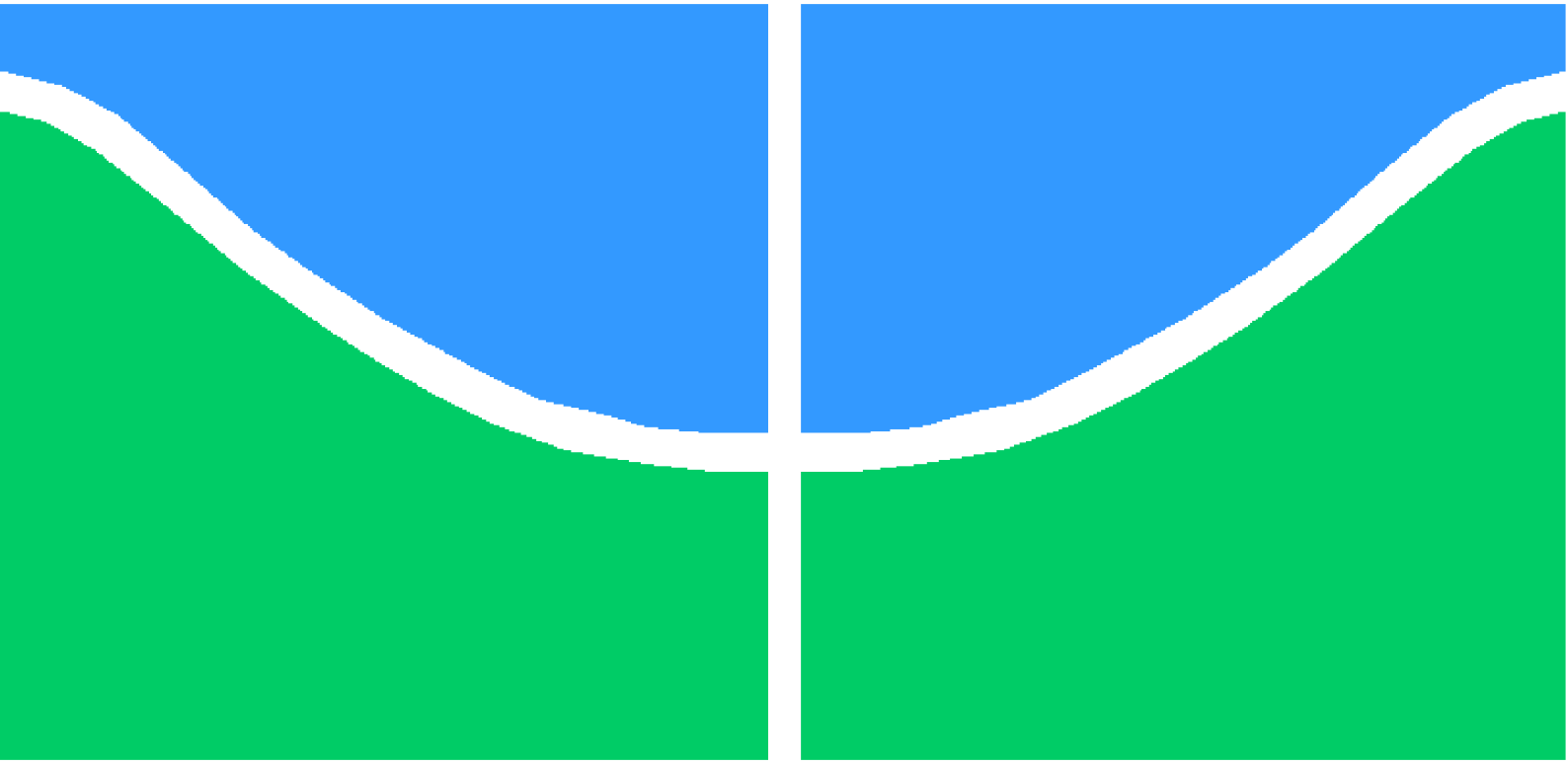

Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia de Software

# Aplicação de Sistemas MultiAgentes em Mobile Social Games

Autor: Levino Moisés Paiva Magalhães Rufino Porto
Orientador: Profa. Dra. Milene Serrano
Coorientador: Prof. Dr. Mauricio Serrano

Brasília, DF
2017

Levino Moisés Paiva Magalhães Rufino Porto

# Aplicação de Sistemas MultiAgentes em Mobile Social Games

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Profa. Dra. Milene Serrano

Coorientador: Prof. Dr. Mauricio Serrano

Brasília, DF

2017

Levino Moisés Paiva Magalhães Rufino Porto
Aplicação de Sistemas MultiAgentes em Mobile Social Games/ Levino Moisés Paiva Magalhães Rufino Porto. – Brasília, DF, 2017-
48 p. : il. (algumas color.) ; 30 cm.

Orientador: Profa. Dra. Milene Serrano

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2017.

1. Sistemas MultiAgentes. 2. Android. 3. *Mobile Social Games*. 4. Jade. I. Profa. Dra. Milene Serrano. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Aplicação de Sistemas MultiAgentes em Mobile Social Games

CDU 02:141:005.6

Levino Moisés Paiva Magalhães Rufino Porto

# Aplicação de Sistemas MultiAgentes em Mobile Social Games

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 01 de junho de 2017:

---

**Profa. Dra. Milene Serrano**
Orientador

---

**Prof. Dr. Mauricio Serrano**
Coorientador

---

**Profa. Dra. Edna Canedo**
Convidado 1

Brasília, DF
2017

# Resumo

O resumo deve ressaltar o objetivo, o método, os resultados e as conclusões do documento. A ordem e a extensão destes itens dependem do tipo de resumo (informativo ou indicativo) e do tratamento que cada item recebe no documento original. O resumo deve ser precedido da referência do documento, com exceção do resumo inserido no próprio documento. (. . . ) As palavras-chave devem figurar logo abaixo do resumo, antecedidas da expressão Palavras-chave:, separadas entre si por ponto e finalizadas também por ponto. O texto pode conter no mínimo 150 e no máximo 500 palavras, é aconselhável que sejam utilizadas 200 palavras. E não se separa o texto do resumo em parágrafos.

**Palavras-chaves**: latex. abntex. editoração de texto.

# Abstract

This is the english abstract.

**Key-words**: latex. abntex. text editoration.

# Sumário

# 1 Introdução

Jogos *multiplayer* atraem pessoas de diversos grupos sociais e faixas etárias. A possibilidade de interação com outros jogadores é um dos maiores atrativos nesse tipo de jogo. Com o avanço da tecnologia, os tipos de dispositivos que comportam tais interações estão cada vez mais acessíveis. Já é possível imergir em um ambiente virtual através de aplicativos para *smartphones*. Com a redução dos preços desses aparelhos, diversos usuários e desenvolvedores foram atraídos para a área de *Mobile Social Games*.

Neste capítulo será apresentado o contexto de inserção, e tópicos de maior relevância a esse projeto. A separação dos tópicos é apresentada através da contextualização, justificativa, questão de pesquisa e objetivos.

## 1.1 Contextualização

Todos os dias, milhões de pessoas interagem entre si em ambientes virtuais conhecidos como *Massively- Multiplayer Online Role-Playing Games* (MMORPGs)(YEE, 2006). As primeiras versões de MMORPGs eram baseadas em interações através de comandos de texto. Esse primeiro estilo de MMORPG era conhecido como *Multi-User Dungeons* (MUDs)(DIETERLE; CLARKE, 1997). O menor custo de produção de dispositivos com maior capacidade de processamento, e conexões de internet mais velozes, possibilitou que interfaces gráficas fossem adicionadas a tais jogos. No entanto, uma característica se manteve durante a evolução desses jogos, as interações sociais entre os jogadores.

As relações sociais oriundas de tais jogos se tornaram mais fáceis através da popularização de redes sociais, como Facebook e GooglePlus. Inúmeros *Social Games* utilizam-se das conexões presentes nessas redes para proporcionar uma maneira mais fácil de interação entre os usuários. Jogos como Clash of Clans(SUPERCELL, 2015) e Candy Crush Saga(KING, 2015) utilizam-se de tais redes sociais para facilitar o encontro de jogadores.

Ambos os jogos mencionados anteriormente são classificados como *Social Games*. *Social game* é um jogo *online* que utiliza serviços de redes sociais(PARK; LEE, 2012). Esses serviços são utilizados para obter dados referentes aos diferentes tipos de conexões entre os usuários. Tais redes sociais podem ser de terceiros, como as citadas anteriormente, ou podem ser gerenciadas pela própria empresa. A partir dessas observações, é possivel notar que jogos *online* são cada vez mais interações sociais do que experiências individuais(KING; BORLAND, 2003).

*Social games* podem ser divididos em dois grandes grupos, *mobile* e *desktop*. Existem diferenças entre *Social Games* de dispositivos *mobile* e *Social Games* de computadores

pessoais. Ao contrário dos *Social Games* de computadores pessoais, *Mobile Social Games* são restringidos por limitações de *hardware* e plataformas de *software*, questões referentes à usabilidade, entre outros(YAMAKAMI, 2011).

Através de observações das restrições existentes em *Mobile Social Games*, foi levantada a possibilidade de aplicação do paradigma MultiAgentes no seu desenvolvimento. Os sistemas MultiAgentes são sistemas compostos por múltiplos elementos computacionais que interagem entre si, conhecidos como agentes (WOOLDRIDGE, 2009). Uma das vantagens na utilização de agentes é que eles provêem características que proporcionam comunicação dinâmica em grupos grandes de usuários(BERGENTI; CAIRE; GOTTA, 2015). Agentes também são componentes de software que podem ser reutilizáveis, reduzindo o tempo de implementação(BERGENTI; HUHNS, 2014).

De acordo com o levantamento bibliográfico realizado, foi possível a identificação de uma única proposta de aplicação do paradigma MultiAgentes no contexto de *Mobile Social Games*. Tal proposta refere-se ao *framework* AMUSE (*Agent-based Multi-User Social Environment*) (SPA, 2015). O AMUSE é uma plataforma de código aberto, ainda em desenvolvimento, para *Social Games*. Ela possui o formato PaaS (*Platform as a Service*)(BERGENTI; CAIRE; GOTTA, 2015), uma ferramenta que possibilita provedores de serviço, como jogos e comunidades de portais, a reduzir da carga de implementação de funcionalidades comuns a vários jogos(BERGENTI; CAIRE; GOTTA, 2015).

## 1.2 Justificativa

A decisão de trabalhar uma abordagem MultiAgentes em *Mobile Social Games* foi instigada devido às características desse paradigma, dentre as quais destaca-se o fato dos Sistemas MultiAgentes proporcionarem melhorias na dinâmica da coordenação de grandes grupos de usuários, permitindo a redução de tempo no desenvolvimento da solução. Tal fato é evidenciado pelos autores Fabio Bellifemine, Giovanni Caire e Dominic Greenwood (BELLIFEMINE; CAIRE; GREENWOOD, 2007). Adicionalmente, tem-se que a redução de tempo de desenvolvimento corresponde à uma característica desejada para produção de software(BERGENTI; CAIRE; GOTTA, 2015).

Outras características presentes no paradigma MultiAgentes, e pertinentes a proposta de um trabalho no contexto de *Social Games* são: autonomia, adaptabilidade, flexibilidade, assíncronismo, colaboração e proatividade(BERGENTI; CAIRE; GOTTA, 2015).

## 1.3 Questão de Pesquisa

Esse TCC procura responder a seguinte questão de pesquisa: Como lidar com *Mobile Social Games* utilizando uma abordagem MultiAgentes?

## 1.4 Objetivos

Esta seção define os objetivos geral e específicos referentes ao trabalho.

### 1.4.1 Objetivo Geral

Investigar uma abordagem MultiAgentes para *Social Game* explorando o desenvolvimento de um *Social Game* assíncrono para plataforma *mobile* Android.

### 1.4.2 Objetivos Específicos

Os objetivos específicos abordados por esse TCC são:

- Realizar uma revisão bibliográfica visando identificar problemas comumente encontrados no desenvolvimento de *Mobile Social Games* (assíncronos);
- Acordar soluções candidatas - orientadas a Sistemas MultiAgentes - para lidar com os problemas identificados no objetivo anterior;
- Realizar prova de conceito, a qual será baseada em ciclos evolutivos de desenvolvimento, seguidos de pesquisa-ação;
- Compilar os resultados obtidos ao longo do processo investigativo usando uma abordagem híbrida quantitativa e qualitativa;
- Aprofundar os conhecimentos em Engenharia de Software, motivado pelo domínio de *Mobile Social Games*, orientando-se por padrões arquiteturais, metodologias, testes e outras boas práticas.

## 1.5 Organização dos Capítulos

Esse TCC está organizado em seis capítulos, sendo este primeiro a Introdução. Os demais capítulos são resumidos brevemente a seguir:

- **Referencial Teórico:** este capítulo explana sobre conceitos relevantes aos domínios de *Mobile Social Games*, Engenharia de Software e Sistemas MultiAgentes;
- **Suporte Tecnológico:** explica sobre as tecnologias que serão utilizadas nesse trabalho para suporte e desenvolvimento da prova de conceito. Tais tecnologias são *frameworks*, sistemas operacionais, ferramentas de controle de versão, entre outras;
- **Metodologia:** descreve a modalidade de pesquisa que será utilizada, o planejamento e a descrição das atividades que serão realizadas, e o cronograma do trabalho;

- **Proposta:** acorda detalhes da proposta de pesquisa e desenvolvimento desse trabalho;
- **Resultados Obtidos:** apresenta os resultados obtidos até o momento, e as futuras atividades a serem executadas.

# 2 Referencial Teórico

Este capítulo apresenta conceitos pertinentes aos tópicos de maior relevância abordados nesse TCC. Os tópicos abordados são apresentados através das seções: seção 2.1, apresenta a base teórica referêncial de *Mobile Social Games*; seção 2.2, apresenta o referencial teórico de Engenharia de Software; seção 2.2.1, apresenta a base teórica de Sistemas MultiAgentes.

## 2.1 Mobile Social Games

A área de *Mobile Social Games* possui diversos jogos com os mais variados estilos. Esses estilos podem ser subdivididos em dois grandes grupos: *Hard-Core Games* e *Casual Games*. Ambos possuem características em comum que os definem como *Social Games*, mas também possuem peculiaridades referentes ao tipo de interação social presente no jogo. Os *Hard-Core Games* são caracterizados por serem ambientes de extrema interação social, onde diversas oportunidades para criação de laços de amizade são presentes (COLE; GRIFFITHS, 2007). *Casual Games*, por outro lado, apresentam interações mais casuais em que laços de amizade são mais fracos (RICCHETTI, 2012).

### 2.1.1 Definição de Mobile Social Games

Existem diversas definições para *Social Game*, entretanto, todas possuem características semelhantes. De acordo com os autores Bergenti, Caire, Gotta, Park e Lee, *Social Game* é todo o jogo que utiliza uma plataforma, ou serviços, de uma rede social (BERGENTI; CAIRE; GOTTA, 2013) (PARK; LEE, 2012). Fields define que *Social Game* é um jogo em que as interações entre usuários ajudam na adoção e manutenção de jogadores conectados através de uma rede social externa, facilitando o alcance desses objetivos (FIELDS, 2014). Já Matt Ricchetti faz uma definição mais social. Ele define *Social Game* como sendo uma comunidade que possui várias interações presentes em relações sociais, tais como: competição, colaboração, rebelião, inveja e compaixão (RICCHETTI, 2012).

Matt Ricchetti vai além, e também define três características para divisão de *Social Games*. Essas características são baseadas no tipo de interação social e sincronia do jogo (RICCHETTI, 2012). As três características relativas a *Social Games* são:

- Interação síncrona vs assíncrona entre usuários. As interações ocorrem simultaneamente em tempo real, ou em tempos diferentes, em um estilo de turnos?;

- Formação simétrica vs assimétrica de relações socias. Formar uma relação social requer entrada de dados de ambos os participantes, ou pode ser formada de maneira unilateral?;
- Laço forte ou fraco na formação de relações sociais. As relações sociais tendem a ser fortes e duradouras, ou tendem a ser fracas e transitórias?

As três definições apresentadas são complementares e utilizadas em conjunto durante esse trabalho (As características alvo desse trabalho são descritas no capítulo 5). Por sua vez, *Mobile Social Games* se refere a *Social Games* que são desenvolvidos para plataformas *mobile*, como Android, iOS e Windows Phone.

### 2.1.2 Perspectiva Histórica de Mobile Social Games

Os primeiros *Social Games*, assim como os demais jogos de computador, eram bastante simples. Suas interfaces gráficas eram simuladas por texto e a quantidade de opções disponíveis eram limitadas. As primeiras versões de *Social Games* são conhecidas como *Multi-User Dungeons* (MUDs)(DIETERLE; CLARKE, 1997). Os MUDs possuiam seu estilo de jogo baseado em turnos, mas também permitiam que usuários interagissem de maneira síncrona ou assíncrona, em ambientes com diversas temáticas, desde guerra no espaço, até o velho oeste americano. Os MUDs eram executados em redes sociais conhecidas como *Bulletin Board Systems* (BBS), redes sociais baseadas em sistemas de texto que permitiam a interação entre usuários através de uma rede de telefone (FIELDS, 2014). *Trade Wars*, Figura 1, é um exemplo de MUD executado em BBS. O *Trade Wars* foi lançado em 1984 e possuia a temática de guerra no espaço. Nesse jogo, os usuários podiam competir e formar alianças para conquistar recursos.


Figura 1: MUD *Trade Wars* (FIELDS, 2014)

A redução no custo de produção de dispositivos com maior capacidade de processamento, e conexões de internet mais velozes e estáveis, permitiu que interfaces gráficas em

nível 2D e 3D, e o acesso aos *Social Games* fossem expandidos em diversas plataformas, como *PC* e *Mobile*.

A evolução dos MUDs conduziu ao desenvolvimento dos *Massively- Multiplayer Online Role-Playing Games* (MMORPGs). Os MMORPGs possuem características que os definem como *Social Games*. Nesse estilo de jogo, é possível a troca de mensagens entre usuários através de uma rede interna, e muitas vezes externa, *chat* em tempo real, formação de grupos para cooperação e vários outros estilos de organizações sociais (FIELDS, 2014).

Um dos MMORPG de maior sucesso para a plataforma PC é o *World of Warcraft*, Figura 2. O *World of Warcraft* foi desenvolvido pela *Blizzard Entertainment* em 2004, e possui, até o momento da escrita desse trabalho, o título de MMORPG mais rentável da história (THURAU; BAUCKHAGE, 2010) (ALTAY, 2015). *World of Warcraft* é baseado em um mundo medieval de fantasias, no qual os jogadores controlam personagens que podem evoluir de nível, desenvolver habilidades, participar de guerras, entre outras atividades. As atividades sociais do *World of Warcraft* são constantes no jogo. Jogadores interagem em ações de dança, festas, grupos de *chat* e formação de grupos para tarefas que não são possíveis de maneira individual, como conquista de itens especiais (THURAU; BAUCKHAGE, 2010) (NARDI; HARRIS, 2006).


Figura 2: MMORPG *World of Warcraft* (WINKIE, 2015)

Os MMORPG começaram na plataforma *mobile* em 2003 com o lançamento do TibiaME, versão *mobile* do MMORPG Tibia para PC (CIPSOFT, 2015a). O TibiaME é um MMORPG no estilo de fantasia medival em que os jogadores podem evoluir o nível dos seus personagens, participar de guerras e grupos sociais. As versões iniciais do TibiaME eram destinadas aos primeiros dispositivos móveis com suporte para J2ME (*Java 2 Platform, Micro Edition*) (CIPSOFT, 2015b). As primeiras versões possuiam gráficos em 2D, Figura 3, devido as restrições de hardware presentes nos primeiros *smartphones*, e necessitavam de acesso constante a internet, sendo essa realizada inicialmente através das primeiras redes 2G.

A evolução do hardware em *smartphones*, novas plataformas, redes móveis mais estáveis e velozes, e paralelamente a criação de novas redes sociais, como o *Facebook*, possibilitou o desenvolvimento e melhoria de vários novos tipos de *Mobile Social Games*, além dos MMORPGs (FIELDS, 2014). Alguns desses estilos de *Mobile Social Games* são: *Turn-Based Building Games*, *Simulation Games*, *Virtual Worlds*, *Non-Persistent Action and RTS Games* e *Online Trading Card Games* (FIELDS, 2014). Atualmente, vários *Mobile Social Games* estão disponíveis em lojas de aplicativos de todas as grandes plataformas, entre elas: Google Play, Apple Store e Windows Phone Store.


Figura 3: TibiaME. Primeiro MMORPG para plataforma *Mobile* (CIPSOFT, 2015b)

### 2.1.3 Problemas Ligados a Mobile Social Games

Os *smartphones* são dispositivos que estão presentes no cotidiano das pessoas. Os mesmos são produzidos por diferentes empresas, possuem sistemas operacionais distintos, tamanhos de telas variados, dentre outras especificações de hardware e software. Essa essência portátil, fragmentação de sistemas operacionais e hardware, rede móvel precária e capacidade de processamento menor que computadores pessoais, são raízes de vários problemas encontrados em *Mobile Social Games*.

De acordo com o *International Data Corporation*, o *marketshare* de sistemas operacionais para *smartphones* é amplamente ocupado pelo Android (CORPORATION, 2015). No segundo quadrimestre de 2015, o sistema operacional Android possuia 82.8% do mercado de *smartphones*, enquanto que o segundo maior sistema operacional, Apple iOS, possuia apenas 13.9% do mercado (CORPORATION, 2015), conforme Figura 4. Esses dados poderiam indicar uma maior facilidade para o desenvolvimento de aplicativos para *smartphones*, dado que a maior parte dos dispositivos possui o sistema operacional Android. No entanto, essa análise é incorreta. Mesmo possuindo um *marketshare* muito inferior ao Android, o iOS ainda gera uma receita maior que o seu concorrente (ANNIE, 2015). Essa característica faz com que desenvolvedores foquem em ambas as plataformas,

visto a oportunidade de geração de receita. Outro problema relacionado ao Android é a sua própria fragmentação, vide Figura 5. Dados da própria Google demonstram a existência de várias versões do Android, tamanhos de telas e suporte a diferentes versões de *engines* de vídeo (ANDROID, 2015b). Todas essas diferenças aumentam os custos e dificultam a criação de *Mobile Social Games* que sejam compatíveis com a maior parte do mercado.

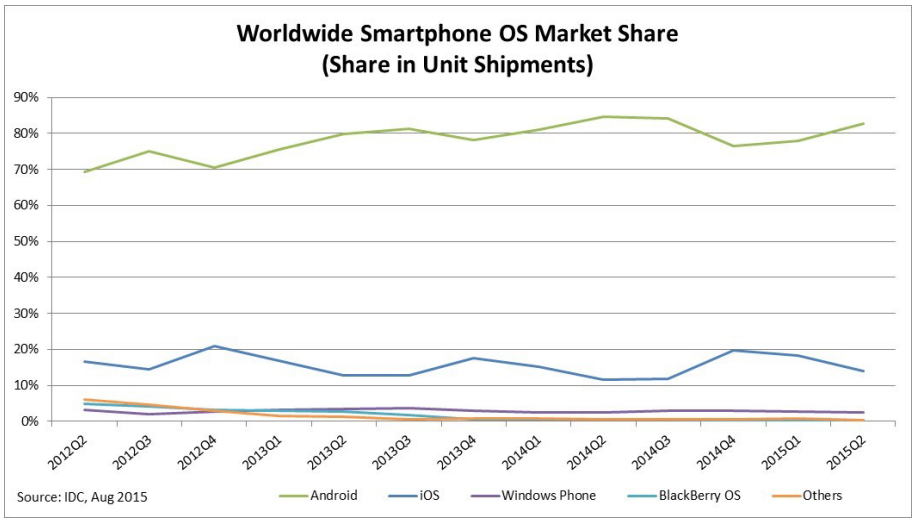


Figura 4: Quota de mercado de *Smartphones* entre 2012 e 2015 (CORPORATION, 2015).

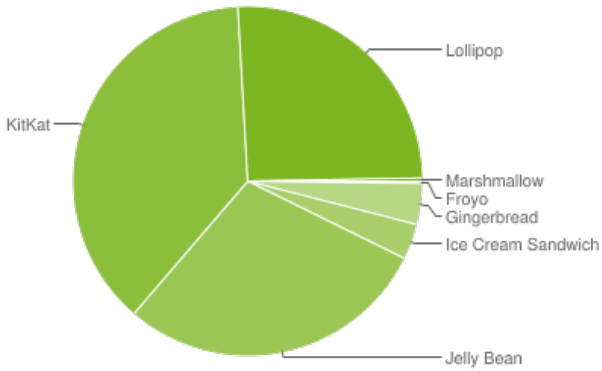


Figura 5: Fragmentação das versões do Android em 2015 (ANDROID, 2015b).

### 2.1.4 Mercado de Mobile Social Games

*Mobile Games* tornaram-se uma das mais importantes plataformas para jogadores e desenvolvedores de jogos. O mercado de *Mobile Games* demonstra crescimento contínuo e números expressivos. Na Figura 6, é possível analisar que em 2014, o mercado de *Mobile Games* apresentou um aumento de receita de 33% e 55%, em *smartphones* e *tablets*, respectivamente. Em 2017, a perspectiva é que *Mobile Games* serão responsáveis por 38% do mercado global de jogos e receita prevista de $40.4 bilhões de dólares (CONFEDERATION; NEWZOO; TALKINGDATA, 2015).

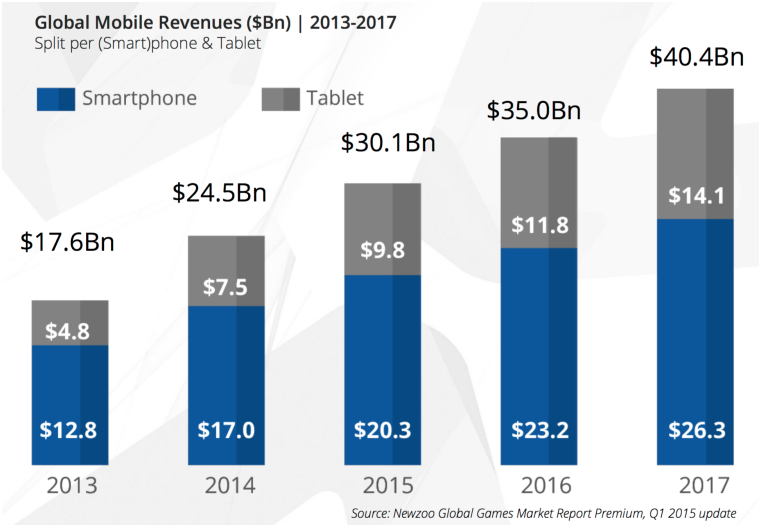


Figura 6: Perspectiva de receita para o mercado *Mobile* (CONFEDERATION; NEWZOO; TALKINGDATA, 2015)

O mercado de *Mobile Games* está em constante crescimento e possui características que estão sendo analisadas para o mapeamento dos motivos para o seu sucesso. Fields (FIELDS, 2014) revela algumas características interessantes do mercado de *Mobile Gaming* nos Estados Unidos da América. Essas características são referentes a distribuição demográfica no ano de 2013 (FIELDS, 2014):

- As mulheres representam 53% dos usuários do mercado de *Mobile Games*;
- 50% do jogadores possuem entre 25 e 30 anos;
- 57% dos jogadores jogam diariamente, e 54% jogam por mais de uma hora por dia;
- 30% dos jogadores jogam na cama, e 16% no ônibus ou metrô.

A receita gerada pelo mercado de *Mobile Games* possui números atraentes para os desenvolvedores, no entanto, a monetização de jogos não é uma tarefa fácil. Atrair usuários para o seu jogo em um mercado repleto de opções é uma tarefa difícil, especialmente quando se trata de uma empresa que está iniciando, ou de porte pequeno. Dado essas dificuldades, várias estratégias de monetização de *Mobile Games* são utilizadas (FIELDS, 2014):

- *Premium download*;
- *Subscriptions*;
- *Freemium*.

O modelo *Premium download* caracteriza-se pela venda completa do jogo para o usuário. O usuário compra o jogo em uma *app store*, como *Google Play* e *Amazon Appstore*, e obtém o jogo em sua completude, sem necessidade a de dispêndios futuros. Esse modelo é normalmente utilizado por estúdios e franquias que já possuem uma grande base de usuários, ou verba suficiente para atrair usuários através de ações publicitárias.

O modelo *Subscriptions* é baseado em assinaturas. O usuário recebe de maneira gratuita os primeiros meses do jogo, sendo necessária uma assinatura posterior para a continuação do jogo. Esse modelo é normalmente utilizado em jogos que necessitam de um grande processamento em *back-end*, como os MMORPG. Esse modelo está em declínio devido ao crescimento de jogos de alta qualidade que utilizam o próximo modelo, o *Freemium*.

O modelo *Freemium* pode ser subdividido em várias categorias, todas com a mesma base: o jogo é gratuito. A monetização com esse modelo é feita através da venda de itens, serviços e propagandas dentro do jogo. Os usuários possuem a possibilidade de comprar itens para aumentar a velocidade de evolução no jogo, liberar áreas indisponíveis e retirar propagandas indesejadas através de pagamento.

Os modelos apresentados podem ser utilizados de maneira singular ou híbrida, o que é necessário, é a análise de qual, ou quais, modelos se adequam ao estilo de jogo implementado, tipo de público alvo, verba disponível para propaganda do jogo, entre outros fatores.

Através dos dados apresentados, fica evidente que o mercado de *Mobile Games* está em um momento de crescimento e geração de oportunidades para desenvolvedores independentes e empresas que já estão solidificadas no mercado de jogos. Novas técnicas, princípios, metodologias e paradigmas podem ser a solução para a melhoria do processo de desenvolvimento de *Mobile Social Games*. O próxima seção aborda tópicos ligados à Engenharia de Software, como Sistemas MultiAgentes, que podem ser utilizados para abordar alguns dos problemas relatados.

## 2.2 Engenharia de Software

Dado os objetivos desse trabalho, faz-se necessário o embasamento teórico quanto aos aspectos associados à Engenharia de Software. Dentre elas está a utilização do paradigma MultiAgentes, testes unitários e automatizados para Sistemas MultiAgentes e para a plataforma *mobile* escolhida e técnicas para análise de qualidade de código fonte.

### 2.2.1 Sistemas MultiAgentes

A visão baseada em agentes oferece suporte de ferramentas, técnicas, e metáforas visando melhorar a maneira como as pessoas conceitualizão e implementam diferentes produtos de software. Agentes são usados em uma variedade crescente de aplicações — desde sistemas pequenos, como filtros personalizados para emails, até grandes, complexos e críticos sistemas de controle aéreo (ITALIA; PARMA, 2014). À primeira vista, pode parecer que tais sistemas possuem pouco em comum. No entanto, esse não é o caso: em ambos, a abstração chave utilizada é a de agentes (JENNINGS; SYCARA; WOOLDRIDGE, 1998).

De acordo com Wooldridge, Sistemas MultiAgentes (SMA) são sistemas compostos por múltiplos elementos computacionais que interagem entre si, conhecidos como Agentes. Agentes são sistemas computacionais com duas importantes capacidades. Eles possuem, pelo menos em certo nível, capacidade de agirem de maneira autônoma, de decidirem por si mesmos o que eles precisam fazer para satisfazer os seus objetivos. Eles são capazes de interagir com outros agentes, não simplesmente compartilhando dados, mas se envolvendo em atividades análogas às atividades sociais em que todos nós nos envolvemos diariamente: cooperação, coordenação, negociação, e similares (WOOLDRIDGE, 2009). As características presentes em um Sistema MultiAgente são (JENNINGS; SYCARA; WOOLDRIDGE, 1998):

- Cada agente possui uma informação incompleta, ou capacidade para resolver um problema, ou seja, cada agente tem uma visão limitada;
- Não existe um sistema global de controle;
- Os dados são descentralizados e;
- A computação é assíncrona.

Alguns dos motivos pelo interesse em SMA são: habilidade de prover robustez e eficiência; habilidade para permitir inter-operação entre sistemas legados; e habilidade para resolver problemas em que *data*, *expertise* ou controle são distribuídos (JENNINGS; SYCARA; WOOLDRIDGE, 1998).

As características e habilidades listadas anteriormente podem ser atingidas através de arquiteturas distintas de Sistemas MultiAgentes. Essas arquiteturas são descritas e abordadas no próximo tópico.

#### 2.2.1.1 Tipos de Arquiteturas Orientadas a Agentes

As arquiteturas de agentes são mecanismos fundamentais, subjacentes aos componentes autônomos, que permitem comportamento efetivo e dinâmico em um mundo real, e

em ambientes abertos. As arquiteturas de agentes podem ser divididas em quatro grupos principais: baseados em lógica, reativos, crença-desejo-intenção(*belief, desire, intention*), e em camadas (ou híbrida)(BELLIFEMINE; CAIRE; GREENWOOD, 2007).

A arquitetura **baseada em lógica** possui sua fundação em técnicas conhecidas de sistemas tradicionais em que o ambiente é representado de forma simbólica e manipulado usando mecanismos de raciocínio(BELLIFEMINE; CAIRE; GREENWOOD, 2007).

Arquiteturas **reativas** implementam a tomada de decisão como um mapeamento direto da situação para ação, baseando-se em estímulos desencadeados por sensores de dados. Ao contrário da arquitetura **baseada em lógica**, a arquitetura reativa não possue qualquer modelo simbólico central, e portanto, não utiliza qualquer raciocínio lógico simbólico complexo(BELLIFEMINE; CAIRE; GREENWOOD, 2007).

A arquitetura **crença-desejo-intenção**, ou BDI (*belief, desire, intention*), é baseada em três estruturas: crença (*belief*), que é o conhecimento que o agente possui do ambiente; desejo (*desire*), que representa os objetivos que ele deve alcançar; e intenção, que representa as tarefas a serem realizadas pelo agente, ou seja, planos estratégicos visando alcançar os objetivos desejados. (*intention*)(BELLIFEMINE; CAIRE; GREENWOOD, 2007).

Por fim, a arquitetura em **camadas**, ou híbrida, permite tanto o comportamento reativo quanto o deliberativo. Para permitir essa flexibilidade, subsistemas são dispostos como camadas (horizontais e verticais) de maneira hierárquica para acomodar ambos os tipos de agentes (BELLIFEMINE; CAIRE; GREENWOOD, 2007).

### 2.2.2 Testes de Software

Um dos objetivos desse trabalho é aprofundar os conhecimentos em Engenharia de Software, orientando-se por padrões arquiteturais, testes e técnicas para análise de qualidade de código fonte. Para isso é necessário a apresentação de referências para o desenvolvimento da base teórica. Essas referências são apresentadas através das subseções: subseção 2.2.2, apresenta as bases teóricas para testes de software, com foco em testes unitários e de integração; subseção 2.2.4 apresenta as bases teóricas de qualidade de software, com foco em métricas de qualidade de código fonte.

Teste de Software é um processo, ou uma série de processos, destinado(s) a certificar-se que o código de computador faça o que foi projetado para fazer, e inversamente, não faça algo não planejado(MYERS; BADGETT; SANDLER, 2011).

Myers define os seguintes princípios para o teste de Software(MYERS; BADGETT; SANDLER, 2011):

- Testar é o processo de executar o programa com a intenção de encontrar erros;

- O teste é mais bem sucedido quando não realizado pelo próprio desenvolvedor;
- Um bom caso de teste é aquele que possui uma maior probabilidade de encontrar erros não conhecidos;
- Um caso de teste bem sucedido é aquele que detecta um erro desconhecido;
- Um teste bem sucedido inclui a definição cuidadosa dos dados de entrada e saída;
- Um teste bem sucedido inclui estudar cuidadosamente os resultados obtidos.

Teste está presente em todos os estágios do ciclo de desenvolvimento de Software, mas é feito de uma maneira diferente em cada nível. Lu Luo define os seguintes níveis de teste (LUO, 2001):

- **Teste Unitário:** Teste feito no mais baixo nível. Testa as estruturas unitárias do software, que é a menor parte testável;
- **Teste de Integração:** É realizado quando duas ou mais unidades são combinadas em uma estrutura maior;
- **Teste de Sistema:** Tende a afirmar a qualidade de todo o sistema. Esse teste é muitas vezes baseado em requisitos funcionais do sistema. Atributos de qualidade não funcionais também são verificados;
- **Teste de Aceitação:** É um teste formal destinado a validar a aceitação do sistema diante da avaliação do usuário, cliente ou entidade autorizada.

Existem duas categorias de técnicas para teste de software. Diferentes técnicas revelam diferentes aspectos de qualidade do software. As categorias são funcional e estrutural. Os testes funcionais recebem dados de entrada, e sua saída é avaliada para análise da conformidade com o esperado. O teste estrutural também recebe dados de entrada e tem sua saída avaliada, porém a análise vai além, verificando a estrutura interna do sistema ou componente(LUO, 2001).

O teste funcional, ou caixa-preta, visualiza o sistema como uma caixa-preta. O objetivo não é verificar o comportamento interno ou estrutural, mas sim em quais circunstâncias o programa não se comporta como esperado(MYERS; BADGETT; SANDLER, 2011).

O teste estrutural, ou caixa-branca, permite examinar a estrutura interna do programa. Essa estratégia deriva testes através da lógica do programa. O objetivo é testar todos os caminhos possíveis de execução(MYERS; BADGETT; SANDLER, 2011).

### 2.2.3 Testes para Dispositivos Android

A plataforma Android inclui um *framework* integrado que facilita o teste de todos os aspectos dos aplicativos. O SDK (*Software Development Kit*) também fornece ferramentas para a configuração e execução dos testes.

#### 2.2.3.1 O *Framework* de Testes

O *framework* de testes fornece as seguintes características: as *suites* de teste são baseadas no JUnit (JUNIT, 2015); as *suites* de testes são contidas em pacotes de teste que são similares aos pacotes da aplicação; o *framework* fornece extensões do JUnit para testes específicos do Android; o SDK pode ser utilizado através da IDE Eclipse ou por comandos de terminal, e o SDK fornece integração com o *monkeyrunner*, uma API para testar dispositivos Android através de código Python.

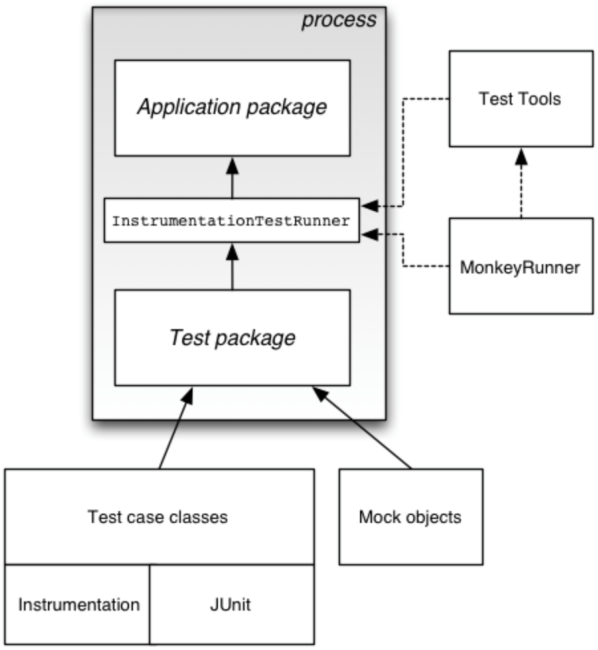


Figura 7: Estrutura do framework de testes (ANDROID, 2015a)

A Figura 7 resume a estrutura do *framework* de testes. É possível ver que a estrutura de testes do Android é baseada no JUnit. Portanto, os testes são organizados em classes, pacotes e projeto. O *framework* também disponibiliza um conjunto de métodos, *InstrumentationTestRunner*, que permite a execução de componentes do Android sem as restrições dos ciclos de vida (ANDROID, 2015a).

O *framework* disponibiliza um conjunto de ferramentas que podem ser utilizadas para diferentes tipos de teste. Como teste de interface gráfica (*Espresso* e *UI Automator*) e testes compatíveis com JUnit3 e JUnit4 (*AndroidJUnitRunner*). Outra ferramenta dis-

ponível é o *MonkeyRunner*, uma ferramenta que fornece a opção para criação de testes fora do escopo de código do aplicativo Android.

### 2.2.4 Qualidade de Software

No contexto de software, a qualidade é vista como um aglomerado de características que são alcançadas para que o resultado obtido no desenvolvimento atenda as necessidades do usuário, sendo elas explícitas ou não (ROCHA; MALDONADO; WEBER, 2001). As características de qualidade de software podem ser divididas em funcionais, e não funcionais. Características funcionais podem ser medidas através de métricas que possuem critérios objetivos, tornando possível a avaliação da qualidade (MEIRELLES, 2013).

#### 2.2.4.1 Métricas de Qualidade de Código Fonte

Métricas de software é um termo que envolve várias atividades para a definição de escalas e métodos que possuem como objetivo a identificação de parâmetros que afetam o desenvolvimento do software (FENTON; BIEMAN, 2015).

As métricas de software podem ser agrupadas em objetivas e subjetivas. As métricas objetivas possuem regras bem definidas, e possibilitam comparações posteriores. Enquanto que as métricas subjetivas podem alterar de acordo com o ator responsável pela sua coleta (MEIRELLES, 2013).

Métricas de código fonte objetivas possuem características que permitem o mapeamento dos seus resultados em intervalos para serem interpretados em uma fase de análise (MEIRELLES, 2013). Alguns exemplos são:

- Número de atributos;
- Média de complexidade ciclomática por método;
- Respostas para uma classe;
- Número de filhos;
- Fator de acoplamento, e
- Complexidade estrutural.

## 2.3 Considerações Finais do Capítulo

As várias particularidades ligadas a *Mobile Social Games* evidenciam que este ambiente virtual possui diversas interações presentes no cotidiano das pessoas. Tal fato

torna esse formato de jogo atrativo. A quantidade de jogadores com interesse nesse estilo é crescente, fazendo com que o mercado de *Mobile Social Games* esteja em crescimento constante.

A aplicação do paradigma MultiAgentes se torna oportunua nesse contexto, dado que expõe possíveis melhorias de Engenharia de Software. Dentre essas melhorias, destacam-se: autonomia, adaptabilidade e flexibilidade. Essas melhorias podem potencializar os ganhos em produtividade e redução de custos de desenvolvimento, visto que as abstrações são mais próximas às necessidades de Mobile Social Games.

Por fim, esse capítulo listou boas práticas de Engenharia de Software, com foco em testes de software e qualidade de software. Essas boas práticas tornarão possível melhorias no contexto de desenvolvimento e Engenharia de Software.

# 3 Suporte Tecnológico

Este capítulo apresenta as ferramentas utilizadas no suporte do desenvolvimento desse TCC. São apresentados os componentes necessários para a aplicação de Sistemas MultiAgentes, seção 3.1, e outras necessárias para o suporte e desenvolvimento, seção 3.2.

## 3.1 Sistemas MultiAgentes

Sistemas MultiAgentes não são largamente utilizados no desenvolvimento de *Mobile Social Games*, somente a plataforma AMUSE(*Agent-based Multi-User Social Environment*) apresenta uma aplicação desse paradigma. O AMUSE possui como base os *frameworks* JADE(*Java Agent DEvelopment Framework*) e WADE(*Workflows and Agents Development Environment*), que serão abordados nesse capítulo.

### 3.1.1 Plataforma JADE

O *framework* JADE(*Java Agent DEvelopment Framework*) é um *framework* de software implementado em Java. Ele simplifica a implementação de Sistemas MultiAgentes através de uma camada *middle-ware* que está em conformidade com as especificações da FIPA(*Foundation for Intelligent Physical Agents*), e através de um conjunto de ferramentas gráficas que suportam a fase de depuração e implantação (ITALIA; PARMA, 2014).

JADE é um software livre, distribuído sob a licença LGPL(*Lesser General Public License*), juntamente com todas as suas extensões e *plugins*. O repositório do projeto e todas as suas extensões são compartilhadas com a comunidade através do repositório *subversion* disponível em: <https://jade.tilab.com/svn/jade/trunk>.

A Figura 8 resume a estrutura da plataforma Jade. Uma aplicação baseada no JADE é composta por vários componentes. Os principais componentes são a plataforma, os contêineres e os agentes. Os agentes executam tarefas e trocam informações através de mensagens (BELLIFEMINE; CAIRE; GREENWOOD, 2007). Os agentes são posicionados no topo de uma plataforma que provê serviços básicos como troca de mensagens. Uma plataforma é composta por um, ou vários contêineres. Cada contêiner pode ser executado em *hosts* distintos. Cada contêiner pode conter zero ou vários agentes. Existe um tipo especial de contêiner, *MainContainer*. Esse contêiner é necessariamente o primeiro a ser inicializado na plataforma, e possui dois tipos especiais de agentes: Páginas Brancas (*Agent Management System* - AMS) e Páginas Amarelas (*Directory Facilitator* - DF). O AMS é o único agente capaz de executar operações em nível de plataforma, e o DF

é responsável por controlar os serviços disponibilizados pelos outros agentes (ITALIA; PARMA, 2014).

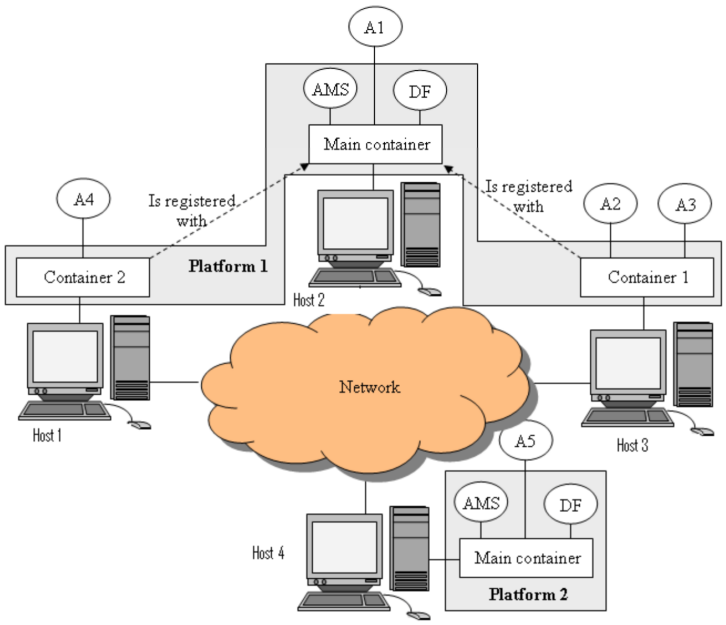


Figura 8: Arquitetura do *framework* JADE (ITALIA; PARMA, 2015a)

#### 3.1.1.1 JADE para Android

A plataforma JADE possui um conjunto de extensões que atendem a diversas necessidades. Uma dessas extensões é o JADE para Android, uma extensão necessária para que dispositivos Android possam implantar agentes JADE. Em detalhes, o JADE é envolvido em um serviço específico Android. Um serviço Android é um componente de aplicação que pode executar operações de longa duração e que não fornecem uma interface para o usuário. Outros componentes da aplicação podem iniciar um serviço e ele continuará a execução em *background* mesmo que o usuário mude de aplicação (BERGENTI; CAIRE; GOTTA, 2014).

O JADE para Android fornece uma interface que possibilita que aplicações iniciem agentes, disparem comportamentos, e em geral, compartilhem dados entre agentes. Por conseguinte, é possível descobrir pares remotos, realizar conversas complexas, explorar ontologias JADE, executar atividades em *background* de acordo com o comportamento, e tirar proveito de todas as funcionalidades do JADE (BERGENTI; CAIRE; GOTTA, 2014).

### 3.1.2 Plataforma WADE

WADE (*Workflows and Agents Development Environment*) é a principal evolução do JADE, e adiciona a habilidade de definir lógicas de sistema de acordo com a metáfora de

fluxo de trabalho, além de prover mecanismos que ajudam a gerir complexidades inerentes a Sistemas MultiAgentes distribuídos, tanto em administração, quanto em tolerância a falhas (CAIRE; QUARANTOTTO; SACCHI, 2009). Até o presente momento, o AMUSE utiliza o WADE apenas pelas suas características de flexibilidade e escalabilidade em *deployment* (BERGENTI; CAIRE; GOTTA, 2015).

O principal componente da plataforma WADE é *WorkflowEngineAgent*, uma classe que estende o agente básico do JADE incorporando um pequeno e leve motor de fluxo de trabalho. Além dos comportamentos normais do JADE, um *WorkflowEngineAgent* é capaz de executar fluxos de trabalho representados de acordo com as especificações do WADE. Um fluxo de trabalho é uma definição formal do processo em termos de atividades a serem executadas, relações entre elas, critérios que especificam sua ativação e términio, e informações adicionais. As informações adicionais podem ser participantes, ferramentas de software a serem invocadas, entradas requeridas, saídas esperadas e informação manipulada durante a execução. Essa abordagem torna possível a combinação da expressividade de metáforas de fluxo de trabalho com o poder de uma linguagem de programação, como o Java; além de possibilitar o uso de fluxos de trabalho para definição de lógicas internas de sistema (ITALIA; PARMA, 2015b).

A Figura 9 exemplifica a plataforma WADE. Os principais componentes da plataforma são: JADE, responsável pelos agentes e comportamentos, comunicação, e distribuição da execução; WADE, responsável pela construção do fluxo de trabalho, administração e controle de falhas; *Application*, a aplicação desenvolvida com suas caracaterísticas específicas; WOLF (*WOrkflow LiFe cycle management environment*), ambiente de desenvolvimento gráfico para a plataforma WADE, caso o usuário utilize a IDE (*Integrated Development Environment*) Eclipse.

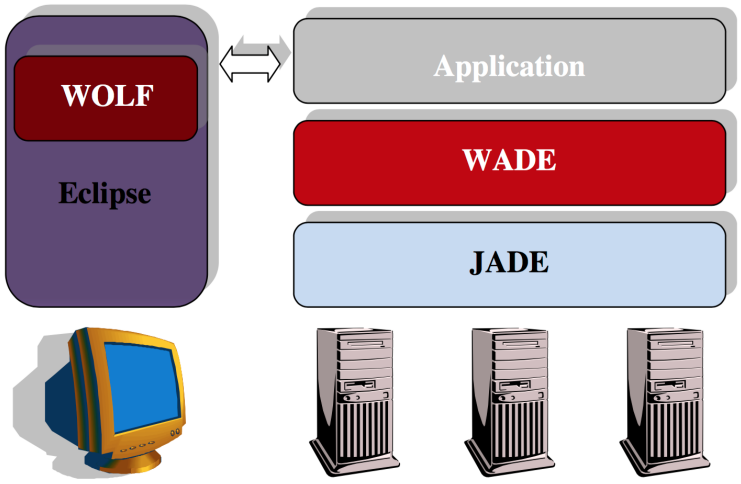


Figura 9: Plataforma WADE (CAIRE, 2013)

### 3.1.3 Plataforma AMUSE

O AMUSE (*Agent-based Multi-User Social Environment*) é uma plataforma de código aberto, baseada no JADE e WADE, que facilita o desenvolvimento de aplicações sociais distribuídas que envolvem usuários em atividades de cooperação e competição. O foco principal do AMUSE é no suporte de jogos *multi-player online* para o sistema Android. O AMUSE utiliza a base do WADE e JADE para controlar todas as comunicações e questões de gestão de componentes. A plataforma aborda aspectos relacionados à organização, coordenação e sincronização de partidas entre jogadores. Ela não oferece suporte para o desenvolvimento de interfaces gráficas (SPA, 2015).

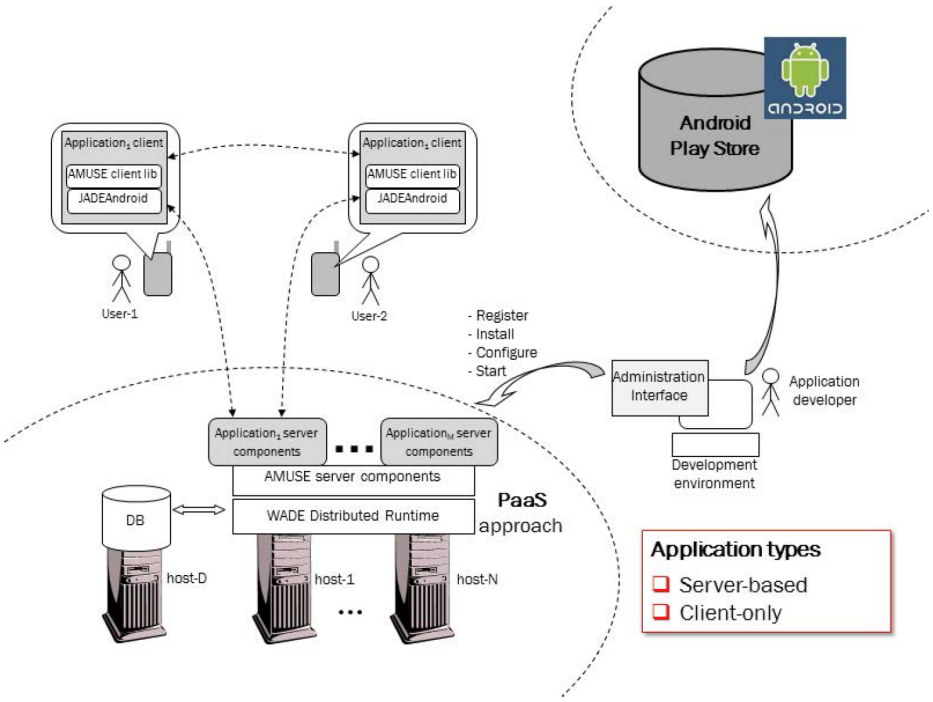


Figura 10: Visão arquitetural alto nível do AMUSE
(BERGENTI; CAIRE; GOTTA, 2015)

A Figura 10 mostra uma visão alto nível da arquitetura do AMUSE. Todas as aplicações baseadas no AMUSE englobam um cliente que fornece a interface com o usuário, e qualquer outra característica específica de lógica do lado do cliente. A aplicação do lado do cliente faz uso da biblioteca do AMUSE para interagir com outros usuários e com o servidor. Aplicações que requerem execução de lógicas no lado do servidor contam com um ambiente PaaS (*Platform as a Service*) para essas execuções. Isso significa que o AMUSE fornece componentes que tornam a plataforma prestadora de serviço. A aplicação não tem conhecimento dos detalhes referentes a hardware, sistemas operacionais e outras características do *host*. Por outro lado, eles são implementados em um ambiente de nuvem, e o AMUSE garante que a aplicação tenha os recursos suficientes para sua execução (CAIRE, 2015).

O AMUSE fornece um conjunto de funcionalidades que não estão estritamente ligadas ao domínio de jogos. Essas funcionalidades são: *Application management*, *User Management*, *Clock synchronization*, *Text message exchange*, *Peer-to-peer pipe management* e *Centralized match coordination*. Essas funcionalidades são implementadas através de recursos provenientes do JADE, WADE e agentes da própria plataforma. Esse conjunto de agentes estão do lado do servidor, e do lado do usuário. As interações entre esses diferentes tipos de agentes fornecem a funcionalidade da plataforma. Os diferentes tipos de agentes disponibilizados pela plataforma são (BERGENTI; CAIRE; GOTTA, 2015):

- MMA*(Match Manager Agent):* Este é o único agente no lado do usuário. Ele tem como responsabilidade fazer a interface entre o usuário e os agentes do lado do servidor, e executar tarefas que não necessitam de interação com os agentes do servidor;
- AMA*(Application Manager Agent):* Agente responsável por realizar o controle dos jogos disponibilizados pela plataforma e seus ciclos de vida;
- GRA*(Games Room Agent):* Agente responsável pelo controle dos dados compartilhados em jogos com interações síncronas;
- UMA*(User Manager Agent):* Este agente realiza o controle dos perfis e relações dos usuários com os demais jogadores;
- MTA*(Match Tracer Agent):* O agente MTA tem como objetivo suprir as necessidades dos jogos que necessitam de opções de reiniciar e persistir os seus estados.

## 3.2 Ferramentas de Suporte e Desenvolvimento

Um conjunto de ferramentas foi selecionado para auxiliar o desenvolvimento desse trabalho. Esse conjunto de ferramentas é necessário para a implementação do código, teste, controle de versão e outras atividades inerentes ao escopo desse projeto.

### 3.2.1 Teste de Software

Para a realização de testes unitários, o *framework* selecionado foi o JUnit. JUnit é um *framework* de código aberto para escrita de testes repetíveis na linguagem Java. Ele é uma instância da arquitetura xUnit para testes unitários (JUNIT, 2015).

### 3.2.2 Controle de Versão

Para o controle de versão do desenvolvimento do projeto, a ferramenta selecionada foi GIT. O Git é um sistema distribuído de controle de versão, disponibilizado de maneira

livre e código aberto. Ele é projetado para lidar com projetos de diversos tamanhos com velocidade e eficiência (GIT, 2015). O repositório do projeto está hospedado no GitHub, que é um repositório web baseado no Git. Ele oferece todas as funcionalidades de controle de versão presentes no Git, como também adiciona suas próprias funcionalidades (GITHUB, 2015).

### 3.2.3 Ferramentas de Desenvolvimento

As ferramentas de desenvolvimento selecionadas para o suporte ao projeto são: Atom, LaTex, Eclipse, Android Studio e MacOS.

O Atom é um editor de texto de código aberto, customizável, com suporte para todos os tipos de códigos abordados nesse trabalho e disponível para os sistemas operacionais Mac OS, Windows e Linux. Ele possui de forma nativa suporte para controle de versão através do Git (ATOM, 2015). A versão utilizada é a 1.16.0 x64.

LaTeX é um sistema de preparação de documentos para a composição tipográfica de alta qualidade. Ele inclui funcionalidades projetadas para a produção técnica e científica de documentos (LATEX, 2015). A versão utilizada é a 2.7.5.

O Eclipse é uma IDE (*Integrated Development Environment*) *open source* para desenvolvimento Java, com suporte para outras linguagens através de plug-ins (ECLIPSE, 2015). A versão utilizada é a Neon.3 (4.6.3).

O Android Studio é uma IDE (*Integrated Development Environment*) para desenvolvimento de aplicações Android, desenvolvida e distribuída pelo Google (GOOGLE, 2017). A versão utilizada é a 2.3.2 para MacOS.

Por fim, o sistema operacional escolhido foi o Mac OS. Ele é desenvolvido pela Apple e baseado no Unix. A versão utilizada no desenvolvimento desse trabalho é a Mac OS Sierra 10.12.5 (OSX, 2016).

## 3.3 Considerações Finais do Capítulo

O objetivo desse capítulo foi descrever as ferramentas e tecnologias utilizadas no suporte do desenvolvimento desse trabalho. A seção 3.1, Sistemas MultiAgentes, descreveu as tecnologias necessárias para a aplicação do paradigma MultiAgentes no contexto de *Mobile Social Games*. A seção 3.2, Ferramentas de Suporte e Desenvolvimento, descreveu as ferramentas necessárias para o suporte e desenvolvimento da proposta, aplicação de boas práticas de Engenharia de Software, e escrita desse projeto.

# 4 Metodologia

O interesse desse capítulo é responder como os objetivos desse trabalho serão alcançados e como a solução do problema de pesquisa será abordado. As seguintes seções são apresentadas: seção 4.1, apresenta o tipo de metodologia de pesquisa utilizada no desenvolvimento desse trabalho; seção 4.2, especifíca o planejamento, atividades, e fluxo de execução; seção 4.3, apresenta o cronograma; seção 4.4, detalha a prova de conceito.

## 4.1 Classificação da Pesquisa

Para a classificação de um tipo de pesquisa, é necessário a definição clara de critérios baseados nos objetivos e procedimentos a serem seguidos (GIL, 2002). Os tipos de pesquisa baseadas nos objetivos são usualmente classificadas em três grupos: exploratória, descritiva e explicativa (GIL, 2002).

A pesquisa exploratória é definida por Theodorson e Theodorson como sendo um estudo preliminar com o objetivo principal de se familiarizar com o tópico abordado, de modo que o estudo posterior possa ser realizado com maior compreensão e precisão. Theodorson e Theodorson também esclarecem que a pesquisa exploratória permite que o pesquisador defina o seu problema de pesquisa e formule hipóteses mais precisas, tornando possível a escolha de técnicas mais adequadas e gerando alertas para quais as potenciais dificuldades e sensibilidades da área (THEODORSON; THEODORSON, 1970).

De acordo com Fonseca, a pesquisa descritiva é utilizada quando o pesquisador tem conhecimento prévio do assunto e possui interesse em descrever um fenômeno. A partir desse conhecimento prévio, o pesquisador pode formular hipóteses, podendo confirmá-las ou não (FONSECA, 2002).

Já a pesquisa explicativa tem como objetivo central identificar fatores que determinam ou contribuem para a ocorrência de fenômenos. Esse tipo de pesquisa explica o porquê a razão, o motivo de acontecimentos (GIL, 2002).

Dado os objetivos de estudo e contexto desse trabalho, a metodologia de pesquisa escolhida foi a pesquisa exploratória.

## 4.2 Modelagem do Fluxo de Atividades

O Diagrama a seguir 11, desenvolvido utilizando o BPMN2 *Modeler Plugin* para Eclipse (ECLIPSE, 2015), ilustra a condução das atividades requeridas para o desenvol-

vimento do TCC 1, e as planejadas para o TCC 2. As descrições das atividades presentes no diagrama são apresentadas em seguida.

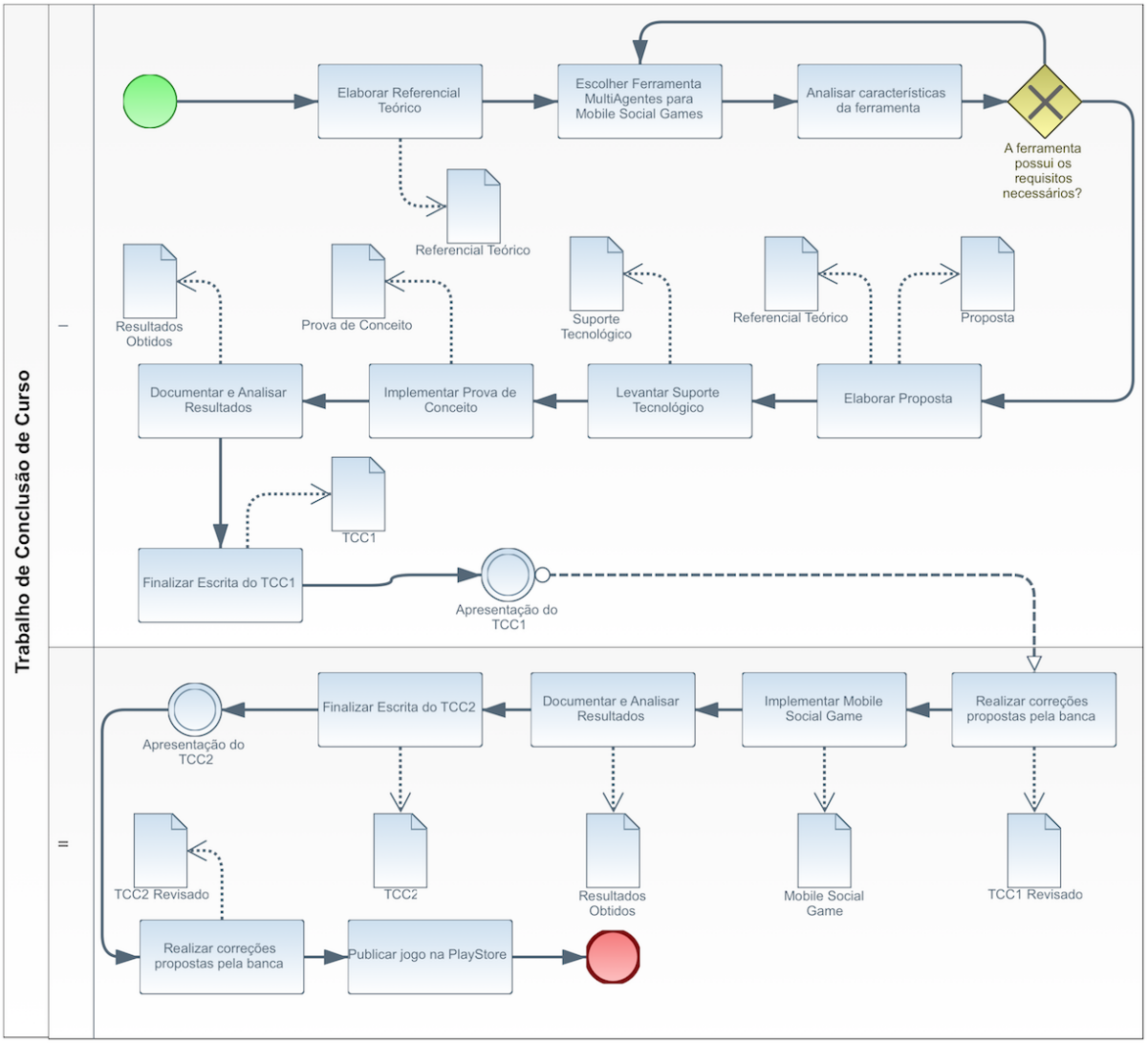


Figura 11: Fluxo de atividades do TCC

Atividades mapeadas para o TCC 1:

**Elaborar referencial teórico:** teve como objetivo construir o conhecimento base para a elaboração do trabalho. Através do mapeamento da literatura, publicações, e autores.

**Escolher ferramenta MultiAgentes para *Mobile Social Games*:** teve como objetivo a pesquisa, e escolha de uma ferramenta que aplicasse o paradigma MultiAgentes no contexto de *Mobile Social Games*. Essa pesquisa obteve como resultado uma única ferramenta, o AMUSE, descrito detalhadamente no capítulo de Suporte Tecnológico.

**Analisar características da ferramenta:** teve como objetivo analisar as carac-

terísticas e restrições da ferramenta escolhida, no caso, o AMUSE.

**Elaborar proposta:** teve como objetivo a definição de uma proposta que contemplasse os objetivos deste trabalho de conclusão de curso. Essa proposta foi composta de uma visão geral dos objetivos, e uma visão técnica arquitetural.

**Levantar suporte tecnológico:** teve como objetivo o levantamento das das ferramentas de suporte necessárias para a elaboração desse trabalho.

**Implementar prova de conceito:** teve como objetivo a implementação da prova de conceito para provar a viabilidade técnica desse trabalho. E que poderá ser utilizada em parte no desenvolvimento da proposta.

**Documentar e analisar resultados:** teve como objetivo a formalização dos resultados obtidos no desenvolvimento da prova de conceito.

**Finalizar escrita do TCC 1:** teve como objetivo o refinamento do trabalho, e adição de observações pertinentes para entrega a banca avaliadora.

Atividades mapeadas para o TCC 2:

**Realizar correções propostas pela banca:** possui como objetivo a correção dos pontos levantados pela banca avaliadora.

**Implementar *Mobile Social Game*:** possui como objetivo a implementação da proposta feita no TCC 1.

**Documentar e analisar resultados:** possui como objetivo a documentação e análise dos resultados obtidos com o desenvolvimento da proposta, e evolução do projeto.

**Finalizar escrita do TCC 2:** possui como objetivo o refinamento do trabalho, e adição de observações pertinentes para entrega a banca avaliadora.

**Publicar jogo na Google Play:** possui como objetivo a monetização da proposta desenvolvida através da publicação na plataforma Google Play.

## 4.3 Cronograma de Pesquisa

O cronograma seguido para o desenvolvimento desse trabalho pode ser observado na Tabela 1 e Tabela 2. A Tabela 2, referente ao cronograma do TCC 2, é passível de mudanças de acordo com o andamento do trabalho.

| Atividade | Janeiro | Fevereiro | Março | Abril | Maio | Junho |
|---|---|---|---|---|---|---|
| Levantamento Bibliográfico | X | X | X | | | |
| Definição do Escopo | | X | X | | | |
| Levantamento do Suporte Tecnológico | | X | X | | | |
| Implementar Prova de Conceito | | | X | X | X | |
| Análise de Resultados | | | X | X | X | |
| Escrita do TCC 1 | | | X | X | X | X |
| Apresentação do TCC 1 | | | | | | X |
| Realizar correções requisitadas pela banca | | | | | | X |

Tabela 1: Cronograma TCC 1

| Atividade | Junho | Julho | Agosto | Setembro |
|---|---|---|---|---|
| Implementar Mobile Social Game | X | X | X | |
| Documentar e Analisar Resultados | | | X | X |
| Escrita do TCC 2 | | | X | X |
| Apresentação do TCC 2 | | | | X |
| Realizar correções requisitadas pela banca | | | | X |

Tabela 2: Cronograma TCC 2

## 4.4 Prova de Conceito

Dado o contexto de estudo desse projeto, é necessário a integração de várias plataformas que utilizam tecnologias distintas. A avaliação dos possíveis riscos de integração dessas plataformas é pertinente no atual momento do projeto, pois ocorre em um estágio inicial de desenvolvimento.

Para provar a viabilidade técnica do desenvolvimento da proposta, foi elaborada uma prova de conceito, que possui como objetivo principal a integração das várias plataformas presentes na proposta desse projeto. As subseções 4.4.1 e 4.4.2 detalham a Descrição da Prova de Conceito e Configuração do Ambiente de Desenvolvimento, respectivamente.

### 4.4.1 Descrição da Prova de Conceito

A pesquisa exploratória realizada, juntamente com o referencial teórico levantado, revelou que será necessário a utilização das plataformas: JADE, WADE, AMUSE e Android para a aplicação do paradigma MultiAgentes em *Mobile Social Games*.

Para validar as integrações entre as plataformas mencionadas anteriormente, proporcionar a configuração do ambiente de desenvolvimento, e criar um subsídio reutilizável para o desenvolvimento da proposta, capítulo 5, a seguinte Prova de Conceito foi desenvolvida: uma versão inicial de controle de autenticação de usuários através de nome e senha, conforme Figura 12.

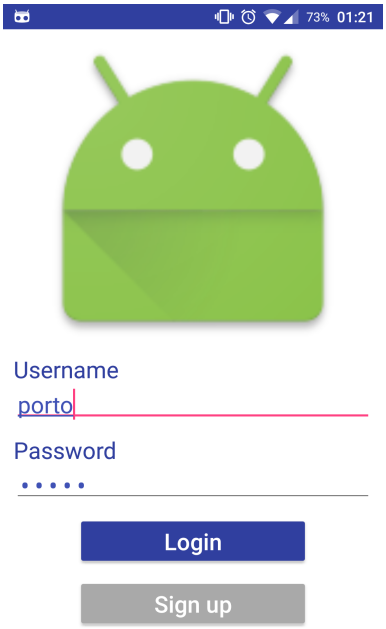


Figura 12: Protótipo de tela de login para Prova de Conceito

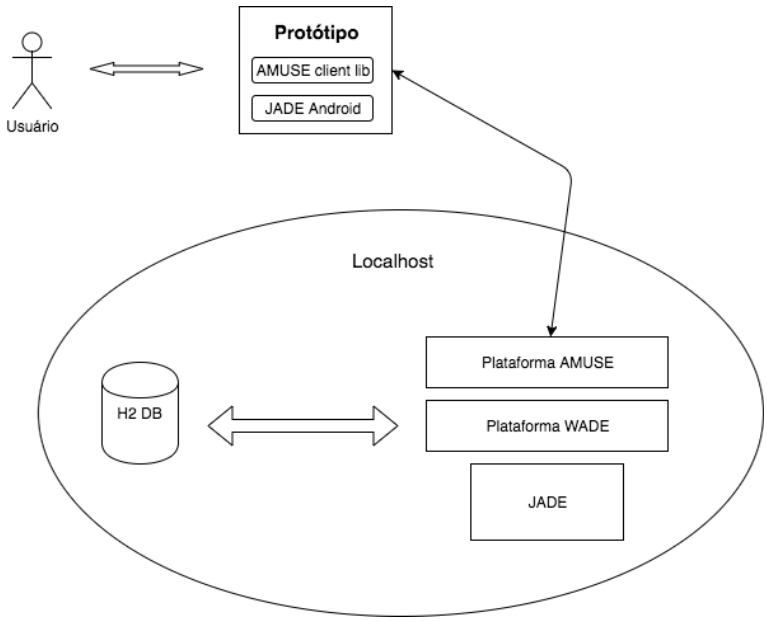


Figura 13: Visão arquitetural alto nível da Prova de Conceito

Uma visão arquitetural alto nível da Prova de Conceito pode ser vista na Figura 13. Essa Prova de Conceito possui a integração entre as plataformas, e utiliza o banco de dados SQL H2 Database (ENGINE, 2017), devido sua simplicidade de integração. No entanto, esse banco pode ser alterado, caso necessário.

O protótipo de aplicativo Android é disposto na Figura 13 e descrito através do diagrama de classes na Figura 14. Ele utiliza as bibliotecas AMUSE *client* e JADE Android para o controle, e comunicação do processo de autenticação do usuário com a plataforma

local AMUSE.

Esse processo de autenticação é realizado através das classes *DefaultLoginManager*, *CallbackManager* e *AmuseClient*. A classe *AmuseClient* é responsável por estabelecer a comunicação com a plataforma AMUSE, a classe *CallbackManager* é utilizada para o tratamento das funções de *callback*, e a classe *DefaultLoginManager* é uma classe utilitária que fornece os métodos de autenticação simples através de nome e senha.

Esta primeira versão de controle de autenticação permitiu que os fluxos de integração entre as plataformas fossem validados, e o ambiente de desenvolvimento configurado. Além de ser um módulo reutilizável para a proposta.

O tópico a seguir descreve os passos necessários para a configuração do ambiente de desenvolvimento, utilizando como base as ferramentas de desenvolvimento, seção 3.2.3.

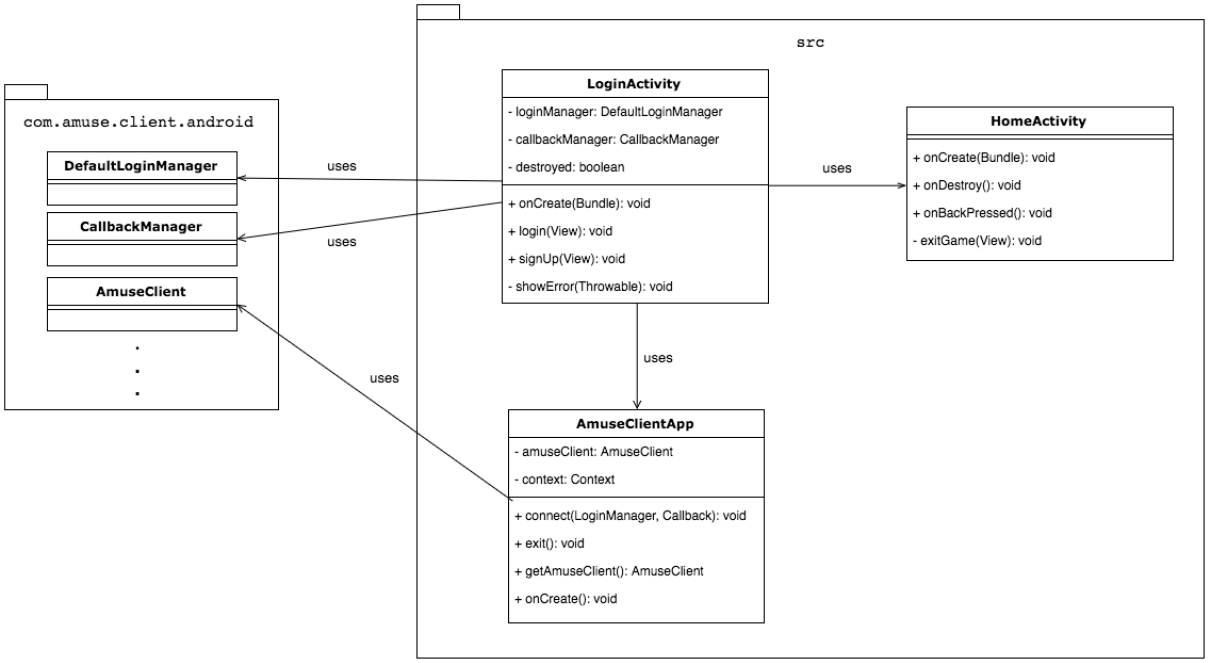


Figura 14: Diagrama de classe do protótipo

### 4.4.2 Configuração do Ambiente de Desenvolvimento

Para realizar o desenvolvimento da Prova de Conceito, utilizou-se de uma máquina com as seguintes especificações: processador 2.6 GHz Intel Core i5, memória ram de 16 GB 1600 MHz DDR3 e sistema operacional MacOS Sierra 10.12.5.

O primeiro passo para a configuração do ambiente foi a verificação da existência do Java (ORACLE, 2017), e Apache Ant (FOUNDATION, 2017) na máquina de desenvolvimento. Visto que as plataformas utilizadas nesse projeto necessitam dessas duas tecnologias para a configuração base.

Após a verificação da existência do Java, seguiu-se os passos disponíveis para a configuração do AMUSE. Tais passos estão disponíveis no tutorial de configuração, disponível em: <http://jade.tilab.com/amuse/doc/Amuse-Startup-Guide.doc>.

## 4.5 Considerações Finais do Capítulo

O objetivo desse capítulo foi apresentar a metodologia utilizada no desenvolvimento desse trabalho de conclusão de curso. Esse capítulo possui a classificação do tipo de pesquisa, modelagem do fluxo de atividades, cronograma, e prova de conceito para evolução da proposta desse TCC.

# 5 Proposta

A pesquisa exploratória realizada, e o referencial teórico levantado, expuseram a plataforma AMUSE como única ferramenta de desenvolvimento de *Mobile Social Games* que utiliza uma abordagem MultiAgentes. Levando em consideração essa condição, características desejadas fornecidas pela plataforma, e objetivo desse trabalho, ela será a ferramenta base para o desenvolvimento.

Até o momento da escrita desse projeto, os agentes da plataforma AMUSE que aplicam os padrões de *design* de jogos não estão completos. Por exemplo, os agentes GRA(*Games Room Agent*) e MMA(*Match Tracer Agent*) só permitem que um tipo de interação entre jogadores seja realizada, sendo ela assíncrona, ou síncrona (BERGENTI; CAIRE; GOTTA, 2013).

Levando em consideração as restrições presentes na ferramenta AMUSE, e consequentemente a abordagem MultiAgentes em *Mobile Social Games*, a proposta desse trabalho é restringinda a um conjunto de características e padrões de *design* predefinidos.

## 5.1 Definição da Proposta

A proposta desse trabalho consiste em desenvolver um *Mobile Social Game* para a plataforma Android com as seguintes características base de *Mobile Social Games*:

- Interações assíncronas entre os jogadores;
- Formação de laços de amizade assimétrica;
- Laço fraco na formação de relações sociais;

Os padrões de *design* de jogos que serão aplicados a princípio são:

- *Ticket-based game*;
- *Public player statistics*;

O estilo do jogo(*gameplay*) será baseado no *Online Trading Card Games*. E o modelo de monetização a ser aplicado é o *Freemium*.

## 5.2 Arquitetura do Sistema

TODO

## 5.3 Condução do Desenvolvimento do Projeto

TODO

## 5.4 Considerações Finais do Capítulo

TODO

# 6 Resultados Obtidos

TODO

## 6.1 Considerações Finais

TODO

# Referências


ALTAY, O. *Top 10 Most Profitable MMOs and MMORPGs*. 2015. <http://mmos.com/editorials/most-profitable-mmos-mmorpgs>. Accessed: 2015-13-11. Citado na página 17.

ANDROID, G. *Android Testing*. 2015. <https://developer.android.com/tools/testing/testing_android.html>. Accessed: 2015-12-10. Citado na página 25.

ANDROID, G. *Dashboards*. 2015. <http://developer.android.com/about/dashboards/index.html>. Accessed: 2015-15-11. Citado na página 19.

ANNIE, A. *App Annie Index: Market Q1 2015*. 2015. <http://blog.appannie.com/app-annie-index-market-q1-2015/>. Accessed: 2015-15-11. Citado na página 18.

ATOM. 2015. <https://atom.io>. Accessed: 2015-12-10. Citado na página 32.

BELLIFEMINE, F.; CAIRE, G.; GREENWOOD, D. *Developing Multi-Agent Systems with JADE*. [S.l.]: John Wiley & Sons, 2007. v. 2. Citado 4 vezes nas páginas 12, 22, 23 e 27.

BERGENTI, F.; CAIRE, G.; GOTTA, D. An overview of the amuse social gaming platform. In: *WOA AI IA*. [S.l.: s.n.], 2013. p. 85–90. Citado 2 vezes nas páginas 15 e 41.

BERGENTI, F.; CAIRE, G.; GOTTA, D. Agents on the move: Jade for android devices. In: *Procs. Workshop From Objects to Agents*. [S.l.: s.n.], 2014. Citado na página 28.

BERGENTI, F.; CAIRE, G.; GOTTA, D. A scalable platform for mobile social gaming. In: ACM. *Proceedings of the 30th Annual ACM Symposium on Applied Computing*. [S.l.], 2015. p. 2239–2244. Citado 4 vezes nas páginas 12, 28, 30 e 31.

BERGENTI, F.; HUHNS, M. N. On the use of agents as components of software systems. 2014. Citado na página 12.

CAIRE, G. *WADE USER GUIDE*. 2013. <http://jade.tilab.com/wade/doc/WADE-User-Guide.pdf>. Accessed: 2015-17-11. Citado na página 29.

CAIRE, G. *AMUSE START-UP GUIDE*. 2015. <http://jade.tilab.com/amuse/doc/Amuse-Startup-Guide.doc>. Accessed: 2015-10-09. Citado na página 30.

CAIRE, G.; QUARANTOTTO, E.; SACCHI, G. Wade: an open source platform for workflows and agents. In: *MALLOW*. [S.l.: s.n.], 2009. Citado na página 28.

CIPSOFT. *TibiaME*. 2015. <https://www.cipsoft.com/index.php/en/products/tibiame>. Accessed: 2015-13-11. Citado na página 17.

CIPSOFT. *TibiaME History*. 2015. <http://www.tibiame.com/?section=screenshots&lan=en&markup=xhtml&phonetype=classic>. Accessed: 2015-13-11. Citado 2 vezes nas páginas 17 e 18.

COLE, H.; GRIFFITHS, M. D. Social interactions in massively multiplayer online role-playing gamers. 2007. Citado na página 15.

CONFEDERATION, G. M. G.; NEWZOO; TALKINGDATA. *Global Mobile Game Industry White Book*. 2015. <http://2015.gmgc.info/ENG_GMGC_Newzoo_Global_Mobile_Games_Market_Whitepaper_Final.pdf>. Accessed: 2015-15-11. Citado 2 vezes nas páginas 19 e 20.

CORPORATION, I. D. *Smartphone OS Market Share, 2015 Q2*. 2015. <http://www.idc.com/prodserv/smartphone-os-market-share.jsp>. Accessed: 2015-15-11. Citado 2 vezes nas páginas 18 e 19.

DIETERLE, E.; CLARKE, J. Multi-user virtual environments for teaching and learning. 1997. Citado 2 vezes nas páginas 11 e 16.

ECLIPSE. 2015. <https://www.eclipse.org/org/>. Accessed: 2015-12-10. Citado 2 vezes nas páginas 32 e 33.

ENGINE, H. D. *H2 Database*. 2017. <http://www.h2database.com/html/main.html>. Accessed: 2017-05-06. Citado na página 37.

FENTON, N.; BIEMAN, J. *Software Metrics: A Rigorous and Practical Approach, Third Edition*. [S.l.]: Taylor & Francis Group, 2015. Citado na página 26.

FIELDS, T. *Mobile & Social Game Design: Monetization Methods and Mechanics*. [S.l.]: A K Peters/CRC Press, 2014. v. 2. Citado 6 vezes nas páginas 15, 16, 17, 18, 19 e 20.

FONSECA, J. J. S. d. Metodologia da pesquisa científica. *Fortaleza: UEC*, p. 65–75, 2002. Citado na página 33.

FOUNDATION, A. S. *Apache Ant*. 2017. <http://ant.apache.org/>. Accessed: 2017-05-06. Citado na página 38.

GIL, A. C. Como elaborar projetos de pesquisa. *São Paulo*, v. 5, p. 61, 2002. Citado na página 33.

GIT. 2015. <https://git-scm.com>. Accessed: 2015-12-10. Citado na página 31.

GITHUB. 2015. <https://github.com>. Accessed: 2015-12-10. Citado na página 31.

GOOGLE. *Android Studio*. 2017. <https://developer.android.com/studio/index.html>. Accessed: 2017-06-04. Citado na página 32.

ITALIA, T.; PARMA, U. of. *JAVA Agent DEvelopment Framework*. 2014. <http://jade.tilab.com/>. Accessed: 2015-20-10. Citado 2 vezes nas páginas 21 e 27.

ITALIA, T.; PARMA, U. of. *JADE Architecture Overview*. 2015. <http://jade.tilab.com/documentation/tutorials-guides/jade-administration-tutorial/architecture-overview/>. Accessed: 2015-20-10. Citado na página 28.

ITALIA, T.; PARMA, U. of. *Workflows and Agents Development Environment*. 2015. <http://jade.tilab.com/wadeproject/>. Accessed: 2015-17-11. Citado na página 29.

JENNINGS, N. R.; SYCARA, K.; WOOLDRIDGE, M. A roadmap of agent research and development. *Autonomous agents and multi-agent systems*, Kluwer Academic Publishers, v. 1, n. 1, p. 7–38, 1998. Citado 2 vezes nas páginas 21 e 22.

JUNIT. 2015. <http://junit.org>. Accessed: 2015-12-10. Citado 2 vezes nas páginas 24 e 31.

KING. *Match-three puzzle video game*. 2015. <http://candycrushsaga.com>. Accessed: 2015-05-10. Citado na página 11.

KING, B.; BORLAND, J. *Dungeons and Dreamers: The Rise of Computer Game Culture from Geek to Chic*. [S.l.]: McGraw-Hill Osborne Media, 2003. v. 1. Citado na página 11.

LATEX. 2015. <https://www.latex-project.org>. Accessed: 2015-12-10. Citado na página 32.

LUO, L. Software testing techniques. *Institute for software research international Carnegie mellon university Pittsburgh, PA*, v. 15232, n. 1-19, p. 19, 2001. Citado 2 vezes nas páginas 23 e 24.

MEIRELLES, P. Monitoramento de métricas de código fonte em projetos de software livre. Instituto de Matemática e Estatística, Universidade de São Paulo, 2013. Citado 2 vezes nas páginas 25 e 26.

MYERS, G. J.; BADGETT, T.; SANDLER, C. *The art of Software Testing*. [S.l.]: John Wiley & Sons, 2011. v. 3. Citado 2 vezes nas páginas 23 e 24.

NARDI, B.; HARRIS, J. Strangers and friends: Collaborative play in world of warcraft. In: ACM. *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*. [S.l.], 2006. p. 149–158. Citado na página 17.

ORACLE. *JAVA*. 2017. <http://www.oracle.com/technetwork/pt/java/javase/downloads/index.html>. Accessed: 2017-05-06. Citado na página 38.

OSX. 2016. <http://www.apple.com/osx/>. Accessed: 2017-27-03. Citado na página 32.

PARK, C.; LEE, J. H. Factors influencing the accessibility of online social game. 2012. Citado 2 vezes nas páginas 11 e 15.

RICCHETTI, M. *What Makes Social Games Social?* 2012. <http://www.gamasutra.com/view/feature/6735/what_makes_social_games_social.php?print=1>. Accessed: 2010-12-10. Citado na página 15.

ROCHA, A. d.; MALDONADO, J.; WEBER, K. *Qualidade de Software - Teorica e Prática*. [S.l.]: Prentice Hall, 2001. Citado na página 25.

SPA, T. I. *Agent-based Multi-User Social Environment*. 2015. <http://jade.tilab.com/amuseproject>. Accessed: 2015-10-09. Citado 2 vezes nas páginas 12 e 30.

SUPERCELL. *Freemium mobile MMO strategy video game*. 2015. <http://clashofclans.com>. Accessed: 2015-05-10. Citado na página 11.

THEODORSON, G. A.; THEODORSON, A. G. *A modern dictionary of sociology*. [S.l.]: Methuen London, 1970. Citado na página 33.

THURAU, C.; BAUCKHAGE, C. Analyzing the evolution of social groups in world of warcraft®. In: IEEE. *Computational Intelligence and Games (CIG), 2010 IEEE Symposium on*. [S.l.], 2010. p. 170–177. Citado na página 17.

WINKIE, L. *The People Who Still Play World of Warcraft Like It's 2006*. 2015. <http://kotaku.com/the-people-who-still-play-world-of-warcraft-like-it-s-2-1704465372>. Accessed: 2015-13-11. Citado na página 17.

WOOLDRIDGE, M. *An Introduction to MultiAgent Systems*. [S.l.]: John Wiley & Sons, 2009. v. 2. Citado 2 vezes nas páginas 12 e 22.

YAMAKAMI, T. A 3-stage transition model of the architecture of mobile social games: Lessons from mobile social games in japan. 2011. Citado na página 12.

YEE, N. Motivations for play in online games. *CyberPsychology & behavior*, Mary Ann Liebert, Inc. 2 Madison Avenue Larchmont, NY 10538 USA, v. 9, n. 6, p. 772–775, 2006. Citado na página 11.