UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DAINF – DEPARTAMENTO ACADÊMICO DE INFORMÁTICA

TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

RENATO CANIZELLA

DESENVOLVIMENTO DE UMA REDE SOCIAL VOLTADA PARA JOGOS DO TIPO MOBA

TRABALHO DE CONCLUSÃO DE CURSO

PONTA GROSSA

2014

RENATO CANIZELLA

DESENVOLVIMENTO DE UMA REDE SOCIAL VOLTADA PARA JOGOS DO TIPO MOBA

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, do DAINF, da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Daniel Costa De Paiva

PONTA GROSSA

2014

TERMO DE APROVAÇÃO

DESENVOLVIMENTO DE UMA REDE SOCIAL VOLTADA PARA JOGOS DO TIPO MOBA

por

RENATO CANIZELLA

Este Trabalho de Conclusão de Curso (TCC) foi apresentado(a) em 02 de dezembro de 2014 como requisito parcial para a obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Daniel Costa de Paiva
Prof. Orientador

André Koscianski
Membro titular

Denis Lucas Silva
Membro titular

Tânia Lúcia Monteiro
Responsável pelos Trabalhos
de Conclusão de Curso

Simone de Almeida
Coordenadora do Curso
UTFPR - Campus Ponta Grossa

AGRADECIMENTOS

Embora estes parágrafos não consigam contemplar todas as pessoas responsáveis direta ou indiretamente pela escolha do tema, ou pelo apoio ao tema escolhido, listo aqui os principais envolvidos no processo de escolha e desenvolvimento do tema.

Primeiramente, aos meus pais, pelo apoio e suporte prestados em todos esses anos, nas minhas escolhas pessoais e profissionais.

A minha namorada, pela paciência, compreensão, dedicação e apoio nestes meses.

Aos meus amigos pessoais ou virtuais que de alguma forma contribuíram para o escopo e testes do projeto em si, sendo os principais colaboradores, Francielli, pelos anos de amizade, as inúmeras ideias e dicas de layout e pelo apoio ao tema escolhido, Jean “Jeanxes”, Fábio “Eton Garnel”, Fabrício “Galf”, Bernardo “Flammi” e Christopher “Chrislix” pelas ideias e por serem os escolhidos para testar a plataforma.

RESUMO

CANIZELLA, Renato. **Desenvolvimento de uma rede social voltada para jogos do tipo MOBA**. 2014. 76 páginas. Trabalho de Conclusão de Curso Tecnologia em Análise e Desenvolvimento de Sistemas - Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2014.

Este trabalho discorre sobre a história dos jogos eletrônicos, sua origem e sua dimensão na atualidade. Abordando principalmente o jogo *League of Legends*, traz-se a proposta de criação de uma ferramenta que englobe os quesitos de rede social, gerenciador de torneios, sincronização com as informações pertencentes a Riot Games e um sistema de nivelamento opcional em torneios amadores. Os tipos de jogos definidos foram os jogos MOBA, que possuem uma grande quantidade de jogadores e pouca alteração no modelo entre os jogos. Os principais MOBA analisados foram os jogos *League of Legends* e *DotA*. A definição do módulo de rede social baseou-se em redes sociais populares, como o Twitter, como um modelo de desenvolvimento. A dinâmica desta rede social traz um foco maior no conteúdo relacionado aos jogos. O modelo de gerenciador de torneios foi concebido analisando-se duas ferramentas populares no ramo, agregando os pontos positivos das duas plataformas distintas. Uma análise sobre as tecnologias web a serem utilizadas permitiu traçar uma melhor abordagem as funcionalidades do sistema, tornando-o com um visual agradável ao usuário sem abandonar a sua funcionalidade. O nivelamento dos times participantes dos torneios é a principal vantagem do sistema, pois o mesmo não permite a inscrição de um time que não esteja conforme as regras de níveis estabelecidas no torneio. O sistema possui checagem de perfis na API fornecida pela *Riot Games*, cadastro de usuários, times e torneios. Além de opções de criação de torneios por qualquer usuário, em todos os mapas disponíveis no jogo *League of Legends* e a opção de nivelamento. Os resultados apresentados foram consistentes com a proposta, realizando com sucesso as interações da rede social, gerenciador de torneios, processo de equilíbrio entre os confrontos e a obtenção dos perfis de usuário pertencentes a *Riot Games*. Melhorias podem ser implementadas a partir do contato e interesse da *Riot Games*.

Palavras-chave: League of Legends, MOBA, ARTS, Rede Social, Torneios.

ABSTRACT

CANIZELLA, Renato. **Development of a dedicated social network for MOBA games.** 2014. 76 pages. Trabalho de Conclusão de Curso Tecnologia em Análise e Desenvolvimento de Sistemas. Federal Technology University - Parana. Ponta Grossa, 2014.

This paper discusses the history of video games, its origin and its dimension today. Mainly addressing the League of Legends game, brings up the proposal to create a tool that incorporates the criteria of social network, tournaments management, synchronization with information pertaining Riot Games and an optional leveling system in amateur tournaments. The types of games set were the MOBA games, which have a lot of players and little change in the model between games. The main MOBA games analyzed were the games League of Legends and DotA. The definition of the social network module was based on popular social networks such as Twitter, as a development model. The dynamics of this social network brings a greater focus on content related to the games. The tournament manager model was designed by analyzing two popular tools in the business, adding the strengths of the two different platforms. An analysis of web technologies to be used allowed to draw a better approach system features, making it with a nice visual to the user without abandoning its functionality. The leveling of the participants of the tournaments teams is the main advantage of the system, because it does not allow the registration of a team that does not conform to the rules established levels in the tournament. The system is checking profiles in the API provided by Riot Games, user registration, teams and tournaments. In addition to creating tournaments for any user options, in all the maps available in the game League of Legends and leveling option. The results were consistent with the proposal, successfully performing the interactions of social network, manager of tournaments, balancing process between the fighting and obtain the user profiles of Riot Games. Improvements can be implemented from the contact and interest of Riot Games.

Keywords: League of Legends, MOBA, ARTS, Social Network, Tournaments.

LISTA DE FIGURAS

Figura 1 – Interações Possíveis do Sistema 16
Figura 2 - Representação genérica de um mapa MOBA.....21
Figura 3 - Mapa DotA 21
Figura 4 - Mapa de *League of Legends* 24
Figura 5 - Final da Primeira Temporada de *League of Legends* (2011).....27
Figura 6 - Final da Segunda Temporada de *League of Legends* (2012).....27
Figura 7 - Final da Terceira Temporada de *League of Legends* (2013).....28
Figura 8 - Finais da Quarta Temporada de *League of Legends* (2014) 29
Figura 9 - Laços de Amizade em uma Rede Social 32
Figura 10 - Modelo de Casos de Uso.....36
Figura 11 - Modelo Entidade Relacionamento 40
Figura 12 - Relacionamento de Usuário e Rede Social.....41
Figura 13 - Relacionamento de Usuário e Time.....41
Figura 14 - Relacionamento Entre Times e Torneios.....42
Figura 15 - Interface de Login 43
Figura 16 - Tela principal do módulo redes sociais da arena moba. 43
Figura 17 - Formulário de Criação de Torneios.....47
Figura 18 - Layout de Busca de Times.....48
Figura 19 - Chave com 6 Times 50
Figura 20 - Aplicação da Regra da Maioria com Duas Escolhas.....51
Figura 21 - Segunda Rodada de Confrontos.....51
Figura 22 - Chaves Finais 52
Figura 23 - Descrição do Torneio Completo.....52

LISTA DE QUADROS

Quadro 1 - Comparativo Entre os Principais Jogos.....22
Quadro 2 – Comparativo de uma requisição através da Riot Games API em JSON e XML.....38
Quadro 3 - Comparativo entre JQuery e JavaScript.....39
Quadro 4 - Linhas de código responsáveis pelo conteúdo apresentado.....44
Quadro 5 - Linhas de código responsáveis pela Requisição na API.....45
Quadro 6 - Retorno JSON de requisição.....46
Quadro 7 - Código de registro de torneios.....48
Quadro 8 - Códigos Responsáveis Pela Exibição do Layout.....49
Quadro 9 - Resultados dos confrontos.....53
Quadro 10 - Resultados dos confrontos.....53

LISTA DE SIGLAS E ACRÔNIMOS

AJAX	*Asynchronous JavaScript and XML*
API	*Application Programming Interface*
ARTS	*Action Real Time Strategy*
DotA	*Defense of the Ancients*
HTML	*Hypertext Markup Language*
JSON	*JavaScript Object Notation*
LoL	*League of Legends*
MER	Modelo Entidade Relacionamento
MMORPG	*Multiplayer Massive Role Playing Game*
MOBA	*Multiplayer Online Battle Arena*
PHP	*Hypertext Preprocessor*
RPG	*Role Playing Game*
RTS	*Real Time Strategy*
SGBD	Sistema Gerenciador de Banco de Dados
SQL	*Structed Query Language*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO 13
2 REFERENCIAL TEÓRICO 17
2.1 TIPOS DE JOGOS 17
2.2 MOBA 19
2.2.1 Defence of the Ancients 20
2.2.2 League of Legends 22
2.3 FORMAS DE COMPETIÇÃO 29
2.3.1 Eliminação Simples 30
2.3.2 Pontos Corridos 31
2.4 REDE SOCIAL ON LINE 32
2.5 SISTEMAS SIMILARES 33
2.5.1 Arena e-Sports 34
2.5.2 CHALLONGE! 34
3 DESENVOLVIMENTO 36
3.1 TECNOLOGIAS WEB 37
3.1.1 PHP 37
3.1.2 JavaScript 37
3.1.3 JSON 38
3.1.4 JQuery 39
3.1.5 Bootstrap 39
3.1.6 Riot Games API 39
3.2 BANCO DE DADOS 40
3.3 IMPLEMENTAÇÃO 42
4 RESULTADOS E ANÁLISE 50
5 CONCLUSÕES 55
5.1 APLICAÇÕES E ABRANGÊNCIA 56
5.2 TRABALHOS FUTUROS 57
REFERÊNCIAS 59
APÊNDICE A - Códigos da Página Principal 62
APÊNDICE B - Códigos da Página de Funções e Procedimentos 69

1 INTRODUÇÃO

A história dos jogos eletrônicos se confunde, em muitos aspectos, com a própria história da computação. (Batista, et al., 2007) Na década de 50, foram criados os primeiros jogos, como forma de estudo de inteligência artificial e de física. Anos mais tarde, surgiram os *consoles* e fliperamas. Com a popularização dos computadores pessoais e o amplo acesso a internet, os jogos ganharam mais interatividade, podendo-se formar equipes ou disputar partidas contra vários adversários ao mesmo tempo. No ramo dos jogos, criaram-se vários estilos diferentes, tendo como principais o MMORPG (Massive *Multiplayer Online Role Playing Game*), RTS (*Real Time Strategy*), FPS (*First Person Shooter*).

Muito populares nas décadas de 90 e 2000, os jogos RTS ganharam destaque entre os jogadores, as partidas exigiam a atenção do jogador em diversos pontos, a elaboração de estratégias diversificadas e o sistema de criação de uma nação, compra de unidades de batalha, aprimoramento destas unidades, estruturas de defesa e ataque popularizaram este estilo de jogo, com os principais títulos sendo Warcraft III, Age of Empires, StarCraft e Command & Conquer (BURO, 2003). Após alguns anos, com a popularização dos computadores pessoais, os jogos RTS se estabeleceram, principalmente nesta nova e crescente plataforma. Com isto, surge a oportunidade dos jogadores se desafiarem via internet ou rede.

Modificações, feitas pelos próprios jogadores deram origem ao estilo Action RTS[1], ou ARTS, que hoje conhecemos como MOBA (*Multiplayer Online Battle Arena*). O primeiro MOBA que se tornou uma febre entre os jogadores foi a modificação do RTS Warcraft III, baseado no mapa Aeon of Strife, de Starcraft. A modificação, chamada de Defense of The Ancients, ou DotA, foi idealizada por Ju An Kim, sócio da Blizzard, empresa criadora de Warcraft III e desenvolvido por “Eul”. Posteriormente, o mapa foi alterado por “IceFrog” e “Guinsoo” e renomeado para DotA: Allstars.

Após anos sem melhorias em seu motor de processamento, ou uma plataforma independente, outras empresas começaram a investir no estilo de jogo

[1] Disponível em: http://www.giantbomb.com/multiplayer-online-battle-arena/3015-6598 acesso em: 15/10/2014

MOBA, a empresa *Riot Games*, trazendo o *League of Legends* ao mercado de games. Também conhecido por LoL, foi desenvolvido em parceria com Steve "Guinsoo" Feak, desenvolvedor de DotA e Steve "Pendragon" Mescon, criador da comunidade DotA-Allstars.com. Rapidamente *League of Legends* se tornou o MOBA mais jogado entre os internautas[2], pelo seu estilo *free to play*, onde o jogo é gratuito e itens ou roupas para os personagens são comprados com dinheiro real, sem que isso interfira ou impacte na jogabilidade do mesmo. O benefício é a competição entre os jogadores para ter o maior número de itens e os itens diferenciados dos outros jogadores.

O objetivo de qualquer MOBA é basicamente o mesmo, coletar dinheiro abatendo tropas neutras, chamadas de *creeps* ou *minions*, abater campeões inimigos, destruir as torres que bloqueiam o acesso até a base inimiga e destruir o centro da base inimiga. Diferentes estilos de jogadores e diferentes níveis de habilidade são notados nos jogos MOBA, assim como em todos os jogos. Alguns jogos possuem uma forma de classificar estes níveis diferentes de habilidade, e tentam o máximo possível equilibrar as partidas, ou torneios para gerar uma boa experiência de jogo aos usuários.

Com a popularização dos jogos eletrônicos, os torneios, que os envolvem alcançam níveis profissionais. A origem destes torneios, muitas vezes, se dava com uma disputa em *arcades* entre vizinhos ou colegas. A febre cresceu e na década de 90 era fácil ver em qualquer estabelecimento que possuísse um *arcade*, uma placa indicando o próximo campeonato daquele lugar. Atualmente, os mais conhecidos torneios são os campeonatos de *League of Legends*, DotA, DotA2, *Counter Strike*, *CrossFire, Street Fighter* e FIFA, surgindo assim os *e-Sports*, do inglês *Eletronic Sports*, ou em tradução livre, esportes eletrônicos.

Após o *e-Sports* ter se popularizado, as premiações alcançam valores astronômicos, chegando a mais de US$ 25 milhões[3], somente no ano de 2013. Tratando-se de *League of Legends*, em 2014, a premiação do campeonato brasileiro

---

[2] Disponível em: http://www.forbes.com/sites/johngaudiosi/2012/07/11/riot-games-league-of-legends-officially-becomes-most-played-pc-game-in-the-world acesso em: 07/092014

[3] Disponível em: http://blogs.estadao.com.br/link/torneios-de-games-ganham-carater-profissional-e-atraem-multidoes acesso em: 14/08/2014.

foi em torno de R$ 100 mil[4] e a premiação total do campeonato mundial está em US$ 2 mi[5].

Com os torneios profissionais em alta, os jogadores se incentivam a melhorar, e a participar de torneios amadores. Afinal muitas empresas oferecem patrocínios para estes torneios amadores, em troca de divulgação da empresa, ou de uma marca específica. Na cidade de Ponta Grossa – PR, em 2013 e 2014 houve dois torneios de eliminação mista, que alcançaram premiações de R$ 500,00 e R$ 2500,00, respectivamente.

Existem diversas formas de torneios, abertos ao público em geral, que se registra em um *site* ou aplicativo disponível, outros torneios são direcionados apenas aos jogadores de nível profissional, possuindo convites restritos a esta comunidade. Em torneios profissionais, o modelo mais utilizado no cenário profissional é o torneio de eliminação simples nas fases iniciais e disputas com três e cinco *rematch* nas quartas de finais e finais, respectivamente. No ambiente de torneios amadores, diversas formas podem ser escolhidas de acordo com o interesse e disponibilidade de tempo dos administradores.

Um torneio, geralmente é construído em eliminação simples ou por pontos corridos. No gênero de esportes eletrônicos, geralmente opta-se pelo modelo de eliminação simples. Porém, nos jogos MOBA, é comum o torneio possuir eliminação simples até as quartas de finais, a partir daí, muda-se o modelo para melhor de três partidas, onde para avançar para a próxima chave deve-se vencer duas de três partidas e nas finais, opta-se por melhor de cinco partidas, campeão é declarado o time que obtiver vitória em três partidas.

O tema apresentado, gira em torno de uma problemática enfrentada por jogadores de jogos MOBA em torneios amadores, onde o principal, que deveria ser a diversão, muitas vezes se transforma em um cenário de frustrações e desânimo. Devido à dificuldade para diferenciar os níveis de cada jogador participante de um torneio amador, o que gera frustração de jogadores menos habilidosos, que participam visando somente o divertimento. Com este problema, surge a necessidade de inserir um sistema que busque a igualdade em torneios.

---

[4] Disponível em: http://br.leagueoflegends.com/pt/news/esports/esports-event/bem-vindos-ao-circuito-brasileiro-2014 acesso em: 01/10/2014

[5] Disponível em: http://www.techtudo.com.br/noticias/noticia/2013/10/league-legends-final-tera-premio-de-r-2-milhoes-e-estadio-lotado.html acesso em: 01/10/2014.

O projeto também conta com uma rede social simples embutida. Nesta rede social é possível seguir pessoas e compartilhar textos próprios, baseado em um limite de caracteres previamente estabelecido. O objetivo desta rede social é manter o usuário em constante contato com possíveis companheiros de time, jogadores profissionais, empresas ou entidades que desejam criar torneios e principalmente, divulgar torneios criados na plataforma. O ideia da rede social é manter um aspecto simples e funcional, baseando-se em redes sociais, como por exemplo o *Twitter* e *Ello*.

Dividido basicamente em três módulos, rede social, integração com o sistema da *Riot Games* e gerenciador de torneios, o sistema possui uma série de ações possíveis exemplificados no diagrama de casos de uso, conforme a Figura 1.

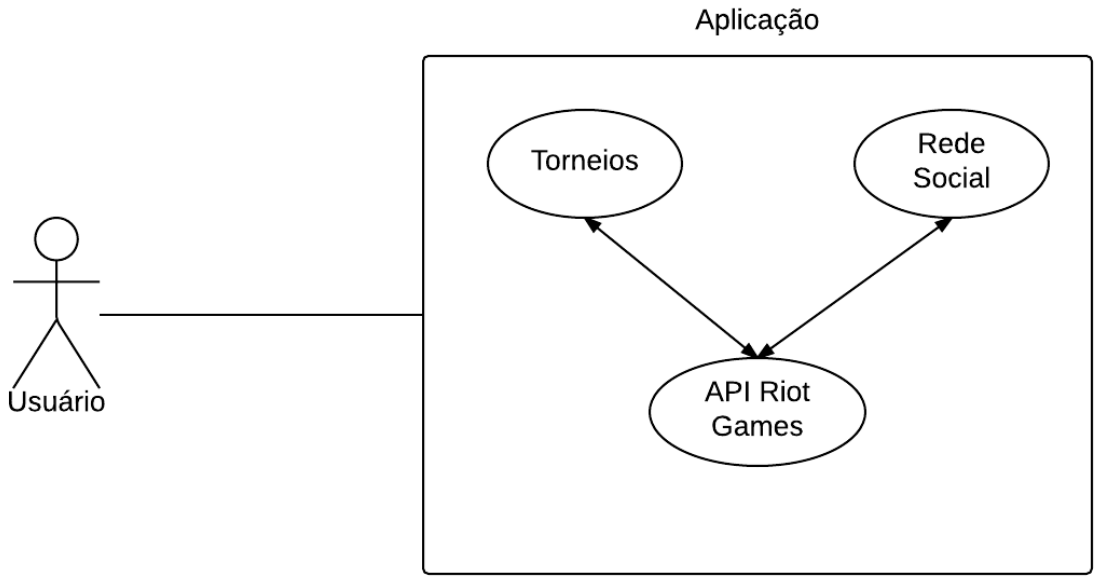

Figura 1 – Interações Possíveis do Sistema
Fonte: Autoria Própria

2 REFERENCIAL TEÓRICO

Conforme Fiani (2006) sugere, deve-se analisar cada situação como um jogo, para que possamos compreender de uma forma geral a lógica de cada decisão, tornando os fatores determinantes das decisões mais bem compreendidos, possibilitando assim a comparação com casos semelhantes.

Apesar do foco deste trabalho apontar para um jogo *on-line,* para a rede social e o gerenciador de torneio, existem desafios em função deste esporte eletrônico ter tomado proporções astronômicas na última década, deixando de ser apenas um *hobby* e se tornando cada vez mais um esporte de alta competitividade e com grandes premiações.

2.1 TIPOS DE JOGOS

Um dos primeiros tipos de jogos a ter grande aceitação do público, o RTS (*Real Time Strategy*) pode ser definido basicamente como um estilo de jogo em que se simula uma estratégia militar, como por exemplo os jogos Age of Empires, StarCraft, Warcraft III. Com fatores de decisão como gerenciamento de recursos, tomadas de decisão baseadas em aspectos incertos, como o posicionamento dos inimigos, noção espacial e temporal avançadas, aprendizagem e modelagem do oponente, planejamento de estratégias em tempo real e colaboração entre os jogadores, forma-se um universo complexo, dependente de vários fatores, onde o jogador mais qualificado e mais experiente tem de processar todas estas decisões em pouco tempo de raciocínio para conseguir as vitórias (BURO, 2003).

Derivados dos jogos RPG (*Role Playing Game*) de tabuleiro, como por exemplo *Dungeons and Dragons* e Vampiro: A Máscara, os jogos MMORPG (*Massive Multiplayer Online Role Playing Game*), foram extremamente impactantes no avanço e prosperidade dos jogos eletrônicos. MMORPG foi o primeiro estilo de jogo onde houve a possibilidade real de interagir cooperativamente ou em confrontos com outros jogadores, além de comércio de itens e a simulação de uma sociedade organizada, com a presença de aldeões, mercadores e outras entidades controladas pelo computador. De acordo com Nascimento e Cordeiro (2008), nos jogos MMORPG o usuário é transportado a um mundo, assumindo o controle de um

personagem interpretado de acordo com as regras do jogo. Devido ao cenário complexo do sistema, o jogador se prende ao personagem de acordo com sua evolução na jornada, levando o jogador a criar um laço emocional com o personagem.

Outro jogo extremamente popular entre os entusiastas da área são os FPS (*First Person Shooter*). FPS é um gênero de jogo que exige mais nível técnico que os outros jogos convencionais, por se basearem na realidade virtual. Este tipo tem como objetivo simular a realidade, paisagens com o ponto de vista do olhar do personagem, fazendo a alusão de que as imagens projetadas no monitor são o que está sendo visto pelos olhos do personagem (VENTURA, 2009).

Considerando o fato de que segundo a revista Forbes[6], os jogos do estilo MOBA são os mais jogados do mundo, tendo como *League of Legends* o detentor do maior número de horas jogadas mensalmente pelos internautas. Ainda, segundo a *Riot Games*, *League of Legends* detém 70 milhões de jogadores cadastrados, uma média de mais de 1 bilhão de horas jogadas por mês, picos diários de 3 milhões de pessoas jogando simultaneamente e um total de 12 milhões de jogadores ativos pr dia[7].

Motivados por estes números e pela grande popularidade do jogo entre universitários, a empresa *Riot Games* criou até mesmo grupos no Facebook destinado a torneios universitários. O grupo League Universitários-BR[8] possui atualmente mais de 162 membros, onde cada membro representa uma universidade diferente. Algumas universidades representadas no grupo possuem grande renome, como por exemplo, todos os campus da UTFPR, UFPR, UFRJ, USP e Unesp. Segundo os administradores do grupo, que são chamados de *rioters*, denominação utilizada para um funcionário da *Riot Games*, a intenção é trazer em breve um torneio universitário a nível nacional.

Este grupo provou ser uma ótima forma dos organizadores manterem contato e a troca de informações, com a conexão de um membro com os demais da comunidade. Apesar de não se conhecerem fisicamente e estarem distantes, os membros constituem uma comunidade que permanece unida e ativa (LARANJEIRA,

---

[6]Disponível em: http://www.forbes.com/sites/johngaudiosi/2012/07/11/riot-games-league-of-legends-officially-becomes-most-played-pc-game-in-the-world/ acesso em: 07/09/2014

[7] Disponível em: http://riot-web-static.s3.amazonaws.com/images/news-br/Outubro-2012/2012_10_31_Infografico/infogr%C3%A1fico-BR.jpg acesso em: 07/09/2014

[8] Disponível em: https://www.facebook.com/groups/467828653273084 acesso em: 10/10/2014

PORTO e PINHEIRO, 2013). Estes mesmos autores identificaram outro fator a ser observado é a interação de uma comunidade relacionado às regras de comportamento dentro de um jogo que são facilitadas para não falantes de inglês.

2.2 MOBA

No estilo MOBA, temos como principais jogos *League of Legends* e DotA2. *League of Legends* possui diariamente 27 milhões de jogadores e mais de 7,5 milhões estão *online* ao mesmo tempo. Em contrapartida, o jogo mais jogado da *steam*, DotA2 possui 624 mil jogadores em simultâneo (MIRANDA, 2014).

O atrativo dos MOBA são as inúmeras oportunidades e estratégias existentes, assim como o exercício do intelecto e da capacidade de reação rápida. De acordo com Costa (2013), em uma partida habitual, um jogador controla um personagem e se aproxima de outro, buscando uma abertura na defesa, procurando o melhor momento para atacar e em contrapartida pensando nas consequências do ataque, em caso de falha e em uma defesa rápida no caso de ser contra atacado. Estes campeões, ou personagens, são controlados por jogadores reais. Este jogador sabe as capacidades do personagem que enfrenta, mas não sabe a capacidade e características do jogador que o controla.

As singularidades de jogos MOBA estão principalmente em termos e padrões de comunicação. Além do mapa, que basicamente é o mesmo padrão para qualquer MOBA. Em termos de comunicação temos algumas abreviações, derivadas da língua inglesa. Os principais termos utilizados são descritos na

Tabela **1** abaixo.

Lane	Linha ou seção do mapa em que o jogador se encontra
MIA (*Missing in Action*)	Jogador ausente em sua *lane.*
TOP	Linha ou seção superior do mapa.
BOT	Linha ou seção inferior do mapa.
MID	Linha ou seção central do mapa.
KDA	Média de Abates, Mortes, Assistências (*Kill, Death, Assist*).

Tabela 1 - Nomenclatura comum em MOBAs

Fonte: Autoria Própria

2.2.1 Defence of the Ancients

Defence of the Ancients, conhecido como DotA, foi o primeiro MOBA popular do gênero. Ainda tendo seu gênero definido como ARTS (*Action Real Time Strategy*), surgiu em 2004 e se instaurou como modelo base para qualquer MOBA.

A popularização de DotA e sua vasta opção de heróis disponíveis definiu um padrão de jogabilidade e de mapa, este padrão segue presente na maioria dos MOBA existentes.

Outro padrão definido por DotA é o número de integrantes em cada time, usualmente os times são compostos por 5 jogadores, que podem criar uma partida juntos, ou encontrar jogadores aleatórios nas plataformas do jogo.

O objetivo de DotA é a destruição da base inimiga. Cada time possui uma estrutura, representado na Figura 2 pelas estruturas semicirculares presentes nas extremidades da figura, que deve ser defendida. A destruição desta estrutura resulta na derrota do time que sofreu a perda. Para um time conseguir avançar até tal estrutura, certos objetivos devem ser alcançados. Existem três acessos a esta estrutura central, um acesso na rota central, um acesso na rota superior e um acesso na rota inferior. Esta estrutura central é defendida por três torres em cada rota, representadas pelos círculos na Figura 2. A cada ciclo de alguns segundos, tropas controladas pelo computador são liberadas. Estas tropas confrontam tropas igualmente geradas pelo time oposto, caso não exista nenhuma tropa inimiga ao alcance, as tropas atacam os heróis mais próximos de seu caminho. Torres, além de eliminar as tropas neutras possuem função de atacar jogadores próximos em determinadas situações e ações. Após a destruição das torres de uma rota, o acesso está liberado a esta estrutura.

Dividindo a imagem (Figura 2) em dois triângulos, conforme a linha pontilhada, podemos observar a divisão do espaço virtual de cada time. A rota superior (*TOP*) é representada pela faixa a esquerda nos dois triângulos, a faixa central (*MID*) cruza o mapa em uma única linha, sendo a *lane* mais curta do jogo. A rota inferior, é representada pelas faixas inferiores nos dois triângulos.

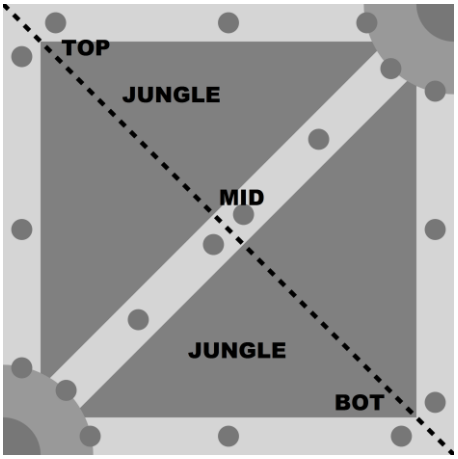

Figura 2 - Representação genérica de um mapa MOBA
Fonte: Adaptado de: wikimedia.org

Objetivos opcionais, tais como habilidades passivas obtidas destruindo tropas que não se encontram nas rotas, também chamadas de tropas neutras por não possuir ligação com um time, que são localizadas na *jungle*, representada na Figura 2 pelos triângulos presentes entre as *lanes*. O abate destas tropas garante ao personagem efeitos adicionais, que podem implicar em uma opressão constante e eficaz no adversário.

Figura 3 - Mapa DotA
Fonte: devilesk.com

Observando o mapa apresentado na Figura 3, podemos facilmente compreender onde está definida cada *lane*.

O jogo em si é definido em duas fases, a fase de rota (*lane phase*), onde os jogadores confrontam somente seus adversários diretos de rota e os confrontos generalizados, onde geralmente ocorrem após a queda de uma ou mais torres em uma ou mais rotas. Estes confrontos ocorrem, em geral, pela perda da zona de conforto dos jogadores. A falta de uma torre coloca os jogadores em risco frente a ataques surpresas do adversário, impossibilitando a fuga até a torre mais próxima e retardando o avanço até a torre inimiga.

2.2.2 League of Legends

League of Legends, popularmente chamado de LoL, tornou-se o jogo mais jogado do mundo (FORBES, 2012) por um fator simples, o fácil acesso ao conteúdo. Ao contrário de DotA, onde em sua primeira versão o jogador necessitava possuir o jogo Warcraft III e sua expansão instalados, para depois instalar a modificação DotA e usar um programa customizado para encontrar outros jogadores online e disputar as partidas, LoL surgiu com a facilidade de possuir sua plataforma própria e o mais importante, sua plataforma é gratuita.

Nome do Jogo	**Total de horas por mês (EUA/UE)**
League of Legends	1,292,502,456
World of Warcraft	622,378,909
Minecraft	371,635,651
Heroes of Newerth	184,520,156
Diablo III	172,907,605
Battlefield 3	171,852,550
MapleStory	165,503,651
StarCraft II	163,980,293
World Of Tanks	145,702,931

Quadro 1 - Comparativo Entre os Principais Jogos
Fonte: FORBES – 2012

O atrativo de ter sido desenvolvido por antigos idealizadores de DotA trouxe os fãs do jogo a esta nova plataforma emergente. *League of Legends* surgiu com o sistema de nivelamento, onde a experiência de jogo de cada jogador era comparada e o mecanismo buscador de partidas procurava jogadores com experiência

semelhante a do usuário, apesar deste método diminuir o rigor de acordo com o passar do tempo em uma fila, ainda se provou eficaz e garantiu o equilíbrio de partidas na maior parte dos casos. Outro grande atrativo deste jogo emergente, foi a implementação de um sistema de punição a jogadores tóxicos. Jogador tóxico, segundo a própria *Riot Games* é a definição de um jogador que usa de linguajar ofensivo, pratica abusos *in-game*, racismo, homofobia, xenofobia, jogadores que se estressam com facilidade e jogadores que abandonam as partidas.

De acordo com a *Riot Games*[9], "sabemos que a comunidade rejeita linguagem homofóbica, racista e sexista. Estamos vendo esse tipo de linguagem em menos de 3% de partidas globalmente e quando ela aparece, é imediatamente denunciada por jogadores, e ações são tomadas".

Este jogo atualmente é um dos mais jogados do planeta, temos os níveis de jogadores divididos pelos chamados *tiers*, que são divisões e subdivisões dentro de um mesmo nível. Os *tiers* são organizados em 7 divisões distintas, sendo elas, Bronze, Prata, Ouro, Platina, Diamante, Mestre, Desafiante.

As divisões Bronze, Prata, Ouro, Platina e Diamante possuem 5 subdivisões. Cada jogador deve alcançar 100 pontos para ser elegível a divisão superior. Após alcançar 100 pontos, o jogador realiza mais 3 jogos, se o mesmo obtiver 2 vitórias em 3 jogos, ele avança de subdivisão. Ao chegar na divisão de nº 1 de seu *tier*, o jogador deve alcançar 100 pontos e vencer 3 jogos, entre 5 para ascender ao *tier* superior, mudando assim o seu *ranking*.

Alguns times possuem três jogadores, como o caso do mapa *Twisted Treeline*, do jogo *League of Legends*. Outros possuem cinco jogadores, como é o caso do mapa Summoner's Rift. Estas variações de mapa geram um desafio no registro de times, ou equipes, além da própria geração dos torneios, nivelados, com o menor impacto possível na distribuição dos níveis. Este desafio deve ser contornado através de uma implementação de tabelas intermediárias no banco de dados. Outro desafio encontrado, é o gerenciamento de propagandas, deve-se definir a melhor estratégia para a inclusão das propagandas e de seu gerenciamento. Ainda no gerenciamento de torneio, um algoritmo em PHP (Hypertext Pre Processor) é o responsável em gerar torneios do tipo *bracket* adaptáveis ao número de jogadores, gerando o menor impacto possível na

[9] Disponível em: http://br.leagueoflegends.com/pt/news/game-updates/player-behavior/valores-de-design-comportamento-do-jogador?ref=rss acesso em: 09/10/2014

distribuição. Na parte de torneios, também necessita-se de uma implementação de níveis do jogador.

A mecânica de jogo de LoL é similar a de DotA. Porém, *League of Legends* possui outros mapas disponíveis, como por exemplo o mapa *Twisted Treeline*, que foi desenvolvido especificamente para dois times compostos por três jogadores cada. O mapa principal (*Summoner's Rift*) é igualmente dividido em três rotas espelhadas, rota superior, inferior e central, onde ocorrem os confrontos isolados e a *lane phase*. Algumas particularidades são notadas, como por exemplo a divisão dos jogadores, usualmente situando-se como um jogador, chamado de invocador, na rota superior, um jogador na rota central, dois jogadores na rota inferior e um jogador na selva, rota alternativa de *League of Legends*. Ocasionalmente este jogador invade uma rota furtivamente, com o objetivo de causar opressão de *farm* (negar o inimigo o acesso as tropas inimigas, impedindo a obtenção de ouro por este jogador) e usar-se da vantagem numérica temporária para o abate do campeão inimigo. Campeão, é a definição do personagem escolhido pelo jogador.

Após o término da *lane phase*, disputas generalizadas ocorrem, em geral em torno de um objetivo principal ou secundário. Entende-se como objetivo principal o foco na destruição de torres, inibidores ou do Nexus, que é a torre principal a ser defendida.

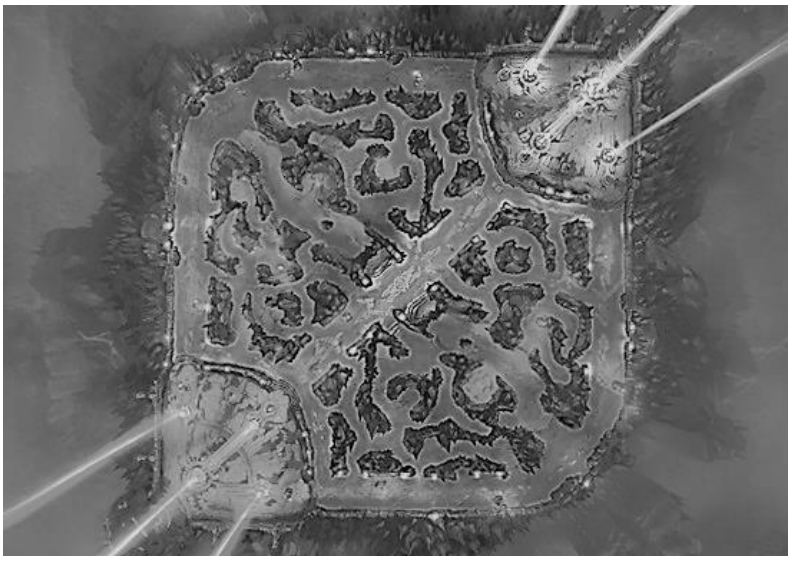

Figura 4 - Mapa de *League of Legends*
Fonte: http://leagueoflegends.wikia.com

A distribuição de torres em *League of Legends* ocorre da seguinte forma, cada rota possui três torres, cada torre possui uma força, vida e defesa maior de acordo com o nível da torre. A primeira torre é a que possui menos vida, defesa e

força. Esta torre está localizada na parte exterior da rota. A segunda torre, que possui atributos mais fortes que a anterior, fica exatamente na metade da rota, esta torre só pode ser atacada se a primeira torre estiver destruída. A terceira torre, que possui atributos de muito mais força e defesa que as anteriores está localizada na entrada da base, esta torre protege o inibidor, uma estrutura que será abordada mais a frente. Existem mais duas torres, igualmente mais fortes que a terceira torre. Estas torres protegem o Nexus, que também será abordado mais a frente neste texto. A eliminação de uma torre gera ouro para todos os jogadores do time que destruiu este objetivo, além de retirar o inimigo da sua zona de conforto, pois caso o inimigo avance, existe uma lacuna muito maior para o *jungler* de seu time efetuar um *gank*.

Inibidores são estruturas que quando destruídas, geram ao time que efetuou este objetivo, tropas muito mais fortes que as tropas padrão. Estas tropas forçam o time que perdeu a estrutura a defender constantemente esta rota afetada, pois as novas tropas geradas derrotam facilmente as tropas padrão. A destruição de um inibidor somente pode ser efetivada caso a torre que o defenda esteja destruída. Esta estrutura é a única a ressurgir após certo tempo. Quando um inibidor está destruído, o acesso as duas torres defensoras do Nexus é liberado, destruindo-se estas duas torres, o acesso ao Nexus é liberado.

O Nexus é a estrutura central que o time deve defender, a perda do Nexus resulta na derrota do time dono desta estrutura. O acesso ao Nexus só é possível se todas as torres de uma rota, incluindo-se as duas torres que o defendem e um ou mais inibidores forem derrubados. O ressurgimento dos inibidores nega o acesso ao Nexus.

Objetivos secundários do jogo podem ser definidos como objetivos de superioridade em habilidades passivas, obtidas na selva, ou no rio, uma rota neutra que divide indiretamente o mapa em áreas de domínio. Na selva, cada tropa neutra proporciona um efeito passivo ao jogador. Efeitos como regeneração de mana (barra azul que diminui de acordo com o uso de magias do personagem) e dano real (dano que não pode ser defendido por resistência mágica ou física) são habilidades passivas que qualquer jogador pode obter, derrotando a tropa neutra correspondente a esta habilidade, outras tropas geram efeitos somente para o *jungler*, estes efeitos facilitam a passagem deste jogador pela selva, com habilidades passivas como retorno de dano em tropas neutras, incapacitação temporária de

tropas e visão de sentinelas detectoras inimigas. Para que o jogador posicionado na selva possa obter uma destas habilidades passivas, é necessário que este execute a tropa neutra com o feitiço “Golpear”, que causa dano verdadeiro (dano real) a uma tropa neutra ou tropa inimiga.

No rio exitem dois monstros a serem derrotados, o dragão, que quando derrotado concede aos jogadores uma habilidade passiva progressiva, que se fortalece de acordo com o número de dragões derrotados, além de ouro recebido para todo o time que o executou. Existe também um monstro chamado de Baron Nashor, este monstro, quando derrotado garante ao time que executou este monstro uma habilidade passiva que fortalece os jogadores por um período de tempo e igualmente garante ouro a toda a equipe que o derrotou. Todos os monstros da selva, assim como os do rio possuem um tempo de retorno, para que o mesmo esteja disponível a outro abate futuro.

Outro objetivo secundário e extremamente importante é o posicionamento de sentinelas detectoras pelo mapa. As sentinelas detectoras são itens que garantem visão de uma determinada área do mapa por um determinado tempo, qualquer jogador pode posicionar uma sentinela e é função de todos o posicionamento das mesmas. O posicionamento de uma sentinela próximo a sua rota pode identificar uma abordagem de algum jogador, na intenção de praticar um *gank* em sua lane.

Com os objetivos explicados podemos entender a complexidade da mecânica de jogo presente em *League of Legends*, esta dinâmica complexa é o fator que atrai tantos jogadores ao jogo. A habilidade de um jogador é medida de acordo com a sua resposta rápida ao cenário de inúmeras variáveis e seu foco em objetivos, além de sua habilidade individual com o campeão selecionado e sua habilidade em grupo, com o time.

Em 2011 ocorreu o primeiro campeonato mundial de *League of Legends*, um dos primeiros pioneiros no estilo de jogo MOBA. A final do primeiro campeonato ocorreu na Suécia, de 18 a 20 de junho, contando em sua maioria com times norte-americanos e europeus. A premiação total oferecida foi de US$ 99.500, com a premiação do campeão sendo o valor de US$ 50,000. As transmissões das partidas finais, que foram todas no formato de *rematch* triplo, alcançaram 1,6 mi de usuários em sites de *streams online.*

Figura 5 - Final da Primeira Temporada de *League of Legends* (2011)
Fonte: curse.com

Em 2012, o número de espectadores presenciais no campeonato mundial aumentou significativamente. O jogo ganhou um *status* de esporte eletrônico e foi o primeiro jogo a tornar realmente popular fora dos países asiáticos.

Figura 6 - Final da Segunda Temporada de *League of Legends* (2012)
Fonte: forbes.com

O aumento exponencial de jogadores e fãs do esporte fez que *League of Legends*, em seu campeonato mundial de 2013 teve um crescimento no número de espectadores durante todo o campeonato. O público total nos jogos das finais foi um total de 13.000 pessoas assistindo presencialmente e, segundo o portal eletrônico

Polygon (2013), mais de 32 milhões de espectadores assistindo em tempo real via *streams online*.[10]

Figura 7 - Final da Terceira Temporada de *League of Legends* (2013)
Fonte: redbull.com

As finais de 2014 ocorreram no estádio *Sangam Stadium*, que recebeu jogos da copa do mundo de futebol de 2002. O público presente no estádio foi de 45 mil espectadores. O número total de espectadores por transmissões *online* oficiais ainda não foram divulgados, mas, apesar do horário demasiado tarde para o público em geral, as *streams* alcançaram 2 milhões de espectadores somente no portal chinês Douyu e 1 milhão no portal Twitch.[11]

[10] Disponível em: http://www.polygon.com/2013/11/19/5121688/league-of-legends-season-3-world-championship-finals-viewers acesso em: 04/11/2014.
[11] Disponível em: http://www.dailydot.com/esports/riot-world-championships-finals/ acesso em: 04/11/2014.

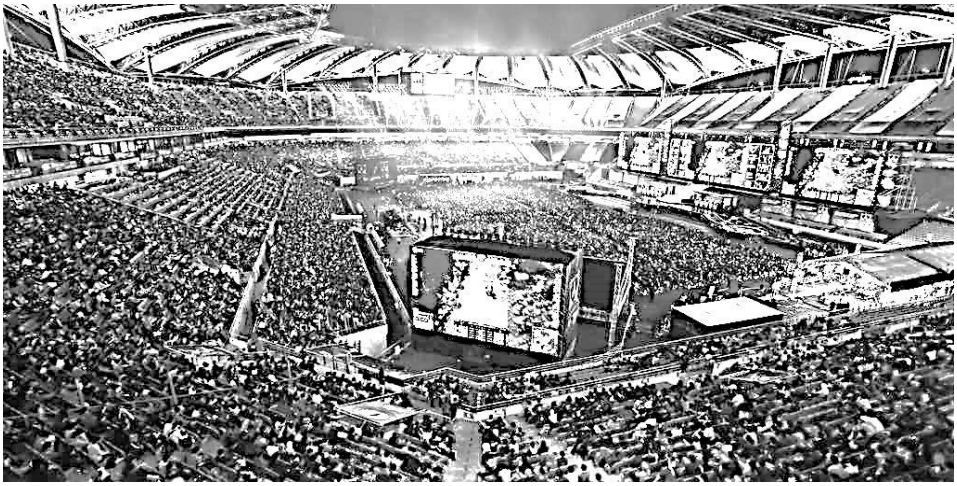

Figura 8 - Finais da Quarta Temporada de *League of Legends* (2014)
Fonte: dailydot.com

2.3 FORMAS DE COMPETIÇÃO

Os principais torneios nos cenários competitivos são os de eliminação simples, eliminação mista, com *rematch* triplo e *rematch* quíntuplo e pontos corridos.

De acordo com Bierman e Fernandez (2011), define-se como regra da maioria com duas escolhas, a escolha de dois candidatos, feitas por eleitores ímpares, onde deve-se obter a maioria dos votos para o candidato consagrado vencedor. Visto que há somente dois candidatos e um número ímpar de votantes, não há possibilidade de empate entre os candidatos.

“Em teoria dos jogos repetidos finitos, os jogadores, ou times envolvidos sabem com antecedência o número de jogos ou situações repetidas que encontrarão até o fim das interações. Essa informação influencia as estratégias e abordagens de um jogador, ou time” (TAVARES, 2009).

A forma de competição por pontos corridos, determina que em um sistema de pontos ganhos, é vetada a alteração das regras do torneio após o início do mesmo e define-se o sistema de ganho dos pontos em três pontos para vitória e um ponto para empate, no caso de partidas desportivas específicas, onde podem ocorrer empates. Em caso de empates de pontos, o critério de desempate é definido pelo regulamento específico da competição[12].

[12] Disponível em: http://www.fpf-pe.com.br/assets/uploads/138564787574.pdf acesso em: 11/10/2014

Em seu regulamento para o campeonato brasileiro de futebol do ano de 2014, o sistema de pontos corridos, ou pontos ganhos, acontece de acordo com turnos e returnos, seguindo o critério de desempate pela ordem de maior número de vitórias, maior saldo de gols, maior número de gols pró, confronto direto, menor número de cartões vermelhos recebidos, menor número de cartões amarelos recebidos e sorteio[13].

Baseando-se nos dois sistemas de pontos corridos presentes em torneios de futebol, temos a estrutura responsáveis por armazenar os confrontos, que podem guardar resultados diversos para futuras alterações em modelos de torneios e implementação do modelo por pontos corridos sem nenhuma repetição de confronto, ou com uma repetição.

Temos como natural das pessoas a competitividade, onde “goste-se ou não, a competição e a concorrência são a alma e o grande motor do desporto e da vida” (REVERDITO, SCAGLIA, *et al.*, 2006).

A competição estimula pessoas. Se realizada de forma saudável, onde todos, teoricamente possuem chances iguais de vencer. Competir em um time não é depender somente de um indivíduo, mas sim do conjunto de indivíduos trabalhando ao máximo para alcançar um mesmo objetivo. A equipe que se mantém unida, com boa comunicação e entrosamento, além de boas habilidades individuais possui muito mais chances de se tornar vencedora unindo o melhor de cada componente.

2.3.1 Eliminação Simples

Em um torneio de eliminação simples, as chaves são geradas aleatoriamente, de acordo com o número de participantes. Após, ocorre uma primeira fase de confrontos, onde, o perdedor de cada confronto é automaticamente eliminado. Terminada a primeira fase de confrontos, em que metade dos participantes foram eliminados, ocorre uma nova rodada de confrontos, onde sucessivamente perdedores vão sendo eliminados até chegarmos a fase final, onde somente dois adversários se enfrentam, definindo assim o vencedor.

[13] Disponível em: http://cdn.cbf.com.br/content/201403/1932371953.pdf acesso em: 11/10/2014

Torneio Misto – Tipo 1

Seguindo a fórmula muito utilizada em esportes eletrônicos. No primeiro modelo, as fases iniciais continuam sendo eliminação simples, as fases de quartas de final e semifinal são decididas em uma melhor colocação de três confrontos diretos entre as mesmas equipes e as finais são decididas em uma melhor colocação de cinco confrontos diretos entre as mesmas equipes.

Torneio Misto – Tipo 2

Em uma segunda forma, os participantes são divididos em grupos, cada equipe enfrenta os demais de seu grupo em um confronto simples, onde uma equipe enfrenta todos os adversários de seu grupo. Os melhores colocados avançam para a etapa de eliminação em chaves. A eliminação de chaves ocorre disputando-se a melhor colocação em três jogos contra o mesmo adversário para as quartas de finais e semifinais, a final é decidida em cinco partidas disputadas contra o mesmo adversário. O time que obter três vitórias é considerado campeão.

2.3.2 Pontos Corridos

O modelo de torneio por pontos corridos caracteriza um modelo mais extenso de torneio, pois neste modelo todos os times batalham contra todos. O time com mais vitórias consagra-se campeão. Este modelo já foi utilizado em várias competições profissionais esportivas. No Brasil, a competição de futebol mais famosa que utiliza este sistema é o Campeonato Brasileiro, descrito na seção 2.3. A vantagem deste modelo é garantir que o melhor time, em um critério geral, será consagrado campeão, pois todos os times se enfrentam, assegurando assim, um nível de equilíbrio entre os confrontos que não ocorre no sistema de eliminação simples. A desvantagem deste modelo é a quantidade de confrontos. Tendo como base o Campeonato Brasileiro, onde ocorrem confrontos duas vezes por semana, entre ao menos quatro equipes simultaneamente, temos um torneio que se estende por meses. Tornando assim um torneio demasiadamente longo. Não podemos descartar

este modelo de torneio, por ser uma das formas mais equilibrada de confrontos possíveis, porém, no sistema este torneio não será o modelo padrão.

2.4 REDE SOCIAL ON LINE

Define-se rede social uma aplicação, ou serviço web que permita que um indivíduo construa um perfil público ou parcialmente público (BOYD e ELLISON, 2007). Uma rede social é constituída de elementos e atores. Estes atores possuem entre si os chamados laços, que podem ser de vínculo afetivo, parentesco ou qualquer outra forma de vínculo (RODRIGUES e MUSTARO, 2006).

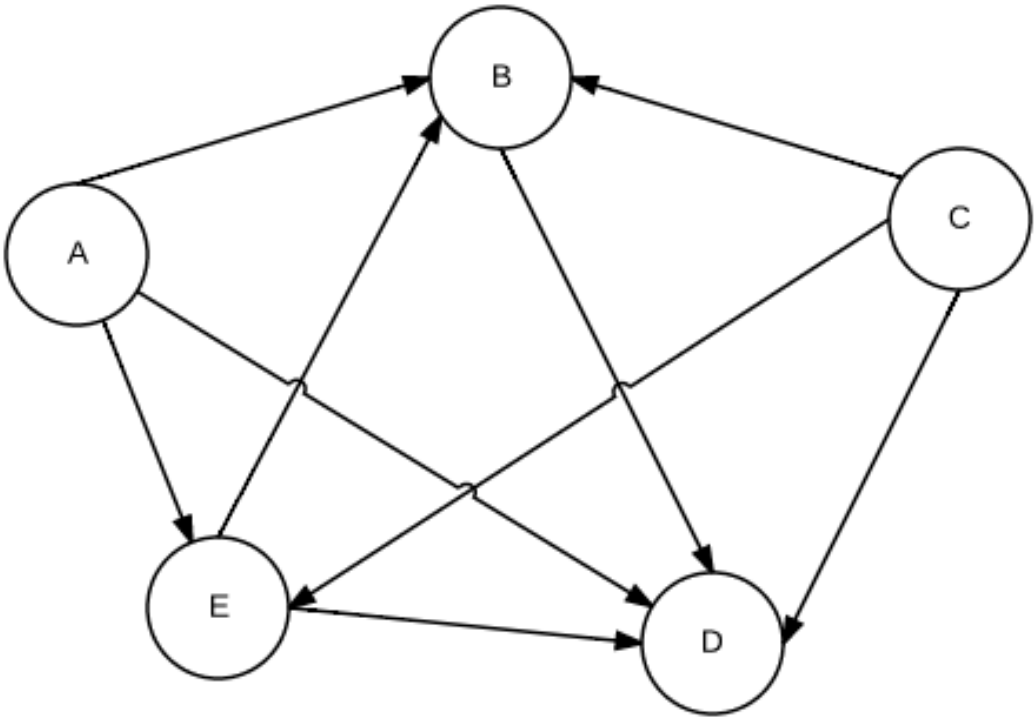

Figura 9 - Laços de Amizade em uma Rede Social
Fonte: Adaptado de Raquel Recuero

Define-se como agente em uma rede social um indivíduo que é a causa de um efeito, uma ação, pessoa ou coisa que age. Uma ação ou efeito pode ser entendido pela interação entre pessoas, elementos textuais e/ou elementos audiovisuais. Este termo vem sido utilizado em várias pesquisas distintas, cada qual com sua definição específica de acordo com o contexto de sua função (PAIVA,

2011). Em redes sociais agentes e atores são termos utilizados para as pessoas que participam ativamente da dinâmica social *online.*

Uma rede social deve seguir um foco de atuação. Cada rede social tem um tipo de interação, redes sociais como a *Ello*, tem por objetivo ser limpa, compacta, sem publicidade e funcional. A rede social *Twitter* é um exemplo de como redes sociais podem ser aplicadas a jogos, torneios, informações de jogadores, notícias do cenário esportivo são publicadas e difundidas em larga escala em uma rede social com suporte a postagens com poucos caracteres, porém todas elas têm um ponto em comum. Segundo Recuero (2009) “É característico das redes sociais na Internet, sua capacidade de difundir informações através das conexões existentes entre os atores”.

Tratando-se de um ambiente público, o compartilhamento de conhecimento e informações é constante, visto que pessoas gostam de compartilhar o que sabem (TOMAÉL e MARTELETO, 2006). Ainda segundo Tomaél e Marteleto (2006), a disposição de compartilhar e o compartilhamento eficiente de informações entre atores de uma rede geram uma melhoria intelectual e social entre os participantes envolvidos, pois cada participante passa a ter acesso a novas informações e assim podem promover um crescimento mútuo.

Neste processo de tomada de decisão, os agentes minimamente cognitivos, podem se comunicar, serem modificados pelas informações as quais foram expostos bem como modifica-las(PAIVA, 2011). Agente cognitivo é a definição de algo relativo a conhecimento, “podendo ser representado por um perfil, personalidade, comunicação ou aprendizado” (PAIVA, 2011).

Encontrar jogadores no início de um jogo pode afetar significativamente a qualidade da experiência. Em um jogo como MOBA, as relações sociais, como por exemplo, de membros de um mesmo time, ou de amizades podem melhorar o desempenho e experiência dentro do jogo(LOSUP, et al).

2.5 SISTEMAS SIMILARES

Foram analisadas as duas maiores plataformas voltadas para torneios amadores.

Estas plataformas não possuem um foco direto a algum jogo, ou alguma forma de nivelamento dos participantes, o que é o grande diferencial para este trabalho.

Dentre as ferramentas estudadas foi realizada uma análise de pontos importantes, para avaliar a concorrência e suas funcionalidades.

O critério de análise escolhido foi encontrar as ferramentas mais acessadas a nível nacional e internacional, onde possa ser realizado um estudo de frequência de ocorrências de novos torneios e os padrões existentes de torneios, tais como premiações e formas de recrutamento.

As opções mais parecidas com o presente trabalho foram Arena e-Sports e CHALLONGE!.

2.5.1 Arena e-Sports

A Arena e-Sports[14] possui a versatilidade de suportar vários jogos, ter um portal com notícias e possuir suporte para as línguas portuguesa, inglesa e espanhola. Até o momento em que foi analisada haviam sido criados 1677 torneios para os jogos Starcraft II: Wings of Liberty, Starcraft II: Heart of the Swarm, *League of Legends*, DotA2, FIFA, CS:GO, HearthStone, Heroes of The Storm. Os pontos importantes analisados foram principalmente a abrangência para diversos jogos, mas também a falta de atualizações na seção de notícias, os torneios serem regulamentados exclusivamente pelo administrador do torneio, não haver uma seção destinada as regras do torneio, os torneios serem constituídos somente por eliminação simples e não haver nenhum tipo de nivelamento dos jogadores.

2.5.2 CHALLONGE!

A CHALLONGE![15] é uma ferramenta capaz de gerar praticamente qualquer tipo de torneio e sem muitas complicações. A interface é amigável e intuitiva, fornecendo várias opções como, por exemplo, limitar o número máximo de participantes, incluir ou não as partidas de disputa do terceiro colocado, realizar *check in* dos

[14] Disponível em: http://arenaesports.com/ acesso em 10/11/2014.
[15] Disponível em: http://challonge.com/ acesso em: 10/10/2014

participantes. A aplicação trata os inscritos somente por participantes ou times, tornando uma ferramenta capaz de gerar torneios independente de plataforma.

3 DESENVOLVIMENTO

No processo de desenvolvimento deste tema, foi realizado o estudo de várias ferramentas, comparando-as ao modelo proposto e tentando-se encaixar as medidas analisadas ao escopo do projeto. Durante o desenvolvimento do projeto, foi analisada uma série de desafios e problemas de implementação. O primeiro desafio encontrado é a adequação de um modelo SQL (Structed Query Language) que seja adequado à todos os tipos de torneios possíveis, mantendo uma flexibilidade para futuras alterações. O segundo desafio encontrado, ainda na base de dados, foi a melhor forma possível de enquadrar os modelos de times.

Na modelagem do sistema, foram avaliados os requisitos do sistema, funcionalidades, assim como o estudo da melhor aplicabilidade dos modelos propostos de torneios.

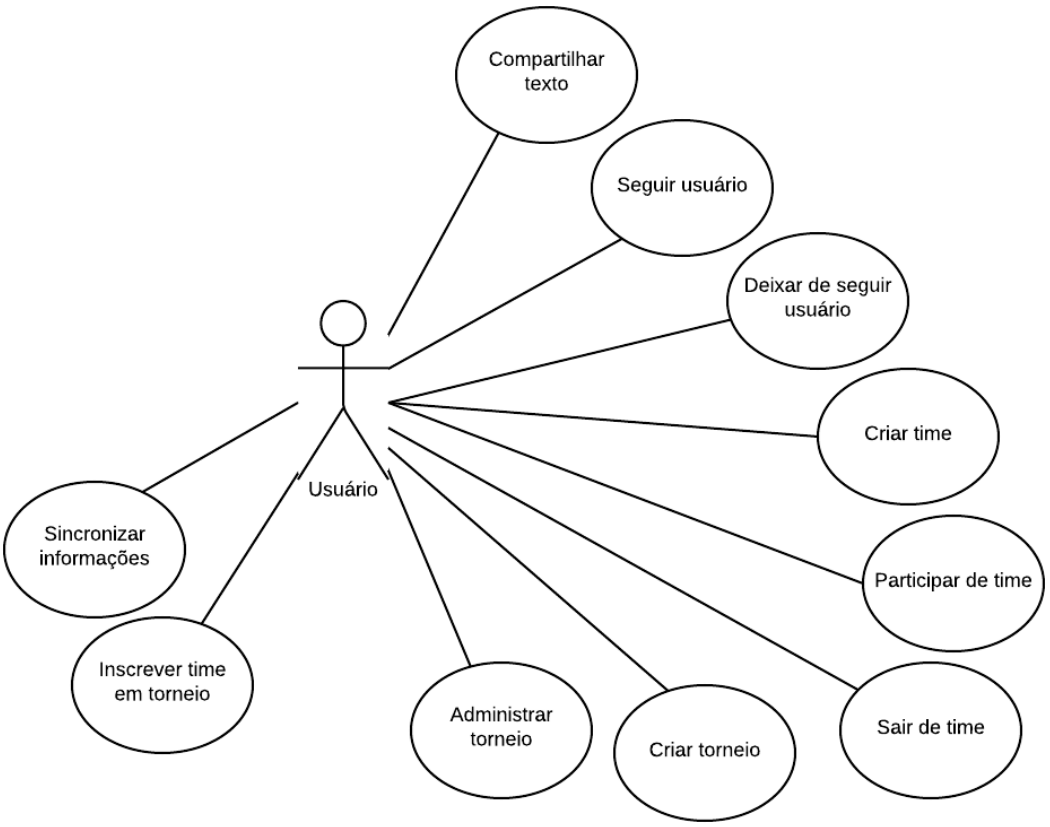

Figura 10 - Modelo de Casos de Uso
Fonte: Autoria Própria

3.1 TECNOLOGIAS WEB

A definição das linguagens web a serem utilizadas foi realizada considerando: a facilidade de aplicação da linguagem, juntamente com a sua gama de recursos fornecidos e custo. Durante o desenvolvimento, optou-se pelo uso de tecnologias gratuitas.

3.1.1 PHP

PHP, *Hipertext Preprocessor*, é uma linguagem dinâmica, de interpretação livre, utilizada originalmente para atuar no lado do servidor em uma aplicação. O PHP é interpretado pelo servidor, de acordo com as requisições do usuário, retornando uma página estática HTML para o usuário. Inserindo códigos PHP dentro de marcações HTML. É gratuito e de código aberto, licenciado pela *PHP License*, tendo apenas restrições ao uso do termo PHP[16].

Surgido em 1994, iniciou-se como um pacote de programas, para substituir scripts utilizados em Perl por Rasmus Lerdorf. Inicialmente possuía o nome de Personal Home Page, sendo alterado posteriormente para o acrônimo recursivo Hypertext Preprocessor. PHP é uma linguagem modularizada, ideal para o uso em servidores web. Alguns de seus módulos, armazenados em um repositório público, são integrados a linguagem em versões posteriores e, além disto possui suporte a praticamente todos os sistemas operacionais do mercado, tornando-a uma excelente opção.

3.1.2 JavaScript

JavaScript é uma linguagem de programação interpretada, originalmente concebida para ser parte dos navegadores, para que os mesmos possam interpretar scripts no lado do cliente, interagindo com o usuário sem a necessidade de comunicação com o servidor e sem atualizações de página. Atualmente, o JavaScript é a principal linguagem utilizada no lado do cliente em todos os navegadores[17]. Originalmente, a

[16] Disponível em: http://php.net/license/ acesso em: 15/09/2014.
[17] Disponível em: http://www.w3schools.com/js/ acesso em: 15/09/2014.

linguagem era chamada de *LiveScript*, pertencente ao navegador Netscape e posteriormente teve seu nome mudado para JavaScript em um anúncio em conjunto com a Sun Microsystems[18]. JavaScript permite que o usuário realize tarefas, ou tenha uma melhor apresentação do site através de transições ou funcionalidades que seriam impossíveis de serem realizadas no lado do servidor, sem a necessidade de recarregar a página e assim, perder informações já inseridas.

3.1.3 JSON

JavaScript Object Notation é um formato de intercâmbio de dados computacionais. JSON é um objeto JavaScript, mas não necessita do JavaScript para suas notações. JSON possui uma vantagem sobre o XML, pelo fato de as informações serem de mais fácil manipulação, pois é mais fácil escrever um analisador para JSON do que para XML (*eXtensible Markup Language*). No caso do PHP, existe uma função nativa (json_decode()) capaz de codificar arrays em JSON, ou decodificar um objeto JSON para um array. Esta função foi utilizada no projeto, pois a API fornecida pela *Riot Games* retorna as consultas no formato JSON. No
Quadro 2 existe um comparativo entre o retorno de uma requisição em JSON e em XML.

{"themirror": { "id": 443884, "name": "The Mirror", "profileIconId": 713, "revisionDate": 1416978497000, "summonerLevel": 30 }}	<summoner id=”themirror”> <id>443884</id> <name>The Mirror</name> <profileIconId>713 </profileIconId> <revisionDate>1416978497000 </revisionDate> < summonerLevel>30 </ summonerLevel> </summoner>

Quadro 2 – Comparativo de uma requisição através da Riot Games API em JSON e XML
Fonte: developer.riotgames.com

[18] Disponível em: http://pt.wikipedia.org/wiki/JavaScript acesso em: 16/09/2014.

3.1.4 JQuery

JQuery é uma biblioteca JavaScript responsável por simplificar os scripts *cliente-side*. Possui funcionalidades responsáveis pela diminuição do código dentro dos scripts. Suas funcionalidades incluem implementações de CSS e AJAX. Abaixo, temos um comparativo entre uma operação de captura do valor de um elemento em JavaScript (a direita) e em JQuery (a esquerda).

var x = $('#Teste').val( );	var x = document.getElementById('Teste').value;

Quadro 3 - Comparativo entre JQuery e JavaScript
Fonte: Autoria Própria

3.1.5 Bootstrap

O *framework* Bootstrap[19] fornece ao desenvolvedor uma série de recursos, em CSS (*Cascade Style Sheet*), JavaScript, componentes a serem utilizados, como por exemplo modelos de painéis, tabelas, barras de navegação, e imagens para serem utilizadas. Com o uso deste *framework* e suas possibilidades de customização, tanto por uma folha de estilos externa quanto a compilação do conjunto totalmente customizado é possível um ganho no desenvolvimento do *front-end*, pois além de já existirem diversas funções em JavaScript presentes e uma vasta gama de classes CSS disponíveis, seu layout é portátil, não necessitando a customização para plataformas móveis, como *smartphones* e *tabets.*

3.1.6 Riot Games API

A API fornecida pela *Riot Games* tem por objetivo a obtenção de dados relativos a jogadores, times, *rankings* de *tier*, histórico de partidas e de perfil de uma forma confiável e segura. A API foi desenvolvida atendendo à pedidos da comunidade de jogadores e desenvolvedores, para uma melhor troca de experiências de jogos e contribuindo com a experiência dos jogadores, pois através

[19] Disponível em: getbootstrap.com acesso em: 04/12/2014

dos dados obtidos nesta API, é possível organizar informações em gráficos, tabelas ou de qualquer forma planejada pelos desenvolvedores.

3.2 BANCO DE DADOS

Durante a concepção da aplicação, definiu-se a necessidade do uso de uma Base de Dados preparada para a web e com suporte à linguagem PHP. Pela grande popularidade da ferramenta, opções de segurança e integração, como formas de requisições via PHP, foi escolhido o SGBD (Sistema Gerenciador de Banco de Dados) MySQL. Possui a facilidade de configuração, criação de tabelas com atributos primários em série, ou seja, com ordenação incremental automática, não necessitando a criação manual de *triggers* ou *stored procedures*.

Na Figura 11 temos um MER (Modelo Entidade Relacionamento) de todas as tabelas do banco de dados.

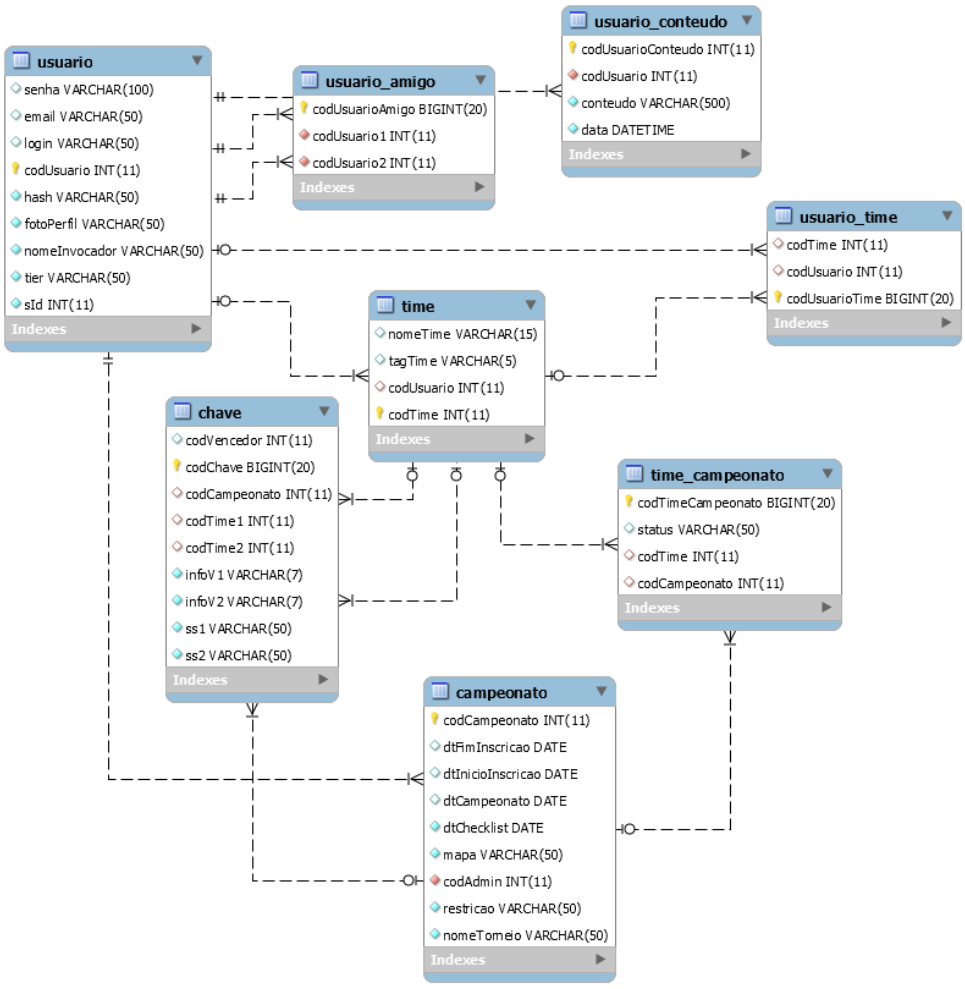

Figura 11 - Modelo Entidade Relacionamento
Fonte: Autoria Própria

Para um melhor entendimento, podemos dividir o MER em pequenos grupos, a Figura 12 apresenta os relacionamentos entre o usuário e suas possibilidades na rede social, tais como listagem de seguidores e conteúdo compartilhado.

Figura 12 - Relacionamento de Usuário e Rede Social
Fonte: Autoria Própria

O relacionamento entre usuário e times ocorre conforme a Figura 13. Onde um time possui um usuário como administrador e uma tabela intermediária com as informações dos usuários pertencentes a este time.

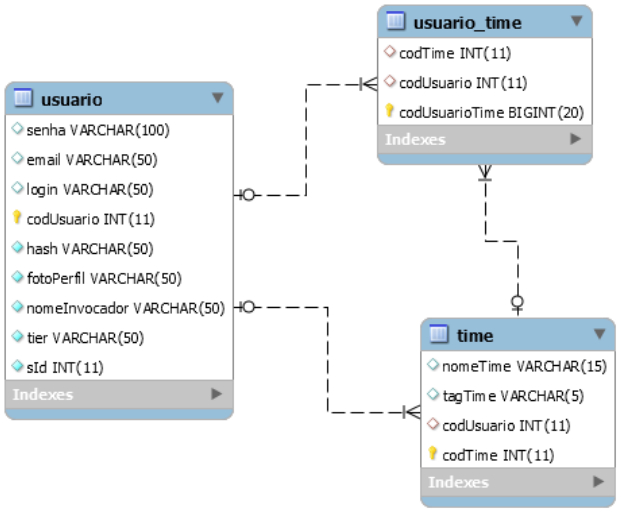

Figura 13 - Relacionamento de Usuário e Time
Fonte: Autoria Própria

Por fim, o relacionamento entre times e torneios ocorre conforme a Figura 14. Nesta figura podemos observar o relacionamento entre times e torneios. A tabela campeonato possui uma tabela intermediária chamada time_campeonato. Esta tabela intermediária armazena o código do time que se inscreveu no torneio, caso o mesmo esteja qualificado a participar, e o código do torneio. Estas informações são necessárias para gerar a chave de um torneio.

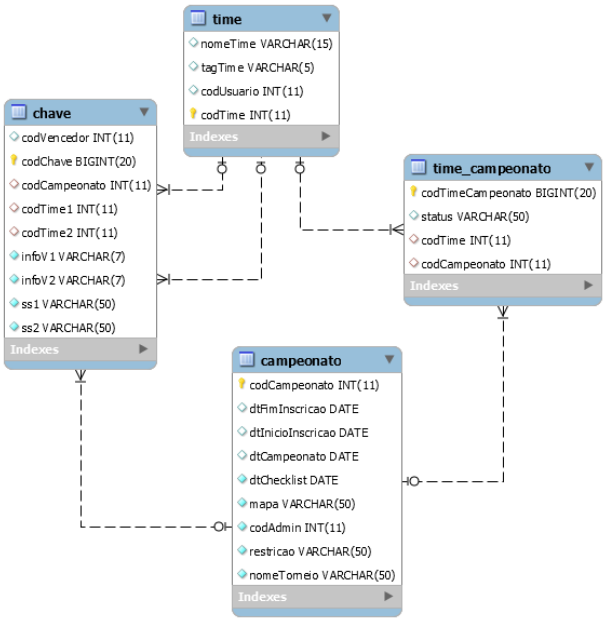

Figura 14 - Relacionamento Entre Times e Torneios
Fonte: Autoria Própria

3.3 IMPLEMENTAÇÃO

Durante o processo de elaboração e aplicação do projeto, buscou-se a simplicidade dos elementos visuais. Baseando-se em redes sociais minimalistas, como o *Twitter* e a rede social *Ello, a*través do *framework* Bootstrap[20], este objetivo foi alcançado. Foi possível a criação de elementos rapidamente, sem a necessidade da criação de vários estilos para os elementos HTML. O próprio *framework* torna possível a customização das classes, através da adição de elementos CSS com os modelos já existentes. O ponto forte da escolha do Bootstrap é a agilidade na criação de menus, formulários, janelas com flutuação sobre a aplicação principal, seja para *feedback* ou

[20] Disponível em: http://getbootstrap.com/ acesso em: 20/10/2014.

formulários embutidos com um visual agradável e inspirado no visual da rede social *Twitter*. O resultado final da aplicação é uma interface como a tela de *login* (Figura 15).

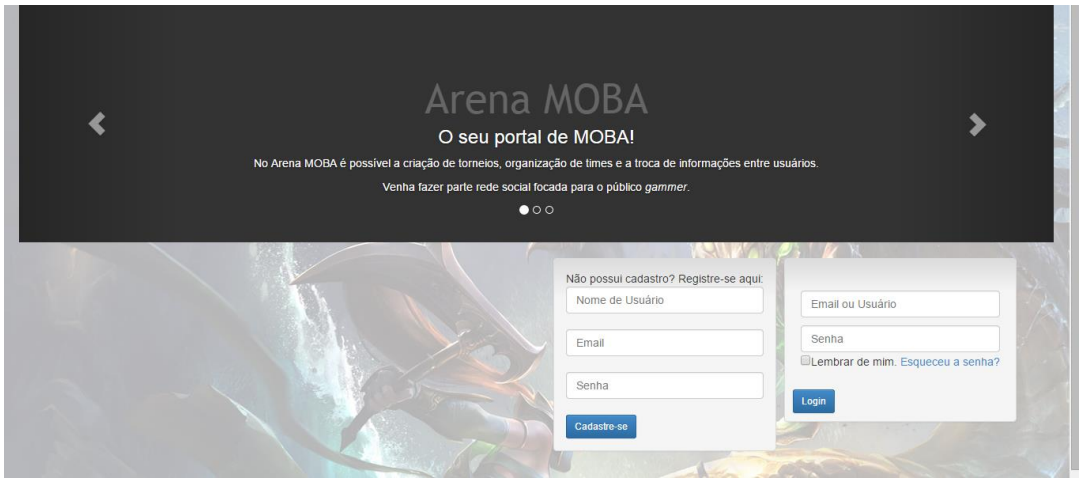

Figura 15 - Interface de Login
Fonte: Autoria Própria

Na tela de boas vindas (Figura 16), temos um espaço destinado ao usuário para compartilhar textos com seus seguidores, além das informações de seu perfil, como Nome de Invocador e Divisão, que são obtidos através da sincronização de seu perfil com a API fornecida pela *Riot Games*. A tela de boas vindas segue um padrão conforme a Figura 16.

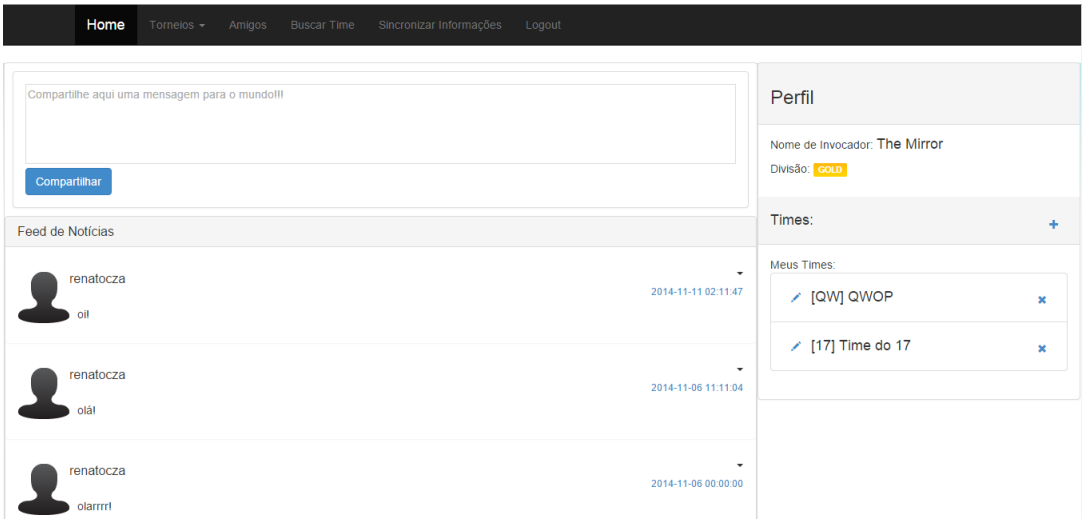

Figura 16 - Tela principal do módulo redes sociais da arena moba.
Fonte: Autoria Própria

As informações de nome de invocador, código do invocador e divisão estão na mesma tabela no banco de dados. As informações são inseridas ou atualizadas através do formulário acessado através do botão “Sincronizar Informações”. Após inseridas, são disponibilizadas na página principal do sistema.

Ao lado esquerdo, estão disponíveis as informações compartilhadas por pessoas que o usuário está acompanhando. Qualquer informação de qualquer usuário pode ser acompanhada. Basta seguir o usuário desejado. No Quadro 4 está o trecho de código responsável pela apresentação do conteúdo compartilhado pelo usuário e pelas pessoas que estão sendo seguidas pelo usuário.

```
<?php
        $res = mysql_query($query);
        while ($linha = mysql_fetch_array($res)) {
        $login = $linha['login'];
        $mensagem = $linha['conteudo'];
        if (@$linha['fotoPerfil'] == "")
        $foto = "img/user.png";
        else
        $foto = $linha['fotoperfil'];
        $codUser = $linha['codusuario'];
        $data = $linha['data'];
        $codCont = $linha['codUsuarioConteudo']
        ?><div class="panel">
        <div class="panel-heading">
        <h5 class="panel-title">
        <a href="<?php echo "user.php?cod=$codUser"; ?>">
        <img src="<?php echo $foto; ?>" width="64px" height="64px" class="pull-left">
        </a>
        <a href="<?php echo "user.php?cod=$codUser"; ?>" class="top">
        <?php echo $login; ?>
        </a>
        <a href="#"><span class="caret pull-right"></span></a></h5>
        <small>
        <a class="pull-right" href="<?php echo $codCont; ?>">
        <?php echo $data; ?>
        </a>
        </small>
        </div>
        <div class="panel-body panel-usuario">
        <p class="panel-postagem"><?php echo $mensagem; ?></p>
        </div>
        </div>
```

Quadro 4 - Linhas de código responsáveis pelo conteúdo apresentado
Fonte: Autoria Própria

A aplicação da API fornecida pela *Riot Games* se dá através de uma requisição de um *hyperlink*[21], onde as informações são retornadas no formato JSON

[21] Disponível em: https://br.api.pvp.net/api/lol/br/v1.4/summoner/by-name/nome?api_key =APIKEY acesso em: 12/10/2014

para a aplicação. No Quadro 5 podemos ver o trecho de código responsável por obter o nome de invocador e o *tier* do jogador.

```
require_once 'componentes/php-riot-api.php';
$riotApi = new riotapi('br');
$nomeInvocador = $_POST['summonerName'];
$summoner = $riotApi->getSummonerByName($nomeInvocador);
$name = str_replace(" ", "", strtolower($nomeInvocador));

if ($summoner != null) {
   $summonerId = $summoner[$name]['id'];
   $summoner = $summoner[$name]['name'];

   $url = 'https://br.api.pvp.net/api/lol/br/v2.5/league/by-summoner/' . $summonerId .
'/entry?api_key=APIKEY';
   echo $url;
   @$json = file_get_contents($url);

   if ($json != null) {
      @$arrTier[0] = json_decode($json)->{$summonerId}[0]->tier;
      @$arrTier[1] = json_decode($json)->{$summonerId}[1]->tier;
      @$arrTier[2] = json_decode($json)->{$summonerId}[2]->tier;
   } else {
      $arrTier = array(0 => 'UNRANKED', 1 => 'UNRANKED', 2 => 'UNRANKED');
   }
   $tier = maiorTier($arrTier[0], $arrTier[1], $arrTier[2]);


   $query = "UPDATE usuario SET nomeInvocador = '$summoner', tier = '$tier', sId =
'$summonerId' "
         . "WHERE codUsuario = $id";

   mysql_query($query);

   $retorno['tipo'] = 'success';
   $retorno['cabecalho'] = 'Legal!';
   $retorno['texto'] = "Parabéns $summoner, seus dados estão sincronizados em "
         . "nossa aplicação.\nBom Jogo!";
} else {
   $retono['tipo'] = 'danger';
   $retorno['cabecalho'] = "Ops.";
   $retono['texto'] = "Servidor malvado!\n"
         . "Não conseguimos identificar seu nome de invocador."
         . "\nTem certeza que ele está correto?";
}
```

Quadro 5 - Linhas de código responsáveis pela Requisição na API
Fonte: Autoria Própria

A requisição possui um retorno em JSON, conforme o . Utilizando-se a função json_decode() do PHP, é possível a extração das informações desejadas.

```
{"themirror": {
   "id": 443884,
   "name": "The Mirror",
   "profileIconId": 713,
   "revisionDate": 1416978497000,
   "summonerLevel": 30
}}
```

Quadro 6 - Retorno JSON de requisição
Fonte: Adaptado de developer.riotgames.com

A biblioteca gratuita php-riot-api, foi utilizada para a obtenção do nome de invocador. Esta biblioteca foi desenvolvida por um usuário do sistema fornecido pela *Riot Games* e está no quadro de bibliotecas gratuitas da API. A segunda requisição foi realizada através da função file_get_contents(), que é passada a requisição através do link[21]. A chave da API foi substituída no quadro pelo termo *APIKEY,* por motivos de segurança. Nesta requisição, é realizada a tarefa de recolher as informações a respeito da divisão do invocador. Após recolher as informações de divisão entre as três filas ranqueadas, uma função de ordenação é selecionada para que possamos obter a maior posição do invocador entre as três filas ranqueadas existentes.

O processo de definir qual a maior divisão a que o usuário pertence é necessário pelo fato de os torneios possuírem a opção de restrição de times e níveis. Este processo garante o nivelamento dos times envolvidos nos torneios, tanto para aqueles jogadores menos competitivos, que buscam um torneio para praticar ou simplesmente se divertir com seu time de amigos quanto para quem quer aprimorar e testar suas estratégias de jogo. O formulário de criação de um torneio segue o exemplo a seguir:

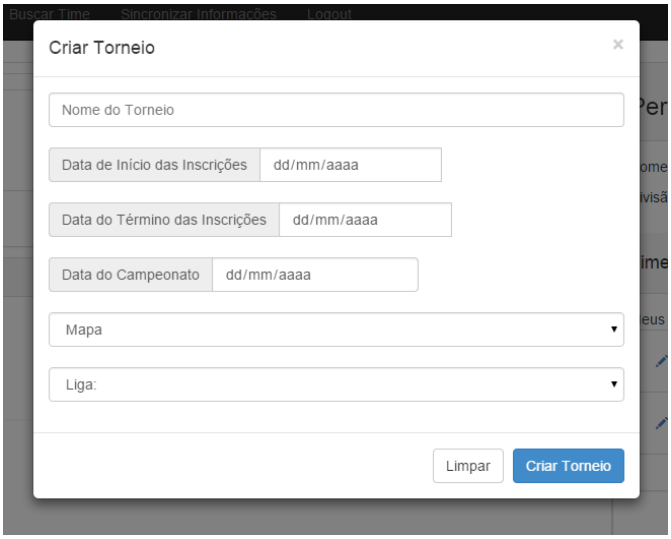

Figura 17 - Formulário de Criação de Torneios
Fonte: Autoria Própria

A operação responsável por processar o formulário é descrita conforme a Figura 17. Recebe-se as informações via *POST*, onde são armazenados em variáveis os valores de datas de início e fim de inscrições, data do campeonato, mapa selecionado e liga de restrição. A resposta da requisição feita ao banco de dados é armazenada no vetor “$retorno”, conforme o código do Quadro 7, onde é configurada uma resposta de *feedback* ao usuário, tanto para o caso de sucesso quanto para o caso de falha no sistema ou de dados inválidos.

```
            $codAdmin = $_SESSION['codUser'];
                $dtFimInscricao = $_POST['datafim'];
                $dtInicioInscricao = $_POST['datainicio'];
                $dtChecklist = $_POST['datachecklist'];
                $dtCamp = $_POST['datacampeonato'];
                $liga = $_POST['liga'];
                $mapa = $_POST['mapa'];

                $query = "INSERT INTO campeonato (dtFimInscricao, dtInicioInscricao, dtCampeonato, dtChecklist, mapa, codAdmin, restricao)"


                        . "VALUES ('$dtFimInscricao', '$dtInicioInscricao' ,'$dtCamp' ,'$dtChecklist', '$mapa', '$codAdmin', '$liga' );";


                $result = mysql_query($query);
                if (mysql_affected_rows() == 1) {
                    $retorno['tipo'] = 'success';
                    $retorno['cabecalho'] = 'Parabéns!';
                    $retorno['texto'] = "Seu torneio foi criado com sucesso. <br>Boa Sorte!";
                } else {
```

Continua...

```
...continuação

                $retorno['tipo'] = 'danger';
                $retorno['cabecalho'] = 'Ops.';
                $retorno['texto'] = "<p>Houve um erro e seu torneio não pôde ser cadastrado.</p>"
                        . "<p>Erro(s) encontrado(s):</p>"
                        . "<p>" . mysql_error() . "</p>";

                $location = "javascript:window.history.go(-1)";
    }
```

Quadro 7 - Código de registro de torneios
Fonte: Autoria Própria

Outra interface importante no sistema é de busca de um time, onde o usuário tem a visão de times disponíveis para inscrição e também a busca de times, conforme as imagens representadas na Figura 10 e Quadro 6 na primeira Figura 10, temos o *layout* da etapa de busca e exibição dos times disponíveis. No Quadro 6, temos o código responsável pela exibição das informações apresentadas no *layout*.

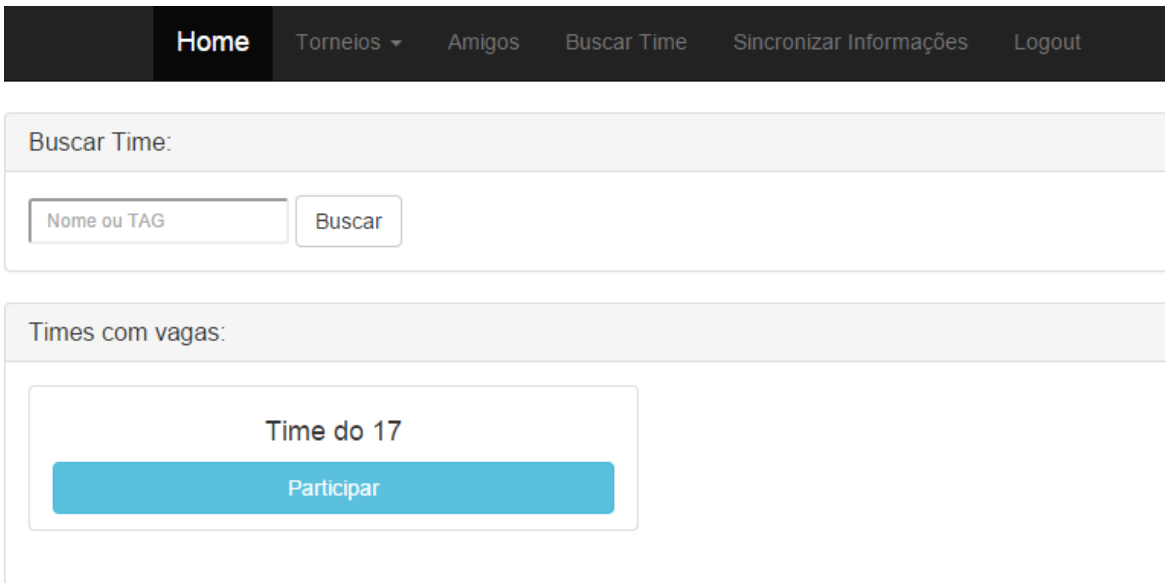

Figura 18 - Layout de Busca de Times
Fonte: Autoria Própria

A busca ocorre conforme o Quadro 6, onde as informações de times disponíveis são obtidas da base de dados e exibidas para o usuário. Ao clicar no botão “Registrar”, uma requisição POST é encaminhada e o feedback ao usuário é exibido.

```
<?php
@$time = $_POST['buscarTime'];
$query = "SELECT t.codTime, nomeTime from time t\n"
. "WHERE (SELECT count(ut.codTime) from usuario_time ut WHERE ut.codTime = t.codTime) <5";
$result = mysql_query($query);
if (mysql_affected_rows() >= 1)
while ($linha = mysql_fetch_array($result)) {

$result2 = mysql_query($query2);
if (mysql_affected_rows() >= 1)
while ($linha2 = mysql_fetch_array($result2)) {
if ($linha2['codTime'] != $linha['codTime']) {
?>
<li id="<?php echo "c$codTime"; ?>" class="list-group-item" style="width:
 30%;">
<p class="h4" style="text-align: center;"><?php echo $nomeTime; ?></p>
<a class="btn btn-info btn-block" href="javascript:registrar(<?php echo
$codTime; ?>)">Participar</a>
</li>
<?php
}
} else {


?>
<li id="<?php echo "c$codTime"; ?>" class="list-group-item" style="width: 30%;">
<p class="h4" style="text-align: center;"><?php echo $nomeTime; ?></p>
<a class="btn btn-info btn-block" href="javascript:registrar(<?php echo
 $codTime; ?>)">Participar</a>
</li>
<?php
}
}
?>
```

Quadro 8 - Códigos Responsáveis Pela Exibição do Layout
Fonte: Autoria Própria

4 RESULTADOS E ANÁLISE

No protótipo desenvolvido, simulamos um torneio gerado para 6 times, disputando um torneio em eliminação simples temos os passos descritos conforme o modelo a seguir (Figura 19).

Figura 19 - Chave com 6 Times
Fonte: Autoria Própria

Supondo no modelo acima que nos primeiros confrontos os vencedores foram o Time 1, Time 4 e Time 5, porém na plataforma colaborativa, devido a um erro na etapa de preencher qual foi o time vencedor, o Time 6 e Time 5 declararam vitória. O modelo necessitaria de uma terceira fonte de consulta, o administrador do torneio, baseado nas capturas de tela enviadas pelos dois times para o critério de desempate, conforme a regra da maioria com duas escolhas, apresentado no item 2.3, conforme a Figura 20.

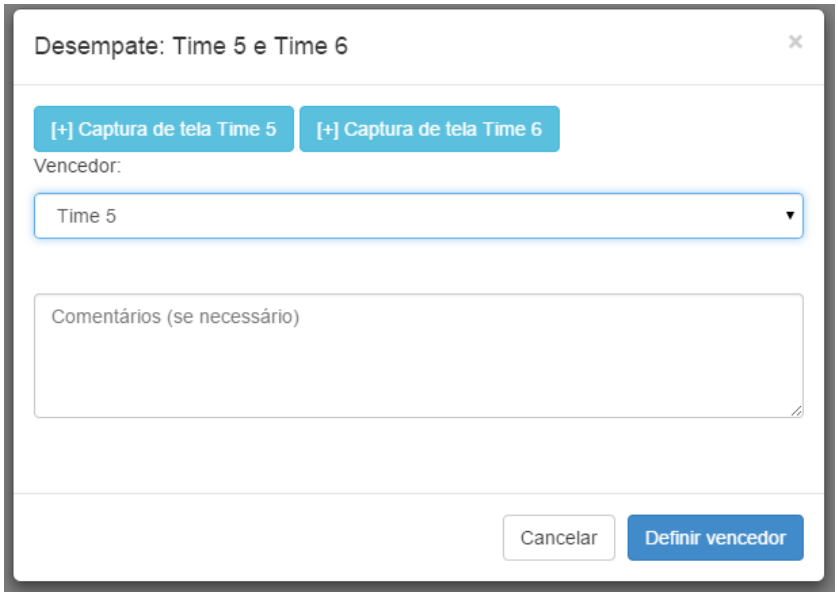

Figura 20 - Aplicação da Regra da Maioria com Duas Escolhas
Fonte: Autoria Própria

Após definido o vencedor da chave dos times 5 e 6, a tabela prossegue normalmente para a próxima interação, conforme apresentado abaixo.

Figura 21 - Segunda Rodada de Confrontos
Fonte: Autoria Própria

Supondo que o Time 1 seja vitorioso, o Time 4 decidirá contra o Time 5 a vaga para a chave final. Adotando no modelo que o Time 5 foi vitorioso contra o Time 4, este está qualificado para a chave final.

No modelo proposto, as chaves finais, com os Times 1 e 5 disputando 3 partidas, temos o Time 5 vencedor, pelo placar de 2 vitórias contra 1 derrota.

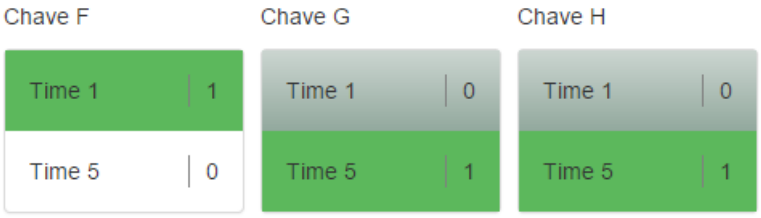

Figura 22 - Chaves Finais
Fonte: Autoria Própria

O Time 5, além da habilidade superior, pode ter se aproveitado do modelo de teoria dos jogos repetidos finitos, apresentado no item 2.3 deste documento, adquirindo uma vantagem baseada no confronto inicial, que o fez vitorioso sobre o Time 1, finalizando assim o torneio de acordo com a Figura .

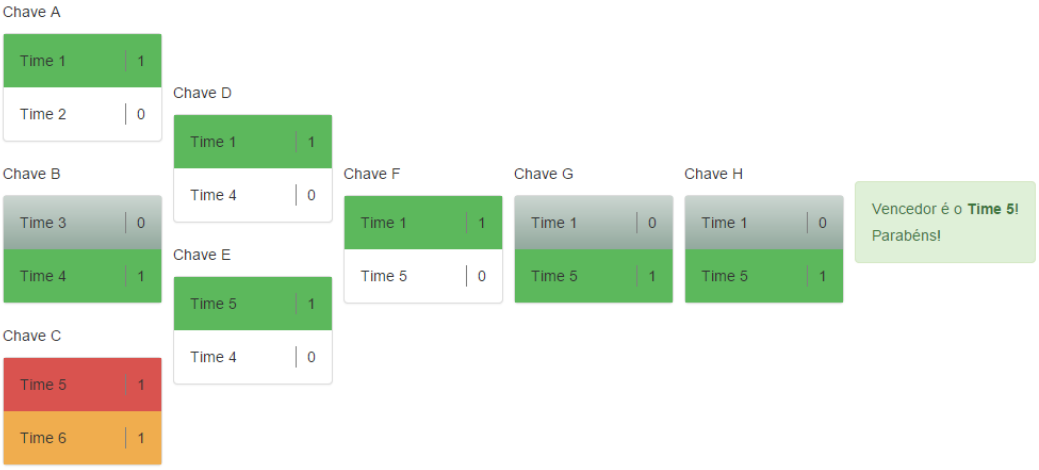

Figura 23 - Descrição do Torneio Completo
Fonte: Autoria Própria

Conforme o modelo apresentado, para um torneio pequeno, com menos de 8 times participantes, o Time 1 e o Time 5 chegaram às finais com duas partidas disputadas, enquanto o Time 4 participou de três partidas. Para torneios pequenos, o modelo ideal proposto é o de pontos corridos. Tomando como base os mesmos seis times, propondo o modelo a seguir, teríamos a seguinte estrutura. Todos os times enfrentam todos os times, em um confronto simples, conforme o Quadro 9. O time com mais vitórias é o vencedor.

	Time 1	Time 2	Time 3	Time 4	Time 5	Time 6	Vitórias
Time 1	--	x	x	x			3
Time 2		--	x	x	x		3
Time 3			--	x	x	x	3
Time 4				--	x	x	2
Time 5	x				--		1
Time 6	x	x			x	--	3

Quadro 9 - Resultados dos confrontos
Fonte: Autoria Própria

Conforme a tabela, temos as estatísticas de cada time. Os times 1, 2, 3 e 6 estão empatados com três vitórias cada. Neste caso, o critério de desempate é o proposto de acordo com o chamado KDA (*Kills, Deaths, Assists*) do time. KDA é a quantidade total de abates (*kills*), mortes (*deaths*) e assistências (*assists*) de um time. Os números expressos no KDA definem, em geral o nível de entrosamento de um time, pois, quando um time está entrosado, sua tendência é conseguir mais abates com menos mortes de seus participantes e com um número maior de assistências.

De acordo com o KDA analisado, temos a classificação considerando maior número de *kills*, menor valor de *deaths* e maior quantidade de *assists*, nesta ordem. Logo, o Time 3 é declarado vencedor, conforme o Quadro 10.

TIME	PONTOS (VITÓRIAS)	KDA	CLASSIFICAÇÃO
Time 1	3	11.5/20/25	
Time 2	3	12/15/10	Terceiro Colocado
Time 3	3	15/13/20	Campeão
Time 4	2	10/15/19	
Time 5	1	11.9/19.6/26	
Time 6	3	15/15/19	Vice-Campeão

Quadro 10 - Resultados dos confrontos
Fonte: Autoria Própria

De acordo com este modelo, cada time participa de 5 jogos, tornando a competição justa para os times. Este modelo é indicado para torneios pequenos, pelo fato de quanto mais times estão registrados, maior é o número de partidas,

tendo em vista que o número total de partidas é definida pela fórmula X = N-1, onde X é o número de partidas totais e N é o número de participantes. Considerando partidas simultâneas com duração aproximada de uma hora cada, em um dia é possível identificar o vencedor.

5 CONCLUSÕES

O objetivo principal da elaboração deste projeto foi a criação de um sistema de torneios amadores que traga uma experiência de nivelamento aos jogadores e times, para que os jogadores casuais de *League of Legends*, possam participar de torneios com chances reais de serem campeões, o foco principal do sistema é trazer à tona torneios amadores equilibrados. Além do sistema de nivelamento, o sistema inclui interfaces para a criação de torneios, sincronização de perfil e busca de amigos. Um sistema baseado no modelo das redes sociais atuais, com grande potencial de expansão e comercialização.

É uma alternativa complementar ao que existe atualmente pois jogadores amadores com *ranking* baixo ou médio tem dificuldade para encontrar oponentes para partidas em nível equilibrado.

Com o sistema de uma rede social, o usuário pode compartilhar com seus seguidores informações sobre o jogo, sobre torneios ou qualquer outra informação.

O protótipo desenvolvido atendeu aos os requisitos propostos inicialmente, onde obtivemos com sucesso o nivelamento dos jogadores e toda a parte de redes sociais funcionando perfeitamente.

Durante a elaboração do protótipo, encontraram-se diversas dificuldades que foram contornadas com sucesso em sua grande maioria, apenas não sendo possível a implementação neste ponto, a implementação de uma forma de confirmar a propriedade do nome do invocador e checagem das últimas partidas. O primeiro desafio encontrado foi a modelagem do sistema de uma forma flexível, onde o escopo completo pudesse ser utilizado em uma base de dados que comporte as informações dos torneios em apenas duas tabelas. A definição no sistema gerenciador de banco de dados sobre como se comportaria um torneio foi o principal problema enfrentado, pois o sistema deve permitir a alteração do tipo de torneio primeiramente em dois modelos: Eliminação Simples e Pontos Corridos. Porém, optou-se por o sistema gerenciar os grupos do modelo de Pontos Corridos e as chaves do sistema de Eliminação Simples em uma mesma tabela, para as consultas e atualizações, desta forma, o sistema pode ser adaptado para outras formas de torneios sem a necessidade de tabelas específicas para cada tipo de torneio. Este problema foi resolvido criando-se uma tabela intermediária que recebe as informações do torneio, o indicativo de qual time saiu vitorioso e as capturas de tela

dos dois times, além do campo de preenchimento automático, no caso dos dois confrontantes concordarem no resultado de qual time foi o vitorioso e de preenchimento manual orientado ao administrador no caso de discórdia ou dúvidas entre os dois times.

Um aspecto complicado de lidar durante a concepção do protótipo foi o objetivo de reduzir o número de requisições feitas diretamente em PHP, para evitar redirecionamentos no sistema. Como por exemplo requisições de seguir um usuário, ou deixar de seguir. Uma função em AJAX ou JQuery realiza uma requisição para um arquivo PHP e recebe a resposta (sucesso ou falha) sem a necessidade de recarregar, ou redirecionar o usuário a outra página. Em algumas funções a implementação de métodos em AJAX e JavaScript obtiveram sucesso e o retorno esperado, outros métodos não foram tão eficazes, como por exemplo uma requisição através de *JavaScript* para validações em tempo real, que algumas vezes não validavam corretamente as informações em tempo real e consequentemente, não foram aplicados, permanecendo assim os formulários e redirecionamentos em PHP.

5.1 APLICAÇÕES E ABRANGÊNCIA

O projeto desenvolvido possui como ponto forte a fácil portabilidade para qualquer outro tipo de torneio focado para o universo de jogos MOBA. Sua aplicação inicial ao sistema de torneios do jogo *League of Legends*, deu-se pelo fato deste jogo ser atualmente, o mais jogado do mundo. Porém, o mesmo modelo seja aplicado a outros MOBA, como por exemplo DotA2 e até outros estilos de jogos.

Existe inclusive uma possibilidade real de aplicação do escopo para tipos de jogos que não fazem parte do universo MOBA, entre estes jogos estão os jogos de tiro, de esportes onde existem formas de torneios individuais ou em grupo.

Com o sistema atual de torneios e definições de grupo, presentes em todos os jogos MOBA, a aplicação deste projeto em outro jogo do mesmo estilo se daria através da inclusão no banco de dados as tabelas que receberiam as informações sobre os jogos, sobre os mapas existentes quando for o caso e também o tamanho de cada time relacionado ao jogo.

5.2 TRABALHOS FUTUROS

Devido ao escopo do projeto, existem várias possibilidades de continuidade do mesmo. Pequenos fatores, como a substituição dos formulários convencionais pelos formulários chamados via JavaScript tornam o ambiente mais agradável aos olhos do usuário e possibilitam o uso de mais funcionalidades.

Existem possibilidades de um sistema de nivelamento relacionado diretamente a qual fila ranqueada o usuário pertence, sendo um nivelamento para o mapa *Twisted Treeline*, outro para o mapa *Summoner's Rift*, baseado na fila respectiva ao desempenho individual, ou em dupla do jogador, e a fila respectiva ao desempenho do jogador em times respectivos ao mapa *Summoner's Rift*. Outra possibilidade é o uso do armazenamento de dados reais de partidas, usando-se a API fornecida pela *Riot Games*, garantindo assim que não hajam trocas de jogadores durante o torneio, ou que um jogador registrado no time não seja o que está efetivamente naquele time. Também há a possibilidade de validação do jogador registrado, para garantir que não sejam usados jogadores falsos na plataforma.

No módulo responsável pela troca de informações do usuário, semelhante a uma rede social atual, pode ser aplicado um filtro de palavras, para a identificação de links, imagens, vídeos e nomes de usuário. Outra implementação futura, ainda nesta parte é a repetição de conteúdo de algum usuário e citação ao mesmo, através de um sistema similar ao *Twitter* de marcações.

No sistema de busca de times, pode ser implementado um sistema de sugestão, onde aparecem os times disponíveis tomando por base a quantidade de vagas e o nível geral dos participantes, similares ao nível do usuário que está realizando a busca. Assim, poderíamos criar times ainda mais nivelados, pois todos os participantes do time possuiriam aproximadamente a mesma experiência de jogo. Nenhum jogador é impedido de entrar em um time, independentemente de seu *tier*, porém o time é impedido de entrar em um torneio se um ou mais jogadores deste time possuir um *tier* maior que o determinado pelo administrador do torneio.

No sistema de torneios, podem ser implementadas outras formas de torneios.

A implementação de registro usando redes sociais como *Facebook*, *Twitter* e *Google* + será outro fator a ser implementado futuramente, além da inserção de uma

área de propagandas relacionadas a empresas especializadas ou qualquer outro link relacionado ao cenário de e-Sports, focado ao tipo de jogo MOBA.

REFERÊNCIAS

BATISTA, M. D. L. S. et al. UM ESTUDO SOBRE A HISTÓRIA DOS JOGOS ELETRÔNICOS, Juiz de Fora, 2007. Disponivel em: <http://re.granbery.edu.br/artigos/MjQ4.pdf>. Acesso em: 14 novembro 2014.

BIERMAN, H. S.; FERNANDEZ, L. **Teoria dos Jogos**. 2. ed. [S.l.]: PEARSON, 2010. 544 p.

BOYD, D. M.; ELLISON, N. B. Social Network Sites: Definition, History, and Scholarship, 2007. Disponivel em: <http://onlinelibrary.wiley.com/doi/10.1111/j.1083-6101.2007.00393.x/full>. Acesso em: 20 Setembro 2014.

BRANDÃO, L. R. G. Jogos Cinematográficos ou Filmes Interativos? A semiótica e a, 2012. Disponivel em: <http://sbgames.org/sbgames2012/proceedings/papers/artedesign/AD_Full21.pdf>. Acesso em: 26 setembro 2014.

BURO, M. Real-Time Strategy Games: A New AI Research Challenge, Edmonton, 2003. Disponivel em: <https://skatgame.net/mburo/ps/RTS-ijcai03.pdf>. Acesso em: 15 novembro 2014.

CONFEDERAÇÃO BRASILEIRA DE FUTEBOL. CAMPEONATO BRASILEIRO DA SÉRIE A DE 2014: REC - REGULAMENTO ESPECÍFICO DA COMPETIÇÃO, 2014. Disponivel em: <http://cdn.cbf.com.br/content/201403/1932371953.pdf>. Acesso em: 10 outubro 2014.

COSTA, P. H. B. D. Ludus online : um estudo ludológico e social de League of Legends, 2013. Disponivel em: <http://bdm.unb.br/handle/10483/6513>. Acesso em: 20 setembro 2014.

FEDERAÇÃO PERNAMBUCANA DE FUTEBOL. RGC - REGULAMENTO GERAL DAS COMPETIÇÕES, 15 dezembro 2010. Disponivel em: <http://www.fpf-pe.com.br/assets/uploads/138564787574.pdf>. Acesso em: 10 outubro 2014.

FIANI, R. **Teoria dos Jogos**. 2. ed. [S.l.]: Elsevier Brasil, 2006. 408 p.

LARANJEIRA, P.; PORTO, E.; PINHEIRO, P. R. D. L. Além da arena: Análise das estruturas de organização e comunicação surgidas em, 2013. Disponivel em: <http://www.sbgames.org/sbgames2013/proceedings/industria/10-full-paper-indtrack.pdf>. Acesso em: 12 Setembro 2014.

LOSUP, A. et al. Analyzing Implicit Social Networks in Multiplayer Online Games, 2014. Disponivel em: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6756709&tag=1>. Acesso em: 18 outubro 2014.

MIRANDA, N. M. V. As capacidades dinâmicas das empresas nacionais de maior sucesso no online gaming:modelo para a indústria nacional, 2014. Disponivel em: <http://repositorio.ipsantarem.pt/handle/10400.15/1023>. Acesso em: 25 outubro 2014.

NASCIMENTO, C. A. D.; CORDEIRO, R. OS JOGOS ONLINE E O COMPORTAMENTO SOCIAL, 2004. Disponivel em: <http://www.dainf.ct.utfpr.edu.br/~graeml/InfoDigital/Arquivos/OS_JOGOS_ONLINE_E_O_COMPORTAMENTO_SOCIAL.pdf>. Acesso em: 12 setembro 2014.

PAIVA, D. C. D. www.teses.usp.br/teses/disponiveis/./Tese_Daniel_Costa_de_Paiva.pdf, 2011. Disponivel em: <www.teses.usp.br/teses/disponiveis/./Tese_Daniel_Costa_de_Paiva.pdf>. Acesso em: 29 Novembro 2014.

RECUERO, R. **Redes Sociais na Internet**. [S.l.]: Sulina, 2009.

REVERDITO, R. S. et al. COMPETIÇÕES ESCOLARES: REFLEXÃO E AÇÃO EM PEDAGOGIA DO ESPORTE PARA FAZER A DIFERENÇA NA ESCOLA, 2006. Disponivel em: <http://www.revistas.ufg.br/index.php/fef/article/view/1207/3279>. Acesso em: 15 Setembro 2014.

RODRIGUES, L. C.; MUSTARO, P. N. Levantamento de características referentes à análise de redes sociais, 2006. Disponivel em: <http://www.lbd.dcc.ufmg.br/colecoes/sbgames/2006/035.pdf>. Acesso em: 01 outubro 2014.

TAVARES, J. M. **Teoria Dos Jogos - Aplicada a Estrategia Empresarial**. 1. ed. [S.l.]: LTC, v. 1, 2009.

TOMAÉL, M. I.; MARTELETO, R. M. Redes sociais: posições dos atores no fluxo da informação, 2006. Disponivel em: <http://www.redalyc.org/articulo.oa?id=14720365008>. Acesso em: 12 Outubro 2014.

VENTURA, M. A. A. Etnografia de uma comunidade de jogadores de FPS, 2009. Disponivel em: <http://repositorio-aberto.up.pt/handle/10216/60351>. Acesso em: 28 setembro 2014.

APÊNDICE A - Códigos da Página Principal

```
<?php
session_start();
if (!$_SESSION) {
    header("Location: index.php");
}
require_once 'componentes/modals.php';
?>
<!doctype html>

<html lang="en">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8">


        <meta name="description" content="Arena MOBA">
        <meta name="author" content="Renato 'The Mirror' Canizella">


        <!--[if lt IE 9]>
        <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
        <![endif]-->
        <link href="bootstrap/css/bootstrap.min.css" rel="stylesheet">
        <script src="http://code.jquery.com/jquery-latest.js">
        </script>
        <script src="bootstrap/js/bootstrap.min.js">
        </script>
        <link href="style.css" rel="stylesheet">
        <script type="text/javascript" src="jquery-1.3.2.min.js"></script>


        <script type="text/javascript">
            function deleta(cod) {
                //alert("funciona");
//                var oi = $.post("operacoes_js.php", {codTime: cod, op: "deletaTime"});
//                //location.href = "principal.php";
//
//                alert(oi);
                var posting = $.post("operacoes_js.php", {op: 'deletaTime', codTime: cod}, function (data,
status) {

                    var res = data;
                    alert('Operação realizada com sucesso!');
                    $('#c'+cod).addClass("hidden");
                });
            }
```

Continua...

...continuação.

```
</script>
      <title>Arena MOBA</title>

   </head>

   <body>
      <?php
      require_once 'componentes/menu.php';
      include 'ad/db.php';
      ?>
      <div class="bg">

         <div class="principal panel"><!-- PANEL PRINCIPAL -->

            <!--PANEL ESQUERDO-->
            <div class="panel panel-default left">

               <!-- POSTAGEM -->
               <form action="operacoes.php" method="post" class="panel panel-default espacado-4lados">
                  <input type="hidden" name="op" value="postagem">
                  <textarea class="painel-compartilhar panel-default" name="mensagem" placeholder="Compartilhe aqui uma mensagem para o mundo!!!"></textarea>
                  <br>
                  <input type="submit" class="btn btn-primary" value="Compartilhar">
               </form>

               <!-- FIM POSTAGEM -->

               <!-- FEED DE NOTÍCIAS -->

               <div class="panel panel-default">
                  <h4 class="panel-heading panel-title">Feed de Notícias</h4><br>
                  <?php
                  $id = $_SESSION['codUser'];


                  $query = "SELECT DISTINCT uc.conteudo, uc.codusuario, u.login, uc.data, uc.codUsuarioConteudo "
                        . "FROM usuario_conteudo uc "
                        . "INNER JOIN usuario_amigo ua on ua.codusuario1 = '$id' "
                        . "INNER JOIN usuario u "
                        . "ON u.codusuario = uc.codusuario "
```

Continua...

...continuação.

```
                        . "WHERE ua.codusuario2 = uc.codusuario "
                        . "OR uc.codusuario = '$id' "
                        . "ORDER BY data desc";

//                    echo $query . '<br>';

                    $res = mysql_query($query);
                    while ($linha = mysql_fetch_array($res)) {

                        $login = $linha['login'];
                        $mensagem = $linha['conteudo'];

                        if (@$linha['fotoPerfil'] == "")
                            $foto = "img/user.png";
                        else
                            $foto = $linha['fotoperfil'];

                        $codUser = $linha['codusuario'];
                        $data = $linha['data'];

                        $codCont = $linha['codUsuarioConteudo']
                        ?>
                        <div class="panel">
                            <div class="panel-heading">
                                <h5 class="panel-title">
                                    <a href="<?php echo "user.php?cod=$codUser"; ?>">
                                        <img src="<?php echo $foto; ?>" width="64px" height="64px" class="pull-
left">
                                    </a>
                                    <a href="<?php echo "user.php?cod=$codUser"; ?>" class="top">
                                        <?php echo $login; ?>
                                    </a>

                                    <a href="#"><span class="caret pull-right"></span></a></h5>
                                <small>
                                    <a class="pull-right" href="<?php echo $codCont; ?>">
                                        <?php echo $data; ?>
                                    </a>
                                </small>
                            </div>
                            <div class="panel-body panel-usuario">
                                <p class="panel-postagem"><?php echo $mensagem; ?></p>
                            </div>
                        </div>
```

Continua...

...continuação.

```
                    <?php
                }
                ?>


            </div>

            <!-- FIM DO FEED -->

        </div><!-- FIM PANEL ESQUERDO -->

        <!-- PANEL DIREITO -->
        <?php
        $query = "SELECT nomeInvocador, tier "
              . "FROM usuario "
              . "WHERE codUsuario = '$id'";
        $result = mysql_query($query);
        $result = mysql_fetch_array($result);
        $summonerName = $result['nomeInvocador'];
        $tier = strtolower($result['tier']);
        $divisão = $result['tier'];
        ?>
        <div class="panel panel-default right">
            <div class="panel-heading"><h3>Perfil</h3></div>
            <div class="panel-body">
                <p>Nome de Invocador: <span class="h4"><?php echo $summonerName;
?></span></p>
                <p>Divisão: <span class="label label-<?php echo $tier ?>"><?php echo $divisão
?></span></p>
            </div>

            <div class="panel-heading">
                <h4>Times:
                    <small>
                        <a data-toggle="modal" data-target="#modalCriarTime" href="#" class="btn btn-
sm pull-right">
                            <span class="glyphicon glyphicon-plus"></span>
                        </a>
                    </small>
                </h4>
            </div>
            <div class="panel-body">

                <ul class="list-group" id="listaTime">
```

Continua...

...continuação.

```
                    Meus Times:
                    <!--PHP AQUI-->
                    <?php
                    $query = "SELECT DISTINCT "
                          . "t.nomeTime, t.tagTime, t.codTime, ut.codUsuario "
                          . "FROM time t "
                          . "INNER JOIN usuario_time ut "
                          . "ON ut.codUsuario = '$id'"
                          . "WHERE ut.codTime = t.codTime";

//                    echo $query;
                    $result = mysql_query($query);

                    while ($linha = mysql_fetch_array($result)) {
                       $time = '[' . $linha['tagTime'] . ']';
                       $time .= " " . $linha['nomeTime'];
                       $codTime = $linha['codTime'];
                       ?>
                    <li id="<?php echo "c$codTime"; ?>" class="list-group-item bg-info">
                          <h4><a href="" class="btn-sm" data-toggle="modal" data-
target="#modalEditarTime">
                             <span class="glyphicon glyphicon-pencil"></span>
                          </a>
                          <?php echo $time; ?>
                          <a href="javascript:deleta(<?php echo $codTime; ?>)" class="pull-right btn-
sm">
                             <span class="glyphicon glyphicon-remove"></span>
                          </a>
                       </h4>
                    </li>
                    <?php
                    }
                    ?>
                    <!--fim do php-->
                 </ul>

              </div>

           </div>

        </div>


     </div>
```

Continua...

```
...continuação.

    </div>
</body>
</html>
```

APÊNDICE B - Códigos da Página de Funções e Procedimentos

```
<?php
require_once 'ad/funcoes.php';
require_once 'ad/db.php';
session_start();

$location = "";
@$id = @$_SESSION['codUser'];

switch ($_POST['op']) {
   case 'inserir':

      $login = $_POST['login'];
      $senha = criptografa($_POST['senha']);
      $email = $_POST['email'];
      $hash = criahash();
      $query = "INSERT INTO usuario (login, email, senha, hash) VALUES ('$login', '$email', '$senha',
'$hash')";

      $result = mysql_query($query);
      if (mysql_error()) {
         $retorno['cabecalho'] = "Erro:";
         $retorno['texto'] = "Não foi possível efetuar o cadastro.<br>Erro(s):<br><b> " . mysql_error() .
'</b>';
         $retorno['tipo'] = 'danger';
      } else if (mysql_affected_rows() == 1) {
         $retorno['cabecalho'] = "Parabéns";
         $retorno['texto'] = "Seu cadastro foi efetuado com sucesso.";
         $retorno['tipo'] = 'success';
      }
      break;

   case 'login':

      $login = $_POST['login'];
      $senha = criptografa($_POST['senha']);

      $query = "SELECT * FROM usuario WHERE login = '$login' and senha = '$senha'";
      $result = mysql_query($query);

      if (mysql_affected_rows() == 1) {
         $linha = mysql_fetch_array($result);

         $_SESSION['login'] = $login;
         $_SESSION['senha'] = $senha;
         $_SESSION['codUser'] = $linha['codUsuario'];
```

Continua...

...continuação.

```
            header("Location: principal.php");
        } else {
            $retorno['tipo'] = 'danger';
            $retorno['cabecalho'] = "Ops.";
            $retorno['texto'] = "<p>Não pudemos realizar o login.</p>"
                    . "<p>Você checou se o seu <b>Login</b> e <b>Senha</b> estão corretos???</p>";
        }


        break;

    case 'cCamp':
        $codAdmin = $_SESSION['codUser'];
        $dtFimInscricao = $_POST['datafim'];
        $dtInicioInscricao = $_POST['datainicio'];
        $dtChecklist = $_POST['datachecklist'];
        $dtCamp = $_POST['datacampeonato'];
        $liga = $_POST['liga'];
        $mapa = $_POST['mapa'];
        $query = "INSERT INTO campeonato (dtFimInscricao, dtInicioInscricao, dtCampeonato, dtChecklist, mapa, codAdmin, restricao)"
                . "VALUES ('$dtFimInscricao', '$dtInicioInscricao' ,'$dtCamp' ,'$dtChecklist', '$mapa', '$codAdmin', '$liga' );";

        $result = mysql_query($query);
        if (mysql_affected_rows() == 1) {
            $retorno['tipo'] = 'success';
            $retorno['cabecalho'] = 'Parabéns!';
            $retorno['texto'] = "Seu torneio foi criado com sucesso. <br>Boa Sorte!";
        } else {
            $retorno['tipo'] = 'danger';
            $retorno['cabecalho'] = 'Ops.';
            $retorno['texto'] = "<p>Houve um erro e seu torneio não pôde ser cadastrado.</p>"
                    . "<p>Erro(s) encontrado(s):</p>"
                    . "<p>" . mysql_error() . "</p>";

            $location = "javascript:window.history.go(-1)";
        }

        break;

    case 'postagem':
        $mensagem = $_POST['mensagem'];
        $id = $_SESSION['codUser'];
```

Continua...

...continuação.

```
        $data = date("Y-m-d h:m:s");
        $query = "INSERT INTO usuario_conteudo (codUsuario, conteudo, data) values ('$id', '$mensagem', '$data')";
        mysql_query($query);
        if (mysql_affected_rows() == 1) {
            header("Location: principal.php");
        } else {
            $retorno['tipo'] = 'danger';
            $retorno['cabecalho'] = 'OPS.';
            $retorno['texto'] = "Forças ocultas nos impediram de postar a sua mensagem. \nTente novamente em alguns instantes";
        }
        break;


    case 'seguir':
        $codUser = $_SESSION['codUser'];
        $codFollow = $_POST['codFollow'];
        $query = "INSERT INTO usuario_amigo (codUsuario1, codUsuario2) VALUES ('$codUser','$codFollow')";
        mysql_query($query);
        header("Location: amigos.php");

        $retorno = "";
        break;


    case 'deixar':
        $codUser = $_SESSION['codUser'];
        $codUnFollow = $_POST['codUnFollow'];
        $query = "DELETE FROM usuario_amigo WHERE codUsuario1 = '$codUser' AND codUsuario2 = '$codUnFollow'";
        mysql_query($query);
        header("Location: amigos.php");

        $retorno = "";
        break;


    case 'nSync':

        require_once 'componentes/php-riot-api.php';
        $riotApi = new riotapi('br');
        $nomeInvocador = $_POST['summonerName'];

        $summoner = $riotApi->getSummonerByName($nomeInvocador);
        $name = str_replace(" ", "", strtolower($nomeInvocador));
```

Continua...

...continuação.

```
        if ($summoner != null) {
            $summonerId = $summoner[$name]['id'];
            $summoner = $summoner[$name]['name'];

            $url = 'https://br.api.pvp.net/api/lol/br/v2.5/league/by-summoner/' . $summonerId . '/entry?api_key=APIKEY';
            echo $url;
            @$json = file_get_contents($url);

            if ($json != null) {
                @$arrTier[0] = json_decode($json)->{$summonerId}[0]->tier;
                @$arrTier[1] = json_decode($json)->{$summonerId}[1]->tier;
                @$arrTier[2] = json_decode($json)->{$summonerId}[2]->tier;
            } else {
                $arrTier = array(0 => 'UNRANKED', 1 => 'UNRANKED', 2 => 'UNRANKED');
            }
            $tier = maiorTier($arrTier[0], $arrTier[1], $arrTier[2]);

            $query = "UPDATE usuario SET nomeInvocador = '$summoner', tier = '$tier', sId = '$summonerId' "
                    . "WHERE codUsuario = $id";

            mysql_query($query);

            $retorno['tipo'] = 'success';
            $retorno['cabecalho'] = 'Legal!';
            $retorno['texto'] = "Parabéns $summoner, seus dados estão sincronizados em "
                    . "nossa aplicação.\nBom Jogo!";
        } else {
            $retono['tipo'] = 'danger';
            $retorno['cabecalho'] = "Ops.";
            $retono['texto'] = "Servidor malvado!\n"
                    . "Não conseguimos identificar seu nome de invocador."
                    . "\nTem certeza que ele está correto?";
        }

        break;

    case 'deletaTime':

        $codTime = $_POST['codTime'];
        $q = "SELECT codUsuario FROM time WHERE codTime = '$id'";
        $r = mysql_query($q);
```

Continua...

...continuação.

```
    if (mysql_affected_rows() == 1) {
        $l = mysql_fetch_array($r);

        if ($id == $l['codUsuario']) {
            $query = "DELETE FROM time "
                    . "WHERE "
                    . "codTime = $codTime";
            mysql_query($query);
            $query = "DELETE FROM usuario_time "
                    . "WHERE "
                    . "codTime = '$codTime'";

            mysql_query($query);
        } else {
            $query = "DELETE FROM usuario_time "
                    . "WHERE "
                    . "codTime = '$codTime' "
                    . "AND codUsuario = '$id'";

            mysql_query($query);
        }
    }



    break;

case 'criarTime':

    $nomeTime = $_POST['nomeTime'];
    $tagTime = $_POST['tagTime'];

    $query = "INSERT INTO time "
            . "(nomeTime, tagTime, codUsuario ) "
            . "VALUES ('$nomeTime', '$tagTime', '$id') ";

    $erro = "";
    mysql_query($query);
    if (mysql_error() != null)
        $erro = 1;

    if ($erro != 1) {

        $query = "select codTime from time where nomeTime = '$nomeTime'";
```

Continua...

...continuação.

```
            //echo $query;
            $result = mysql_query($query);

            $result = mysql_fetch_array($result);
            $codTime = $result['codTime'];
            //echo "<br>$codTime<br>";

            $query = "INSERT INTO usuario_time "
                    . "(codTime, codUsuario) "
                    . "VALUES ('$codTime','$id')";
            mysql_query($query);
            if (mysql_error() != null)
                $erro = 1;
        }

        if ($erro == "") {
            $retorno['tipo'] = 'success';
            $retorno['cabecalho'] = 'Legal!';
            $retorno['texto'] = "Parabéns, seu time foi criado.<br>"
                    . "Agora só falta chamar seus amigos e se divertir!";
        } else {
            $retono['tipo'] = 'danger';
            $retorno['cabecalho'] = "Ops.";
            $retorno['texto'] = "Servidor malvado!\n"
                    . "Não conseguimos criar o seu time.<br>"
                    . "Forças ocultas estão agindo sobre nosso servidor";
        }

        break;
}
?>
<!--
To change this template, choose Tools | Templates
and open the template in the editor.
-->
<!DOCTYPE html>
<html lang="en">
    <head>
        <?php
        if ($location == "")
            $location = 'index.php';
        ?>

        <meta charset="utf-8">
```

Continua...

...continuação.

```
      <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
      <meta name="author" content="Renato 'The Mirror' Canizella">
      <!--[if lt IE 9]>
      <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
      <![endif]-->
      <link href="bootstrap/css/bootstrap.min.css" rel="stylesheet">
      <link href="style.css" rel="stylesheet">
      <script src="http://code.jquery.com/jquery-latest.js"></script>
      <script src="bootstrap/js/bootstrap.min.js"></script>
   </head>
   <body>
      <div class="bg-index"></div>
      <div class="central panel panel-<?php echo $retorno['tipo']; ?>">
         <div class="panel-heading"><h2 class="panel-title"><?php echo $retorno['cabecalho']; 
?></h2></div>
         <div class="panel-body">
            <p><?php echo $retorno['texto']; ?></p><br>
            <a class="btn btn-default" href="javascript:window.history.go(-1)">Voltar</a>
         </div>
      </div>
   </body>
</html>
```