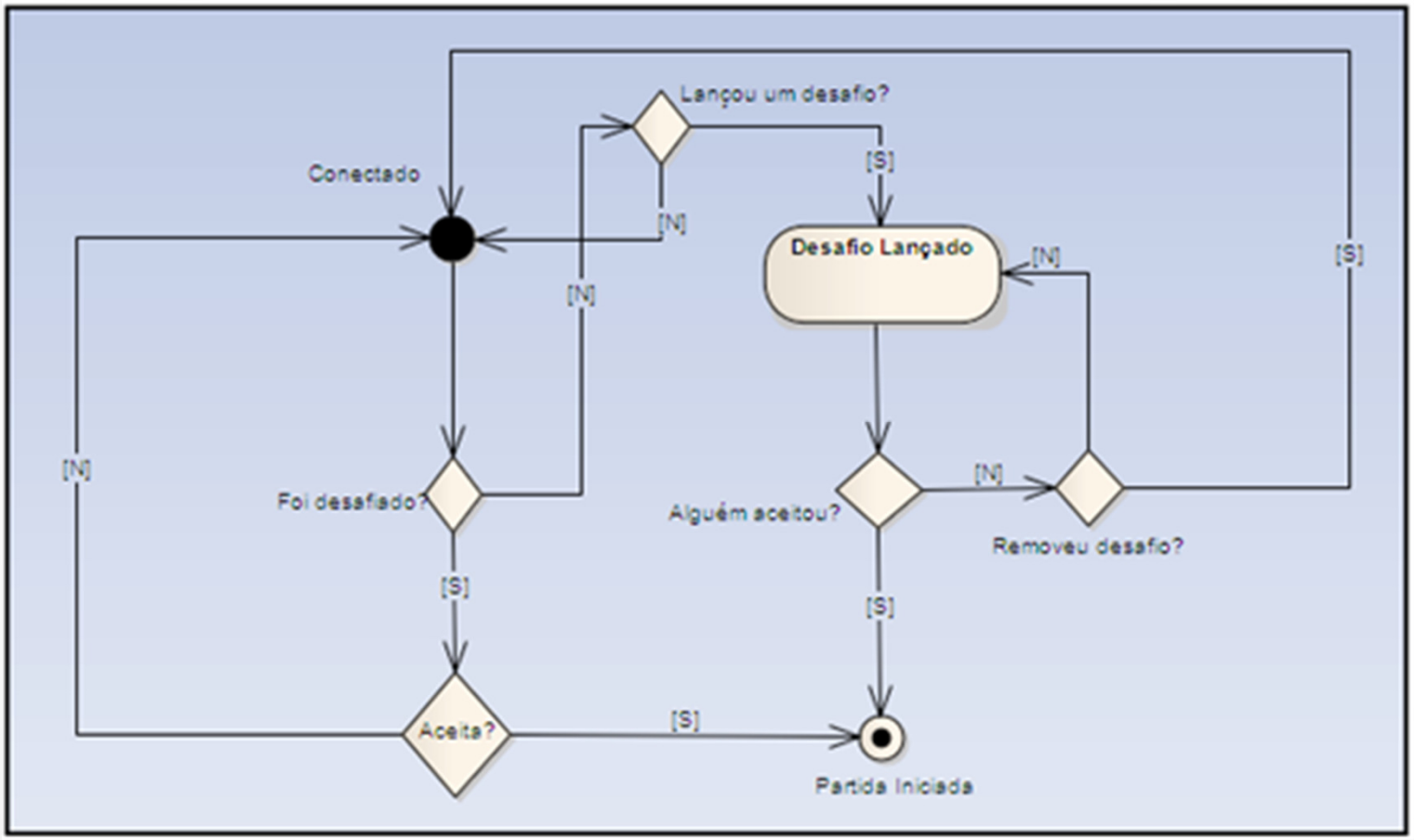


IMPLEMENTAÇÃO

- Efetuando login:
  - Tela LoginGUI valida os dados através do ControleCliente;
  - Caso valide, fecha a janela e abre a tela PrincipalGUI;
  - Threads de renderização disparados;

- Lançando um desafio:
  - Ao lançar um desafio na tabela, o atributo desafios, da classe ControleServidor, recebe um novo objeto;
  - O thread DesafiosThread pergunta contantemente se tem algum desafio na lista desafiosNominais, na classe ControleServidor;
  - Quando o desafio é aceito, é removido da lista e é adicionado um objeto partida na lista partidas, da classe ControleServidor;

IMPLEMENTAÇÃO



IMPLEMENTAÇÃO

- Os métodos de listagem retornam uma String no lado do servidor e são desmontados no cliente:

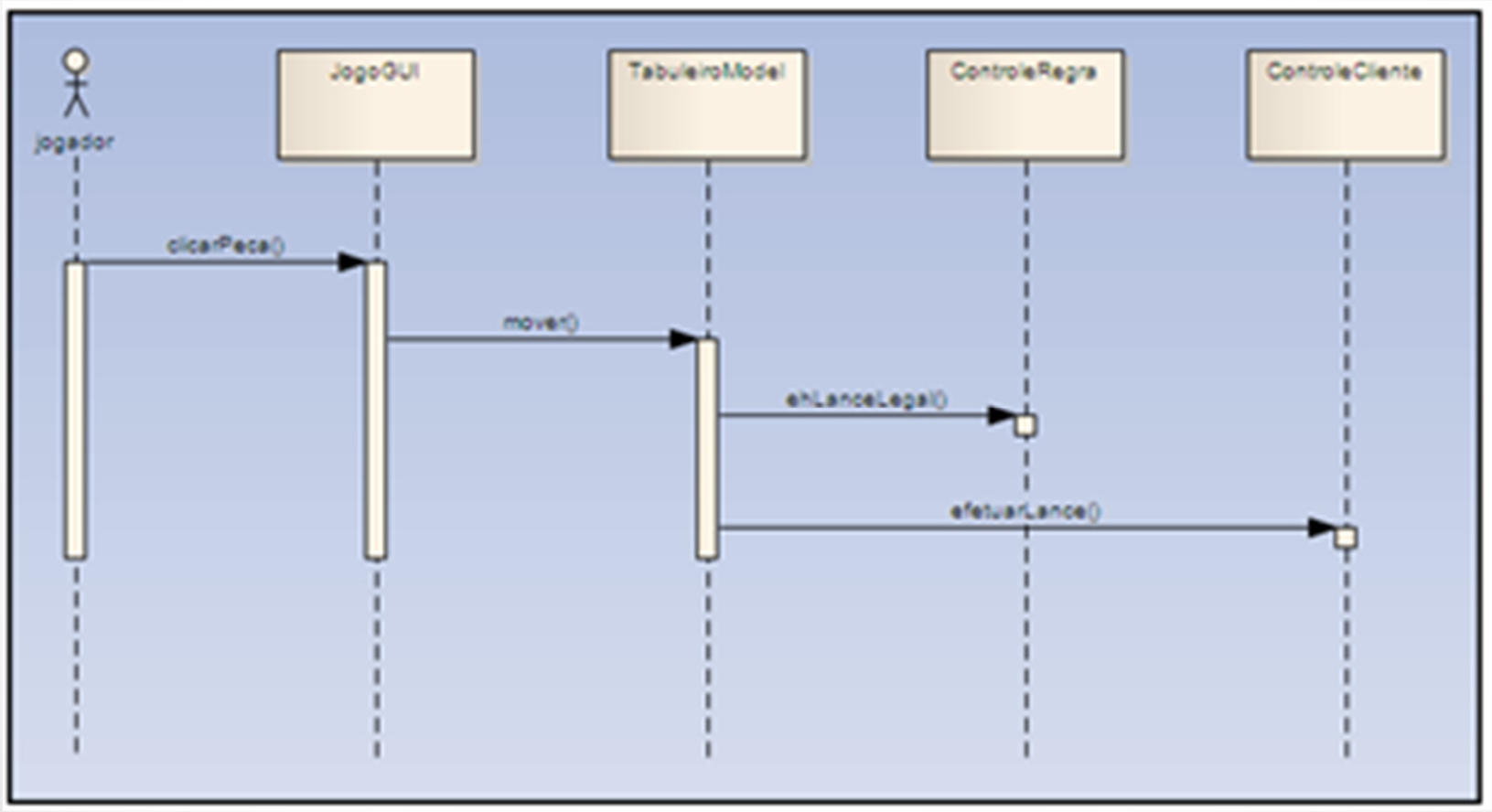
```
public static synchronized ArrayList<Jogador> listarJogadoresLogados() {
        XadrezWebServiceService webservice = new XadrezWebServiceService();
        XadrezWebService remoto = webservice.getXadrezWebServicePort();
        XStream arquivo = new XStream(new DomDriver());
        arquivo.alias("jogador", Jogador.class);
        ArrayList<Jogador> lista = (ArrayList<Jogador>)
arquivo.fromXML(remoto.listarJogadoresLogados());
        return lista;
    }
```

IMPLEMENTAÇÃO

- Movimentação:
  - Padrão: saber se de uma linhaInicial + colunaInicial é possível atingir uma linhaFinal + colunaFinal;
  - Com exceção do cavalo, para todas as peças é executado um algoritmo de verificação do caminho;
  - A cada lance, é verificado se o jogador entra em xeque;

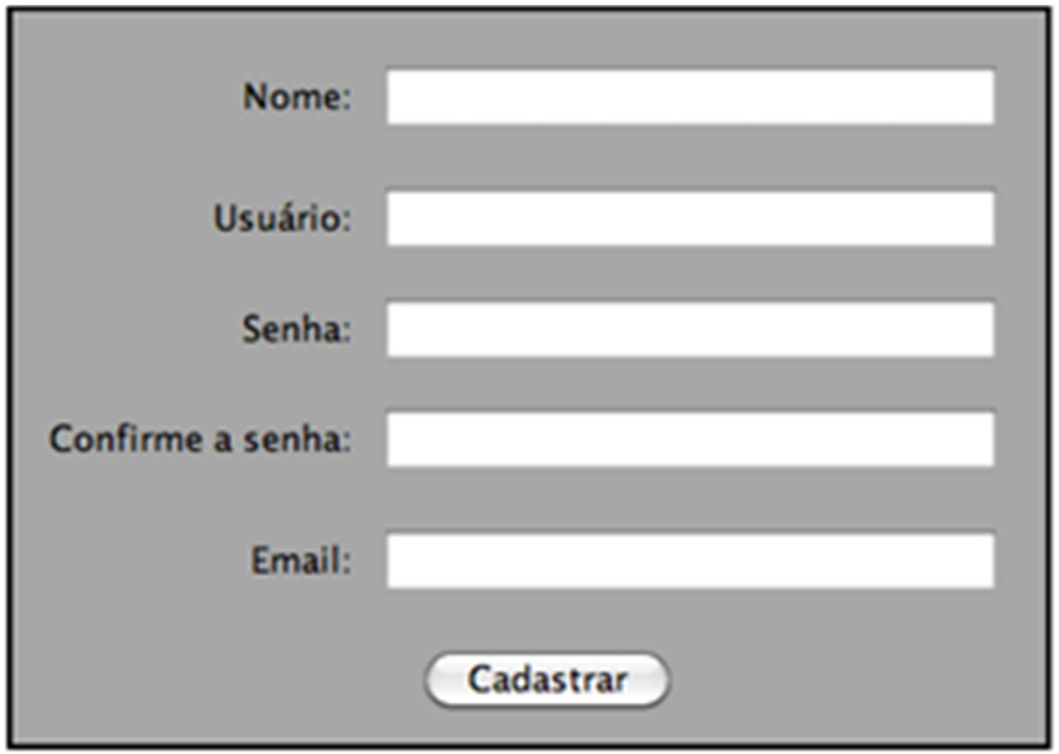
- Partidas:
  - Compara-se a posição guardada em memória local com a do servidor;
  - Se estiver diferente, atualiza a posição local e dá a vez ao jogador;

IMPLEMENTAÇÃO



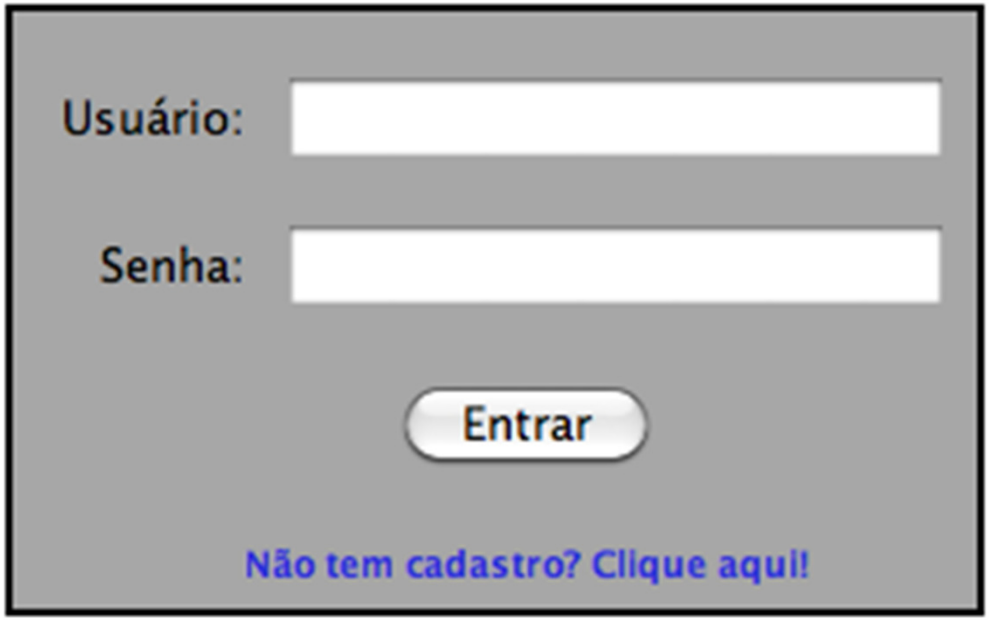
Operacionalidade

- Usuário efetua cadastro:



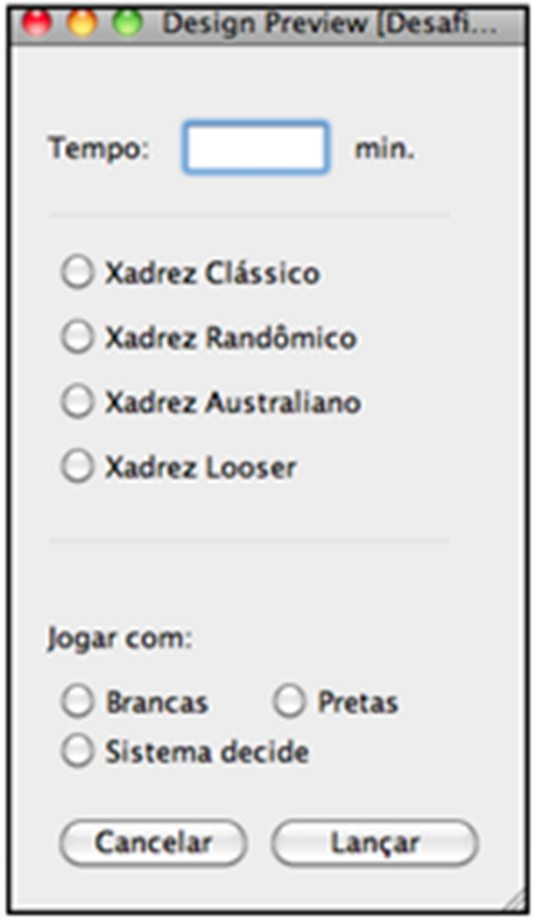
Operacionalidade

- Efetua Login:

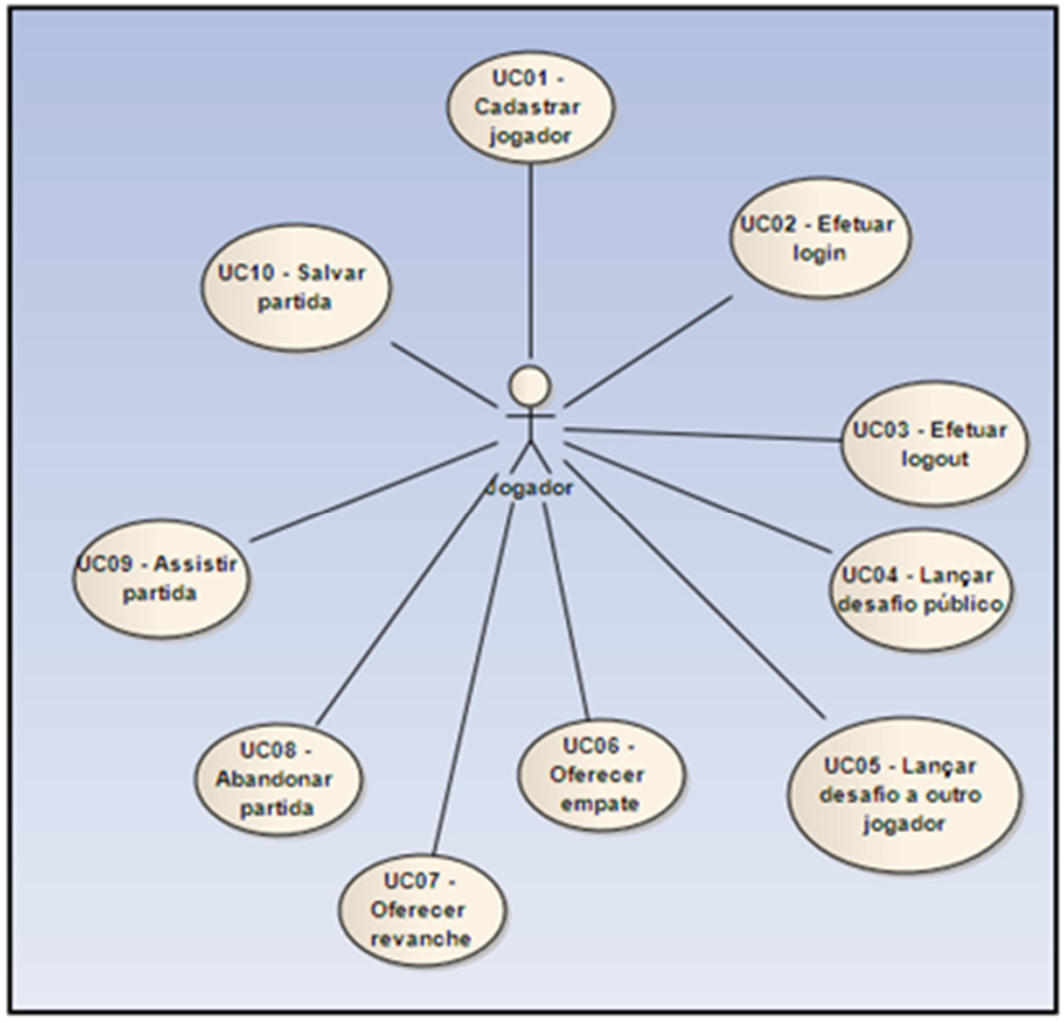


Operacionalidade

- Para jogar, efetua um desafio:



Operacionalidade



Resultados

- Perde um pouco em desempenho devido ao alto consumo de processamento dos threads;
- Regras do xadrez funcionaram normalmente;
- Interface gráfica sem perdas nem lentidão, mesmo as peças sendo passadas como array de bytes por XML;

CONCLUSÕES

- JAX – excelente ferramente para se trabalhar com webservices;
- Java Web Start permitiu usar o Java e a modelagem por Orientação a Objetos, facilitando o desenvolvimento das regras;
- NetBeans é a ferramenta ideal para o projeto;

CONCLUSÕES

- Limitações: alto consumo dos threads e necessidade de uma JVM instalada;
- Possível solução: padrão de projetos Observer.

EXTENSÕES

- Implementar uma Engine;
- Padrão de projetos Observer;