

# Projeto Mega Senha

## Especificação de Requisitos de Software

Versão 2.0

30/11/2016

Versão	Autores	Data	Ação
1.0	Leonardo P. Horstmann e Gabriel L. Chittolina Amaral	07/09/2016	Definição de requisitos
1.1	Leonardo P. Horstmann e Gabriel L. Chittolina Amaral	21/09/2016	Revisão do documento de especificação de requisitos
1.2	Leonardo P. Horstmann e Gabriel L. Chittolina Amaral	26/10/2016	Correção das etapas anteriores e elaboração dos diagramas de sequência e máquina de estados
1.3	Leonardo P. Horstmann e Gabriel L. Chittolina Amaral	22/11/2016	Desenvolvimento dos diagramas de modelagem de algoritmo e correção de etapas anteriores
2.0	Leonardo P. Horstmann e Gabriel L. Chittolina Amaral	30/11/2016	Correção etapas anteriores e finalização do projeto

### Conteúdo:

1. Introdução
2. Visão Geral
3. Requisitos de Software
4. Esboço de Interface Gráfica

### 1. Introdução:

## 1.1. Objetivo:

O sistema tem como objetivo a emulação de um jogo para dois jogadores utilizando arquitetura cliente-servidor.

## 1.2. Regras do Jogo:

O jogo possui dois tipos diferentes de jogador, mestre e advinho, o mestre é o responsável por fornecer as dicas, e o advinho por tentar adivinhar as senhas. O usuário que solicitar o início da partida deve ser o mestre, restando ao outro o papel de advinho. Os papéis de cada usuário dentro do jogo não se alteram (i.e., o mestre e o advinho não podem trocar de funções entre si).

Ao iniciar uma partida, o mestre recebe uma senha aleatória e um conjunto de dicas possíveis para esta senha. As dicas são separadas em conjuntos conforme seu nível, deixando com que o mestre defina sua estratégia de jogo. Para dar sequência no jogo o mestre deve escolher uma das dicas e enviar.

Para concluir a rodada o advinho recebe a dica enviada e faz uma tentativa. Ao fim da rodada, se o advinho acertou é finalizado o turno, caso contrário, o turno pode se estender por mais duas (2) rodadas (são no máximo 3 rodadas por turno) ou até que o advinho acerte a senha em questão. A cada nova rodada o mestre deve fornecer uma nova dica, e cada vez que uma rodada termina os pontos referentes a ela devem ser contabilizados.

### Definições:

**Senha:** a senha consiste na palavra que deve ser adivinhada (deve ser apenas uma palavra), e um conjunto de dicas referentes a ela;

**Dica:** cada dica é uma palavra ou expressão que facilite a descoberta da senha. Cada dica deve ter um valor referente ao seu nível (potencial de ajuda para que o advinho acerte a senha).

**Pontuação:** a pontuação é dada aos jogadores conforme seu tipo e sua performance. Para o tipo mestre a pontuação é dada pelo nível da dica enviada, já para o tipo adivinho, a pontuação é dada conforme a rodada em que houve acerto e o tipo da dica recebida. Caso não haja acerto o adivinho não pontua.

**Níveis de dica:** as dicas podem ser de 3 tipos: Boa, Regular e Ruim.

### Pontuações por dica:

Dica boa:

Mestre: 100 pontos.

Adivinho: 0 pontos.

Dica Regular:

Mestre: 50 ponto.

Adivinho: 50 ponto.

Dica Ruim:

Mestre: 10 pontos.

Adivinho: 100 pontos.

**Observação:** a pontuação referente ao tipo de dica só é dada ao adivinho quando o mesmo acerta a senha.

**Pontuação referente a rodada de acerto:** o jogo pontua o advinho conforme a rodada em que houve acerto seguindo o seguinte critério:

1ª Rodada: 100 pontos.

2ª Rodada: 50 pontos.

3ª Rodada: 10 pontos.

**Duração do Jogo:** o jogo é formado por 5 turnos, cada um com até 3 rodadas.

**Definição de Vencedor:** ao final do jogo, o vencedor é o jogador que mais somou pontos.

## 2. Visão Geral

### 2.1. Premissas de desenvolvimento

- O sistema deve ser desenvolvido na linguagem de programação Java, seguindo o paradigma Orientado a Objetos.
- O sistema deve usar a arquitetura cliente-servidor utilizando o NetGamesNRT.
- O sistema deve utilizar a plataforma de programação Eclipse para Java.

## 3. Requisitos de Software

### 3.1 Requisitos Funcionais

1. **Conectar:** o sistema deve permitir ao usuário conectar-se ao servidor NetGames que é responsável pela gestão dos usuários conectados.
2. **Desconectar:** deve permitir ao usuário que possa se desconectar do servidor via interface gráfica.
3. **Iniciar Partidas:** uma vez conectado, o sistema deve permitir que o usuário inicie uma partida. O programa deve permitir partidas contendo dois jogadores.
4. **Finalizar Partidas:** uma vez a partida estando em andamento, o sistema deve permitir que o

jogador finalize o jogo.

5. **Procedimento de Lance:** o jogador deve permitir que o usuário, em sua vez, faça sua jogada desde que a mesma não infrinja as regras do jogo.
6. **Definir Vencedor:** após o término da partida o sistema deve informar qual o vencedor.

### 3.2 Requisitos não funcionais

1. **Especificação do projeto:** deve ser elaborada a especificação do projeto baseada em UML segunda versão.
2. **Dados do jogo:** o sistema deve fornecer a visualização dos status da partida (as pontuações dos jogadores, o turno e a rodada) para os jogadores.

## 4. Esboço de interface gráfica

O esboço da interface gráfica para o jogo MEGA SENHA é apresentado em um layout organizado. No topo esquerdo, o título "MEGA SENHA" é exibido em letras grandes, com "MEGA" em laranja e "SENHA" em azul. No topo direito, há uma caixa de status com o título "Pontos" e as pontuações "Mestre: XXX" e "Adivinho: YYY", além de "Turno/Rodada T/R". Abaixo do título, o texto "Sua Dica Foi:" em amarelo precede um campo de dica em um retângulo arredondado com uma borda vermelha, contendo o placeholder "< campo contendo a dica >". À direita, o texto "Tipo: <tipo dica>" em vermelho indica o tipo da dica. Abaixo, o texto "Sua tentativa:" em amarelo precede um campo de resposta em um retângulo com uma borda preta, contendo o placeholder "< campo onde digita-se a resposta >". Na base esquerda, um botão "Sair" com uma seta para a esquerda indica a saída. Na base direita, um botão "Enviar Tentativa" com uma borda vermelha indica o envio da resposta.

**MEGA**  
**SENHA**

Sua senha é:

**FOTO/IMAGEM DA  
SENHA**

Pontos

Mestre: XXX

Adivinho: YYY

Turno/Rodada

T/R

Dicas Boas

Dicas Regulares

Dicas Ruins

---

---

---

---

---

---

---

---

---



SAIR

ENVIAR