UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA
CIÊNCIA DA COMPUTAÇÃO

ALEXANDRE BERNDT

Uma Arquitetura para Desenvolvimento de Aplicações Gamificadas para Suporte ao Paciente com Alzheimer

Goiânia
2017

sistema de bibliotecas ufg

UFG

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR AS TESES E DISSERTAÇÕES ELETRÔNICAS NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei nº 9610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou *download*, a título de divulgação da produção científica brasileira, a partir desta data.

1. Identificação do material bibliográfico: **[X] Dissertação** **[ ] Tese**

2. Identificação da Tese ou Dissertação

Nome completo do autor: Alexandre Berndt

Título do trabalho: Uma Arquitetura para Desenvolvimento de Aplicações Gamificadas para Suporte ao Paciente com Alzheimer

3. Informações de acesso ao documento:

Concorda com a liberação total do documento [x] SIM [ ] NÃO[1]

Havendo concordância com a disponibilização eletrônica, torna-se imprescindível o envio do(s) arquivo(s) em formato digital PDF da tese ou dissertação.

Assinatura do (a) autor (a)

Data: _04_ / _05_ / _2017_

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. A extensão deste prazo suscita justificativa junto à coordenação do curso. Os dados do documento não serão disponibilizados durante o período de embargo.

ALEXANDRE BERNDT

Uma Arquitetura para Desenvolvimento de Aplicações Gamificadas para Suporte ao Paciente com Alzheimer

Dissertação apresentada ao Programa de Pós–Graduação do Instituto de Informática da Universidade Federal de Goiás, como requisito parcial para obtenção do título de Mestre em Ciência da Computção.

Área de concentração: Ciência da Computação.

Orientador: Prof. Dr. Eduardo Simões de Albuquerque

Goiânia
2017

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Berndt, Alexandre
Uma Arquitetura para Desenvolvimento de Aplicações Gamificadas para Suporte ao Paciente com Alzheimer [manuscrito] / Alexandre Berndt. - 2017.
229 f.

Orientador: Prof. Dr. Eduardo Simões de Albuquerque.
Dissertação (Mestrado) - Universidade Federal de Goiás, Instituto de Informática (INF), Programa de Pós-Graduação em Ciência da Computação, Goiânia, 2017.
Bibliografia. Apêndice.
Inclui siglas, lista de figuras, lista de tabelas.

1. Arquitetura de Gamificação. 2. Gamificação. 3. Doença de Alzheimer. 4. Terapia de Reminiscência. I. Simões de Albuquerque, Eduardo, orient. II. Título.

CDU 004

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UFG

ATA Nº 03/2017

ATA DA SESSÃO DE JULGAMENTO DA DISSERTAÇÃO DE MESTRADO DE ALEXANDRE BERNDT

Aos cinco dias do mês de abril de dois mil e dezessete, às treze horas e quarenta e cinco minutos, na sala 151 do Instituto de Informática da Universidade Federal de Goiás, Campus Samambaia, reuniu-se a banca examinadora designada na forma regimental pela Coordenação do Curso para julgar a dissertação de mestrado intitulada **"Uma Arquitetura para Desenvolvimento de Aplicações Gamificadas para Suporte ao Paciente com Alzheimer",** apresentada pelo aluno Alexandre Berndt como parte dos requisitos necessários à obtenção do grau de Mestre em Ciência da Computação, área de concentração Ciência da Computação. A banca examinadora foi presidida pelo orientador do trabalho de dissertação, Professor Doutor Eduardo Simões de Albuquerque (INF/UFG), tendo como membros os Professores Doutores Sérgio Teixeira de Carvalho (INF/UFG) e Sibelius Lellis Vieira (PUG/GO). Aberta a sessão, o candidato expôs seu trabalho. Em seguida, o aluno foi arguido pelos membros da banca e:

( ) tendo demonstrado suficiência de conhecimento e capacidade de sistematização do tema de sua dissertação, a banca concluiu pela **aprovação** do candidato, sem restrições.

(X) tendo demonstrado suficiência de conhecimento e capacidade de sistematização do tema de sua dissertação, a banca concluiu pela **aprovação** do candidato, condicionado a satisfazer as exigências listadas na Folha de Modificação de Dissertação de Mestrado anexa à presente ata, no prazo máximo de 60 dias, a contar da presente data, ficando o professor-orientador responsável por atestar o cumprimento dessas exigências.

( ) não tendo demonstrado suficiência de conhecimento e capacidade de sistematização do tema de sua dissertação, a banca concluiu pela **reprovação** do candidato.

Os trabalhos foram encerrados às 17 horas. Nos termos do Regulamento Geral dos Cursos de Pós-Graduação desta Universidade, lavrou-se a presente ata que, lida e julgada conforme, segue assinada pelos membros da banca examinadora.

Prof. Dr. Eduardo Simões de Albuquerque _____

Prof. Dr. Sérgio Teixeira de Carvalho _____

Prof. Dr. Sibelius Lellis Vieira _____

Alexandre Berndt

Possui graduação Em Ciência da Computação (Universidade do Vale do Itajaí) e especialização em Informática na Educação (Universidade Federal de Lavras). Docente efetivo do curso de Ciência da Computação (Universidade do Estado de Mato Grosso). Tem experiência na área de Ciência da Computação, com ênfase em Linguagens de Programação, atuando principalmente nos seguintes temas:Orientação Objeto, Lógica de Programação, Estrutura de Dados.

Dedico este trabalho à minha noiva Cinthia, aos meus queridos pais Arlindo e Margarete, aos meus irmãos Andréa e Marcelo, à minha sobrinha Bianca e a todos os meus familiares, que guardo no coração.

Ao Prof. Dr. Eduardo Simões de Albuquerque, que me orientou ao longo desta jornada.

Agradecimentos

Inicio agradecendo meu orientador, Professor Dr. Eduardo Simões de Albuquerque. A ti meus sinceros agradecimentos pela condução na busca do conhecimento e me orientar em todos os momentos partilhando seus conhecimentos e sabedoria.

Agradeço a minha noiva e futura esposa, Cinthia por me amar e suportar a distância. Obrigado por permitir que eu entrasse na sua vida.

A minha família, especialmente aos meus pais, Arlindo e Margarete, meus irmãos, Andréa e Marcelo, e minha sobrinha Bianca, pelo amor e carinho que me dedicam.

Aos meus colegas do mestrado, em especial os colegas, Paulo e Áurea companheiros em todos os momentos. Momentos de alegria, tristeza e tensão, que passamos durante o mestrado.

Aos professores da Universidade Federal de Goiás, da pós-graduação, que partilharam seus conhecimentos conosco, contribuindo para o meu crescimento como pessoa.

Aos servidores do INF/UFG, pela disposição em nos atender com toda atenção e competência.

Agradeço à Universidade do Estado de Mato Grosso por permitir o afastamento das minhas funções, para a realização do mestrado em Ciência da Computação.

Aos meus amigos e colegas do campus de Barra do Bugres que incentivaram nesta jornada e torceram por mim.

Agradeço à Fundação de Amparo à Pesquisa do Estado de Mato Grosso pelo apoio financeiro para o desenvolvimento dessa dissertação.

Resumo

Berndt, Alexandre. **Uma Arquitetura para Desenvolvimento de Aplicações Gamificadas para Suporte ao Paciente com Alzheimer**. Goiânia, 2017. 229p. Dissertação de Mestrado. Ciência da Computação, Instituto de Informática, Universidade Federal de Goiás.

O uso da *gamificação* como técnica de motivação, engajamento e interação de aplicações móveis tem se tornado popular, impulsionado pelos *smartphones* e a capacidade destes dispositivos oferecerem designs de jogos atraentes. No contexto da saúde o uso de elementos de *gamificação* tem contribuído na motivação e engajamento aos tratamentos médicos e à mudança de hábitos e bem-estar dos pacientes de todas as idades. Quando o paciente a ser assistido é idoso, dificuldades extras podem ser acrescidas à motivação e engajamento a tratamentos médicos. A *gamificação* contribui para tornar mais atrativo apicativos computacionais de apoio terapêutico. No entanto, identificar os elementos de jogos mínimos necessários e o tempo despendido para desenvolver estes aplicativos, restringem a sua produção. Este trabalho propõe uma arquitetura de componentes, para desenvolvimento de aplicações móveis *gamificadas* de apoio à reminiscência de pacientes com a doença de Alzheimer, auxiliando na adesão ao processo terapêutico, ampliando os efeitos positivos provocados pela terapia de reminiscência, ligados à sensação de bem-estar e no retardo dos sintomas provocados pela doença de Alzheimer. Foram identificados dezesete componentes de gamificação integram a arquitetura definida e implementada. Dois protótipos de aplicativos foram desenvolvidos para exemplificar e validar a proposta de arquitetura de *gamificação*.

Palavras–chave

Arquitetura de Gamificação, Gamificação, Doença de Alzheimer, Terapia de Reminiscência.

Abstract

Berndt, Alexandre. **An Architecture for the Development of Gamified Applications for Patient Support with Alzheimer's**. Goiânia, 2017. 229p. MSc. Dissertation. Ciência da Computação, Instituto de Informática, Universidade Federal de Goiás.

The use of gamification as a motivation, engagement, and interaction technique for mobile applications has become popular, boosted by smartphones and the ability of these devices to deliver compelling game designs. In the context of health, the use of gamification elements has contributed to the motivation and engagement to medical treatments and to the change of habits and well-being of patients of all ages. The elderly are a growing portion of the population and age-related diseases are becoming common, among them are dementias such as Alzheimer's disease. When the patient is elderly, extra difficulties can be added to motivation and to medical treatments supported by computerized tools. However, identifying the minimum gaming elements needed and the time spent to develop these applications restrict their production. This work proposes an architecture of components for the development of mobile applications to support reminiscence therapy of patients with Alzheimer's disease, aiding in the motivation and engagement to the therapeutic process, amplifying the positive effects caused by reminiscence therapy, leading to a well-being feeling and to retarding the symptoms caused by Alzheimer's disease. We have identified seventeen elements of gamification integrated to the architecture defined and implemented. Two application prototypes Were developed to exemplify and validate the gamification architecture proposal.

Keywords

Gamification Architecture, Gamification, Alzheimer's Disease, Reminiscence Therapy.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas e Abreviaturas

API	*Application Programming Interface*
CDR	*Clinical Dementia Rating*
COM-B	*Capability, Opportunity, Motivation, Behaviour*
DA	*Doença de Alzheimer*
DeSA	*DeStress Assistant*
DVM	*Dalvik Virtual Machine*
DTD	*Document Type Definition*
FBM	*Fogg Behaviour Model*
Gamificação	*Aplicação de técnicas de design de jogos*
GoF	*Gang of Four*
HIV	*Human Immunodeficiency Virus*
ID	*Identity*
IHC	*Interação Humano-Computador*
MEEM	*Mini Exame do Estado Mental*
NFC	*Near Field Communication*
SDK	*Software Development Kit*
SQL	*Structured Query Language*
SQLite	*Structured Query Lite*
SMS	*Short Message Service*
TEC	*Terapia de Estimulação Cognitiva*
TIC	*Tecnologia da Informação e Comunicação*
TR	*Terapia de Reminiscência*
XML	*Extensible Markup Language*

CAPÍTULO 1

Introdução

O termo *gamificação* originado na indústria da mídia digital é definido, por Deterding e colaboradores [30], como a aplicação de técnicas de *design* e mecânica de jogos, com a finalidade de motivar e engajar usuários na solução de problemas do mundo real, em contextos alheios aos jogos.

Nos diversos contextos de utilização da *gamificação*, o desenvolvimento de aplicações voltadas para o bem-estar e o acompanhamento de dados relacionados à saúde pessoal tem auxiliado a mudança do comportamento das pessoas [64].

A motivação pode ser dividida em dois tipos [49]: motivação intrínseca e motivação extrínseca. A motivação intrínseca tem importante papel na alteração do comportamento uma vez que, está relacionada à satisfação interna do ser humano em realizar uma atividade. A mudança de hábitos ligados à saúde é um exemplo de motivação intrínseca. Já a motivação extrínseca tem um efeito menos duradouro, mas mais rápido de ser percebido uma vez que, é representada por recompensas virtuais ou físicas ligadas a um bom trabalho executado.

Segundo Kapp [49] um misto das motivações é o que se deseja para uma mudança comportamental efetiva.

A tecnologia tem contribuído para auxiliar a mudança de comportamento. Um exemplo dessa contribuição pode ser vista no crescimento de aplicações *gamificadas* ligadas à saúde. Dois fatores importantes têm contribuindo para o surgimento de aplicações *gamificadas* na saúde: a busca pelo monitoramento da saúde e a popularização de *smartphones* em todas as faixas etárias [50].

Os idosos são uma parcela da população brasileira e mundial em crescimento. A expectativa de vida populacional vem se tornando maior, com isso as doenças provenientes da idade surgem com mais frequência [67].

Doença de caráter neurodegenerativo sem cura, a doença de Alzheimer compromete o funcionamento do cérebro, afetando funções cognitivas (memória, percepção, localização espaço temporal, funções executivas, atenção e linguagem), humor e comportamento dos pacientes [18].

No combate aos sintomas provocados pela doença de Alzheimer, estratégias farmacológicas e não farmacológicas de tratamento têm sido adotadas. A preferência por terapias não medicamentosas tem ganhado espaço, pela baixa eficiência dos fármacos e os efeitos colaterais provocados no paciente [6, 42].

Um exemplo de estratégia terapêutica de foro psiquiátrico, a terapia de reminiscência estimula a recuperação de memórias antigas e recentes do paciente. Durante uma sessão terapêutica artefatos são utilizados como gatilho da reminiscência [44].

A terapia de reminiscência tem contribuído para o tratamento da depressão, ocorrência comum em pacientes diagnosticados com doença de Alzheimer [22, 57]. Contudo, pessoas diagnosticadas com um quadro de Alzheimer tem uma dificuldade a mais para manter seus níveis de motivação, mesmo sabendo que atividades terapêuticas podem ajudar a reduzir os efeitos da doença.

A *gamificação* tem se mostrado uma alternativa viável na tentativa de aumentar a motivação das pessoas [64]. Ao integrar às aplicações móveis construídas, para apoiar as sessões de terapia de reminiscência, a *gamifiocação* pode contribuir para potencializar os efeitos terapêuticos positivos da reminiscência.

1.1 Motivação

Apesar do crescimento do uso da *gamificação* em diversos contextos, *gamificar* uma aplicação é algo complexo de ser feito. A inserção indiscriminada dos elementos de jogos não garante uma *gamificação* efetiva. Deve-se também considerar o público-alvo para se ampliar as chances de sucesso da aplicação [79].

A diversidade de nichos de utilização e a baixa frequência no uso da *gamificação* apontam a dificuldade de determinar quais elementos de jogos são adequados na construção de uma aplicação [20].

A forma inapropriada de identificar os elementos de jogo adequados é um fator complicador no desenvolvimento de aplicações para auxiliar tratamentos de saúde, pois potencializam as chances de falhas de projeto. Ao não atingir a finalidade para a qual foi concebida a aplicação, as melhoras no estado de saúde do paciente ficam sem efeito.

O progressivo declínio das habilidades dos pacientes provocados pela doença contribui para a redução da efetividade das aplicações, havendo a necessidade de reformulação desses *softwares* para continuar a atender a nova condição do paciente. Uma arquitetura de *gamificação* pode facilitar a construção de novas aplicações adequadas às atuais habilidades do paciente.

Três usuários tem papel importante no uso da arquitetura proposta neste trabalho. O cuidador/familiar, sendo conhecedor das condições e habilidades do paciente, pode construir com o desenvolvedor na criação de aplicações adequadas à nova realidade do

paciente. O paciente participativo fornece informações sobre suas preferências (*interface de software*, acontecimentos significativos relacionados a sua vida etc). O desenvolvedor munido das informações adivinda do paciente e cuidador utiliza da arquitetura de *gamificação*, para a criação e desenvolvimento de aplicações.

1.2 Problema

Identificar os elementos de *gamificação* e como utilizá-los no desenvolvimento de aplicações móveis, de suporte a sessões terapêuticas de reminiscência ministradas a pacientes diagnosticados com doença de Alzheimer.

1.3 Objetivos

O objetivo principal é criar uma arquitetura de componentes, para desenvolvimento de aplicações móveis *gamificadas* de apoio à reminiscência de pacientes com doença de Alzheimer, motivando e engajando os pacientes em realizar as atividades terapêuticas, ampliando os efeitos positivos provocados pela terapia de reminiscência, ligados à sensação de bem-estar e no retardo dos sintomas provocados pela doença de Alzheimer.

Para com as seguintes fases construir a arquitetura de *gamificação*:

- Especificar o modelo de componentes de *gamificação* (módulos);
- Implementar os módulos de *gamificação*;
- Definir e criar um arquivo DTD (*Document Type Definition*) com as *tags* dos módulos de *gamificação*);
- Descrever cenários de aplicação; e
- Definir e implementar uma aplicação teste.

1.4 Contribuições

As contribuições apresentadas pela dissertação "Uma Arquitetura para Desenvolvimento de Aplicaç oes Gamificadas para Suporte ao Paciente com Alzheimer", pode ser listado:

- Identificação de 17 elementos de *gamificação* importantes no desenvolvimento de aplicações *gamificadas* para pacientes diagnosticados com Alzheimer;
- Desenvolvimento de uma arquitetura de *gamificação* que disponibiliza os 17 elementos de *gamificação* na forma de componentes de software agrupados em módulos;

- Criação de uma estrutura de banco de dados para registro evolutivo da doença de Alzheimer, perfil e habilidades restantes no paciente;
- Desenvolvimento de documento XML para uso de ferramenta de autoria, para geração automatizada de atividades terapêuticas *gamificadas*; e
- Publicação no XV Congresso Brasileiro de Informática em Saúde do artigo "Classificação para Aplicações Terapêuticas Gamificadas para Pacientes com Alzheimer.

1.5 Estruturação da Dissertação

Esta dissertação é composta por oito capítulos.

O capítulo 2 apresenta o conceito de *gamificação*, a estrutração em níveis dos elementos de *gamificação* e o uso da *gamificação* no apoio mudaną dos hábitos de saúde.

No capítulo 3 versa sobre os efeitos da doença de Alzheimer na cognição dos pacientes, o mini exame de estado mental utilizado para o acompanhamento da evolução do paciente e a reminiscência como terapia auxiliada ou não por *software*.

O capítulo 4 trata dos trabalhos relacionados à adoção de aplicações móveis para o bem-estar e exemplos de trabalhos que utilizam as TICs na terapia de reminiscência.

Na sequência o capítulo 5 são apresentadas as *strings* de busca utilizadas na revisão da literatura para fundamentação do trabalho e os requisitos de *gamificação* para aplicações voltadas para pacientes com doença de Alzheimer, o capítulo 6, apresenta a arquitetura de *gamificação* de aplicações voltadas para pacientes com doença de Alzheimer.

No capítulo 7 são apresentadas as ferramentas computacionais utilizadas para o desenvolvimento da arquitetura de *gamificação*.

A descrição dos cenários e a utilização dos componentes de *gamificação* propostos na dissertação que se encontra no capítulo 8. E na sequência um capítulo com as conclusões, limitações e trabalhos futuros.

CAPÍTULO 2

Gamificação

Neste capítulo são apresentados os elementos de jogos, *gamificação* no apoio à mudaça dos hábitos de saúde e recomendações de usabilidade e acessibilidade de *interfaces* de aplicativos móveis para idosos.

2.1 Elementos de Jogos

O termo *gamification* originado na indústria da mídia digital, foi empregado oficialmente pela primeira vez em 2008, mas a popularização aconteceu depois da segunda metade de 2010 [30]. Palavra de origem inglesa o termo *gamification*, sem tradução oficial na Língua Portuguesa, tem sido traduzido como *gamificação*.

Segundo a definição utilizada por Kapp [49] e proposta por Deterding e colaboradores [30], *gamificação* é a aplicação de técnicas de *design* e mecânica dos jogos, com a finalidade de motivar e engajar usuários na solução de problemas do mundo real, em contextos alheios aos jogos.

Dentre os diversos contextos de aplicação da *gamificação*, Prokash e Rao [66] destacam a utilização bem sucedida do emprego de elementos dos jogos na ciência.

Um exemplo de sucesso da *gamificação* à aplicação *Foldit* [43, 35] desenvolvida pela Universidade de Washington, com o objetivo de acelerar o processo de dobra de proteínas.

No uso da aplicação cada proteína dobrada de forma válida, o usuário colaborador recebe uma pontuação. As soluções melhores colocadas são analisadas pelos cientistas, para verificar se a configuração proposta pode ser utilizada realmente.

Em dezembro de 2011, as descobertas realizadas pelos usuários do *Foldit* foi manchete na Revista Nature. O grupo desvendou os segredos de uma proteína chave na luta contra o *Human Immunodeficiency Virus* (HIV), resolvendo em dez dias, com auxílio de 40.000 colaboradores *on-line*, o que os cientistas já buscavam por mais de uma década.

De modo geral, a aplicação dos elementos de *gamificação* em atividades da vida real busca estimular uma maior interação, desenvolver experiências positivas, explorar as aptidões pessoais de cada indivíduo, recompensar virtual ou fisicamente, por realizações

[79]. É importante ter em mente que a *gamificação* não significa criação de um jogo para a diversão dos usuários, mas a utilização de forma inteligente de elementos e mecanismos dos jogos, com intuito de despertar o interesse, o engajamento na realização das atividades propostas por empresas, educadores, profissionais de saúde, entre outros.

Três orientações têm sido seguidas para a utilização da *gamificação*: orientação externa, orientação interna e mudança do comportamento. Na orientação externa, a mecânica dos jogos é direcionada a criar uma identificação, o estreitamento das relações de clientes com empresas, produtos e/ou serviços. Na orientação interna, a mecânica dos jogos é direcionada para a criação de um sentimento de colaboração, competição saudável, entre os colaboradores de instituições empresariais e governamentais. Na orientação para a mudança do comportamento, a mecânica dos jogos é aplicada para alterar hábitos relacionados à saúde, às finanças, para educar motivando a construção do conhecimento de educandos, além de auxiliar na adoção de políticas públicas e sociais [79, 83].

A atração pelos jogos levou pesquisadores da área de Interação Humano Computador (IHC), a utilizarem elementos dos jogos, como mecânicas, dinâmicas e componentes. A mecânica corresponde aos processos básicos que promovem a ação e a participação das pessoas. A dinâmica corresponde aos principais aspectos considerados e administrados na *gamificação* de uma aplicação. Os componentes são elementos básicos de suporte a mecânica e dinâmica dos jogos [49].

Os elementos dos jogos são classificados em três níveis, dinâmicas, mecânicas e componentes [83]. Os níveis são apresentados a seguir:

1. **Nível das Dinâmicas**
    - **Emoções/Sentimentos:** Está relacionado à sugestão da curiosidade, competitividade, frustrações, felicidade etc;
    - **Restrições:** Corresponde às limitações ou escolhas forçadas;
    - **Narrativas:** Propicia a exposição de acontecimentos de modo continuo e consistente;
    - **Relacionamento:** Está ligado ao gerenciamento das interrelações sociais, status, altruísmo etc;
    - **Progressão:** Está relacionado ao crescimento e desenvolvimento do usuário;
2. **Nível das Mecânicas**
    - **Sorte:** Refere-se aos elementos de recompensa poderem surgir ou não de modo aleatório;
    - **Desafios:** Representam atividades que requerem esforço para serem solucionadas;
    - **Competição:** Diz respeito à vitória de um indivíduo ou grupo em detrimento da derrota de outro ou outros;

- ***Feedback*:** Está relacionado às informações sobre as ações do usuário;
- **Cooperação:** Corresponde ao trabalho conjunto dos usuários em prol de um objetivo comum;
- **Aquisição de Recursos:** Está ligada às conquistas de habilidades ou recolha de artefatos;
- **Transações:** Refere-se às trocas de recursos entre os usuários de forma direta ou trocas por meio de intermediários;
- **Recompensas:** Representa as bonificações por realizações ou ações;
- **Turnos:** Está relacionado à forma de participação dos usuários (sequencial ou em turnos); e
- **Estados de Vitória:** Está ligado aos objetivos que tornam um usuário ou grupo vencedor ou perdedor.

3. **Nível dos Componentes**

- **Avatares:** Está relacionado com a representação dos usuários no ambiente virtual;
- **Insígnias:** Responsável pela representação visual das conquistas relacionadas aos objetivos;
- **Conquistas:** Refere-se à representação virtual dos feitos;
- **Desafio de Níveis:** Está relacionado aos grandes desafios para mudança de nível;
- **Coleções:** Correspondente ao conjunto de itens acumulados;
- **Combates:** Está ligado com as batalhas e disputas individuais ou em grupo dentro do ambiente virtual;
- **Desbloqueio de Conteúdo:** Representa a liberação de novos conteúdos à medida que o usuário progride no ambiente virtual;
- **Doação:** Está relacionado à possibilidade de partilha de recursos entre usuários;
- **Tabela de Líderes:** Está ligada à colocação de um usuário em relação aos outros usuários;
- **Níveis:** Está associado aos passos de progressão dentro do ambiente virtual;
- **Pontos:** refere-se à representação numérica da progressão do usuário no ambiente virtual;
- **Missões:** Está relacionada com o estabelecimento de desafios predefinidos e os objetivos almejados e compensações;
- **Gráfico Social:** Representa a rede de contados do usuário dentro do ambiente virtual;
- **Equipes:** Está ligado ao grupo de usuários que atua junto em prol de um objetivo comum; e

- **Bens Virtuais:** Representa aos recursos virtuais que podem ser trocados por itens virtuais ou moeda real.

Motivar extrinsecamente alguém consiste no ato de oferecer recompensa por um bom trabalho realizado. Para Werbach e Hunter [83], no processo de *gamificação* a motivação extrínseca pode ser utilizada com os seguintes objetivos:

- Ampliar a satisfação na realização de uma atividade;
- Reforçar o sentimento de autonomia;
- Engajar atividades de baixo valor de interesse;
- Aumentar o foco na atividade reduzindo o tempo de realização; e
- Envolver-se em atividades de importância desconhecida.

As recompensas físicas, exemplo de ganho de milhagens, são uma boa representação de motivação extrínseca.

A motivação intrínseca está relacionada à satisfação interna do ser humano, ao realizar uma atividade, com objetivo de despertar sensações de autonomia (sensação de controle da situação), competência (capacidade de controlar a aprendizagem dos conteúdos) e proximidade (sensação de ligação com os outros por meio das ideias) [82]. As sensações destacadas por Werbach e Hunter [83] podem ser mapeadas para a *gamificação* da seguinte maneira:

- Sensação de controle;
- Confiança na capacidade de realização de desafios;
- Clareza nas formas de domínio do conhecimento;
- Recompensas incrementais ao alcançar conquistas e premiar a conquista do objetivo final; e
- Fortalecimento do sentimento de proximidade, por meio de tabelas classificatórias, desafios entre pares e outras formas de relacionamento social.

A motivação de caráter intrínseco acontece quando as pessoas realizam atividade física, geralmente com a finalidade de melhoria da condição física e psicológica.

A adição de elementos motivacionais intrínsecos e/ou extrínsecos, na construção de aplicações *gamificadas* precisa ser cuidadosamente observada, para não causar desinteresse do usuário em realizar as atividades propostas [82].

A determinação do tipo de motivação a utilizar no processo de *gamificação* de uma aplicação é algo difícil de prever, como também, qual das duas modalidades de motivação empregar. O aconselhável está em mesclar as duas formas de motivação, considerando a finalidade da aplicação a ser construída [49].

2.1.1 *Gamificação* no Apoio à Mudaça dos Hábitos de Saúde

Com a popularização da *gamificação* nos últimos anos, a indústria da saúde digital tem apostado na adoção dos elementos de jogos na construção de aplicativos móveis [23, 70].

Embora seja recente a utilização do conceito de *gamificação*, no âmbito da saúde este conceito vem se tornado familiar. Dois fatores são destacados por King e colaboradores [50]. O primeiro fator sustenta-se na popularidade dos *smartphones* e a capacidade destes dispositivos oferecerem *designs* de jogos atraentes, para utilizar na concepção de intervenções de saúde interativas. O segundo fator é o entusiasmo e a vontade dos desenvolvedores incorporarem as últimas descobertas comportamentais em intervenções eletrônicas.

No entanto, o conhecimento sobre os reais efeitos da *gamificação* na construção e a mudança de comportamento relacionados á saúde são imprecisos, não havendo um consenso de quais e como utilizar os elementos de gamificação [24, 56].

O uso da *gamificação* de modo positivo nas mudanças do comportamento na saúde tem se apoiado nas teorias da Ciência Comportamental (COM-B (*Capability, Opportunity, Motivation, Behaviour*) [61], FBM (*Fogg Behavior Model*) [37]), mostrando a relação dos princípios da *gamificação* com os princípios da tecnologia de mudança de comportamento em saúde [24, 56].

De acordo com Pereira [64], aplicações *gamificadas* relacionadas ao contexto saúde seguem duas vertentes, aplicações para manutenção do bem-estar e aplicações na saúde. Das inúmeras aplicações no contexto da saúde, os aplicativos voltados para o bem-estar são bastante populares entre os usuários de *smartphone*. Dentre eles se destacam aplicações para *fitness* reconhecidas como *exergames*.

Os *Exergames* têm sido vistos pelo setor de saúde como uma boa alternativa para a redução de custos de governos e empresas uma vez que, a manutenção da boa forma física das pessoas e especialmente dos mais velhos pode segurar as demandas, no aumento dos custos com cuidados médicos e amenizar as deficiência de equipes de saúde [17].

Exemplo da popularidade das aplicações destinadas ao bem-estar das pessoas, o aplicativo *S Health* desenvolvido pela *Samsung*, integra o pacote de aplicativos presente no *smartphone Galaxy S5*.

O aplicativo destina-se a ajudar o usuário administrar sua ginástica e bem-estar geral, pelo acompanhamento das informações e medidas relacionadas às atividades físicas realizadas. Além de artigos, materiais relacionados a hábitos saudáveis e *links* para outras aplicações relacionadas.

Plataformas como *Healthper Hub* (healthper.com) oferecem aplicações *gamificadas* personalizadas, para uma única pessoa ou grupos de pessoas. As características mo-

tivantes e envolventes da plataforma têm proporcionado uma vida mais saudável e bem-estar, para colaboradores de instituições dos mais variados setores (tecnologia, transporte, varejo, indústrias, medicamentos etc).

As aplicações desenvolvidas pela plataforma *Healthper Hub* possibilitam aos colaboradores das empresas e instituições contratantes, o engajamento e a interação dos usuários. Os canais de comunicação (*e-mail*, SMS, *chats*, redes sociais) são utilizados para compartilhar dados relacionados, a evolução da saúde e o bem-estar [54].

Pereira [64] também destaca a existência de outros tipos de aplicativos de auxílio ao bem-estar e a manutenção de hábitos saudáveis (controle do peso, hábitos alimentares, tabagismo, higiene das mãos).

No contexto da saúde a *gamificação* tem colaborado na adesão a tratamentos de longa duração, treinamento de profissionais de saúde. Os tratamentos médicos podem ser dolorosos, de recuperação lenta e muitas vezes entediantes [64].

Quando o paciente a ser assistido é idoso, dificuldades extras podem ser acrescidas à adesão e engajamento, a tratamentos médicos apoiados por ferramentas computadorizadas, devido a restrições físicas, sensoriais e/ou cognitivas, à falta de interesse ou aversão as inovações tecnológicas. Segundo Cugelman [24] a aversão à tecnologia acontece quando os elementos de persuasão envolvidos não convencem o suficiente para promover a mudança do comportamento, crenças e ações do indivíduo. Culgeman em [24] propõe sete estratégias persuasivas para um projeto de *gamificação* com influências positivas e duradouras. As pesquisas relatadas em [24, 56] destacam que a *gamificação* deve ser persuasiva e seus efeitos devem sustentar impactos a longo prazo e oferecer mais do que um efeito de curto prazo.

Do ponto de vista da medicina comportamental, a *gamificação* na saúde somente é considerada efetiva se utilizar princípios e táticas cientificamente comprovados na influência da saúde.

Cugelman em [24] mapeia sete estratégias persuasivas para a *gamificação* eficaz na mudança de comportamento, apresentadas na tabela 2.1.

Tabela 2.1: *Estratégias Persuasivas de gamificação e os ingredientes para a mudança do comportamento[24].*

Estratégias de Persuasão	**Elementos de Mudança do Comportamento**
Estabelecer metas	- Mudança comportamental efetiva - Ajuste do objetivo
Capacidade de superar desafios	- Gerenciamento visotemporal - Planejamento de ações
Dar *feedback*	- Agilidade no retorno do monitoramento comportamental - Agilidade no monitoramento comportamental
Reforço	- Recompensar comportamentos de sucesso
Comparar progresso	- Agilidade no retorno do monitoramento comportamental - Informações normativas sobre o comportamento de outras pessoas
Conectividade social	- Influências sociais (normas) - Suporte ao planejamento social/mudança social
Diversão e brincadeira	- Sem relação

As pesquisas desenvolvidas por Cota e Ishitani [21] identificaram fatores que motivam e desmotivam os usuários idosos a utilizarem jogos em dispositivos móveis, como fontes de diversão, engajamento e socialização e melhoria das funções cognitivas mostrado na tabela 2.2.

Os fatores identificados por Cota e Ishitani, aliadas as estratégias de *gamificação* relatadas em [24, 64], ampliam a probabilidade de sucesso de adesão na utilização de aplicativos móveis, inclusive construídos para o contexto da saúde.

Outro fator importante no sucesso à adesão as aplicações móveis pelos idosos se relaciona ao desenvolvimento de *interfaces* adaptadas as suas habilidades existentes nesses usuários [42, 51, 55, 76]. A tabela 2.3 traz os principais problemas de interação destacados pelos idosos, relativo a *interfaces* das aplicações móveis.

As *interfaces* mais acessíveis dos *smartphones* segundo King e colaboradores [50], têm se mostrado plataformas de *hardware* e *software* eficazes, nas intervenções de saúde (combate ao tabagismo e incentivar a adesão a medicação).

A gama de sensores embarcados e serviços aliados às melhorias de *interface* dos *smartphones* os transformaram em dispositivos importantes no desenvolvimento de aplicações *gamificadas* à saúde. Exemplos de sensores e serviços incluem serviços de *Global Positioning System* (GPS), acelerômetros embutidos (medem o movimento) e sensores externos (que podem medir a frequência cardíaca e pressão arterial).

Tabela 2.2: *Elementos que exercem efeitos positivos e negativos na motivação de jogadores idosos [21].*

Elementos de Motivação	**Elementos de desmotivação**
Jogos e seus elementos consistem em ferramentas de apoio a melhoria na qualidade de vida e tratamento das doenças cognitivas.	Os idosos não gostam de jogos com contextos violentos.
A descoberta da *interface* precisa ser intuitiva para facilitar a experiência do jogador, sem a necessidade de ler manuais antes de jogar.	Jogos fáceis são desencorajadores.
A importância do *feedback* para o jogador sobre cada evento no jogo.	Jogos com limite de tempo são inapropriados para os idosos.
As recompensas devem ser fornecidas após a conclusão de um nível ou realização de uma atividade correta.	A *interface* deve evitar pequenos elementos, controles de tela perto um do outro e muita informação em torno da tela são inadequados para os idosos.
Os níveis de dificuldade devem aumentar gradualmente à medida que o jogador melhora sua experiência.	
Os idosos preferem jogos do tipo casual.	
Os idosos prestam mais atenção em jogos com narrativas (ou histórias reais).	

Tabela 2.3: *Dificuldades de interação com interfaces móveis expressados pelos usuários idosos[42].*

Problemas	**Descrição**
Ícones pequenos	Dificuldade de visualização do ícone
Rótulos inadequados	Termos e expressões difíceis de entender
Hierarquia de *menus*	Hierarquia de menus complexa
Teclas com múltiplas funções	Não identificação das muitas funções das teclas
Assistência automatizada	Ajuda ocorre sem solicitação
Sequência de ações/navegação	Sequência de ações/navegação difíceis de realizar
Barra de *scroll*	Visualização difícil drasa barra de *scroll*
Tempo de sessão	Sessão expira muita rápido
Feedback	*Feedback* inadequado ou inexistente
Ajuda	Ajuda inadequada ou inexistente
Tamanho da fonte	Dificuldade em visualizar palavras
Deslocamento de ítens da *interface* inesperadamente	Ítem da *interface* se desloca inesperadamente sem a intenção do usuário
Contraste de cores	Contraste reduzido das cores da *interface*
Mecanismo de Busca	Busca ineficiente/ inexistente
Mensagens de erro	Mensagens de erro com termos técnicos
Tamanho do teclado	Dificuldade de visualização e uso do teclado

Por se tratarem de dispositivos cada vez mais comuns no dia a dia das pessoas, o uso de aplicações que auxiliam médicos e os próprios pacientes, a avaliar e acompanhar a evolução do quadro clínico tem se tornado mais comum [60].

2.2 Recomendações de Usabilidade e Acessibilidade para *Interface* de Aplicativos Móveis com Ênfase em Idosos

O envelhecimento consiste em um processo natural, que reduz gradativamente as habilidades cognitivas, motoras e sensoriais das pessoas. No projeto e desenvolvimento das *interfaces* de aplicações para dispositivos móveis, dois conceitos importantes têm sido utilizados: acessibilidade e usabilidade.

Ferreira e colaboradores [36] e Vechiato e Vidotti [77] conceituam acessibilidade como a possibilidade de utilização dos recursos de *software* e dispositivos a todos os usuários independentemente das condições físicas, cognitivas, intelectuais, sociais, culturais etc. E usabilidade com a qualidade da interação dos usuários e o ambiente digital oferecido por *softwares* e dispositivos.

A qualidade da interação está intimamente ligada à rapidez de aprendizado e uso de um recurso, na eficiência de uso, no grau de propensão a erros de uso e na satisfação em utilizar os recursos.

Ainda segundo [36] e [77], uma aplicação orientada a usabilidade não necessariamente pode ser acessível a todos os grupos potenciais de usuários e vice-versa. Pode ser fácil de usar para usuários comuns e inacessível para usuários com algum tipo de restrição.

A facilidade que um usuário tem em utilizar uma aplicação, não está ligada somente à usabilidade, mas está fortemente relacionada à capacidade dos usuários, em detectar, interpretar e responder às interações realizadas com o sistema [36]. Dessa forma, o sucesso da usabilidade perpassa pelo conhecimento do público-alvo, pois o perfil do usuário influencia o *design* e a avaliação da *interface* [77].

As limitações naturais dos idosos exigem um maior cuidado com a interação, ocasionadas por restrições impostas pela idade [28]. *Interfaces* restritivas afugentam usuários idosos [21, 42, 51, 55, 76]. O projeto de usabilidade e acessibilidade de *interfaces* para aplicações móveis, quando desenvolvidos para idosos exige mais cautela. As tabelas 2.4 e 2.5 adaptadas de [42, 77] mostram os cuidados necessários no desenvolver *interfaces* para idosos. A figura 2.1 representa uma *interface* móvel bem adaptada às possíveis restrições de usuários idosos. Os botões e rótulos com dimensões maiores, contraste de

cores e ícones com significados não ambíguos. Todas estas características da *interface* contribuem para a redução de dúvidas e erros de utilização.

As *interfaces* são as portas de entrada de uma aplicação e quanto menores forem as barreiras, maior é a possibilidade de a aplicação conseguir motivar e engajar o usuário aderir à aplicação [2, 27, 77].

Tabela 2.4: *Recomendações de usabilidade para aplicativos móveis para idosos.*

Princípio	**Recomendação**
Prevenção e tratamento de erros	Mensagens claras e objetivas, quando da ocorrências de erros cometidos pelo o usuário ou erros do sistema
Consistência	Terminologia deve ser clara, navegação simplificada
	Botões e menus com mesmo rótulo devem direcionar o usuário sempre para os mesmos locais
	Manter a padronização na apresentação nas informações, no posicionamento dos objetos na *interface*, e formas dos objetos
	Textos na voz ativa
Feedback	Fornecer descrição de abreviaturas ou siglas por extenso e destaque nas siglas ou abreviaturas
Visibilidade	Ícones simples e significativos
Fácil aprendizado	Ícones simples e significativos
Compatibilidade	Linguajar simples e claro
Padronização da funcionalidade e da informação	Elementos gráficos e animação pertinentes
	Evitar informações irrelevantes
	Destacar informações importantes
Affordance	Pistas de localização dos elementos da *interface*
Ajuda	Explicar o funcionamento das atividades

Tabela 2.5: *Recomendações de acessibilidade para aplicativos móveis para idosos.*

Princípio	**Recomendação**
Controle	Não utilizar menu *pull-down*
	Não utilizar ações de clique duplo
	Utilizar tempo suficiente para leitura de informações e mensagens
	Evitar rolagem automática de texto
Visibilidade	Utilizar fontes não serifadas, como arial ou helvética
	Não utilizar longos trechos de texto em caixa alta
	Utilizar tamanho de fonte 12 ou 14
	Utilizar texto alinhado a esquerda
	Utilizar área de botões, ícones, barra de rolagem mais espaçadas para facilitar o acionamento pelo toque
	Utilizar contrastes negativos
	Reduzir brilho de fundo das páginas
	Utilizar espaço entre linhas maior
Padronização da funcionalidade e da informação	Utilizar cores coerentes
	Evitar intermitência (piscar) das informações
Baixo esforço físico	Posicionar rótulos próximos aos objetos de interação (botões, campos de texto etc)
	Evitar ações repetitivas
	Evitar muitos passos e janelas para realização de ações
Fácil memorização	Não utilizar organização hierárquica profunda
	Categorizar as informações
Uso equitativo	Utilizar vários canais de reprodução de textos e mensagens (áudio e texto)
	Textos explicativos associados as imagens

Figura 2.1: *Exemplo de interface móvel acessível.*

2.3 Considerações Finais

O capítulo apresentou a definição do termo *Gamificação* proposta por Deterding e colaboradores [30] que consiste na aplicação de técnicas de *design* e mecânica dos jogos, com a finalidade de motivar e engajar usuários na solução de problemas do mundo real, em contextos alheios aos jogos.

Porém, *gamificar* uma aplicação não significa produzir um jogo para a diversão do usuário, mas motivá-lo a mudar seus hábitos relacionados as suas atividades profissionais e qualidade de vida.

A importância dos elementos de *gamificação*, como veículos facilitadores da mudança de hábitos e comportamentos ligados à saúde, seja para o bem-estar e manutenção de hábitos saudáveis [50], ou adesão a tratamentos médicos [64].

Um bom exemplo de aplicativos *gamificados* na busca pelo bem-estar e manutenção de hábitos saudáveis, os *exergames* compõem o pacote de aplicativos nativos de *smartphones* (aplicativo *S Health* no *Samsung Galaxy S5*).

Na adesão a tratamentos médicos, doenças crônicas (diabetes, tabagismo, hipertensão arterial) comuns da socidade atual, têm sido tratadas com auxílio aplicativos *gamificados*, para facilitar a adesão dos pacientes ao tratamento.

A acessibilidade consiste na possibilidade de recursos de *software* e *hardware* poderem ser utilizados por qualquer usuário, independente das suas condições físicas, cognitivas, intelectuais, sociais, culturais. E a usabilidade consiste na qualidade da interação dos usuários e o ambiente de operação oferecida por *softwares* e dispositivos [36, 77].

A acessibilidade e usabilidade são outros fatores importantes para a aceitação de aplicativos móveis, como veículos de adesão aos tratamentos médicos, principalmente se o paciente for idoso [28].

A aversão à tecnologia aliada às restrições físicas e cognitivas tem sido uma grande barreira à utilização de aplicativos móveis em muitos contextos. Entre eles, o de mudança comportamental relacionada aos hábitos saudáveis e tratamentos médicos.

A atenção nos requisitos de acessibilidade, usabilidade, a inserção dos elementos de *gamificação* no projeto e construção de aplicativos, a versatilidade e gama de recursos integrado nos dispositivos móveis fazem das aplicações móveis *gamificadas* uma ferramenta importante no tratamento e bem-estar de pacientes de todas as idades.

CAPÍTULO 3

Doença de Alzheimer e Terapia de Reminiscência

Doença de caráter neurodegenerativo, a Doença de Alzheimer (DA), foi identificada e relatada por Alois Alzheimer em 1906, após a descoberta de um novo marcador patológico (emaranhados neurofibrilares intraneuronais) identificado no caso clínico da paciente Auguste Deter.

A descoberta dos emaranhados neurofibrilares intraneuronais caracterizaram uma variação de doença senil, que mais tarde foi batizada por Kraepelin como Doença de Alzheimer [33].

Com o passar do tempo, a DA tornou-se uma das mais presentes doenças degenerativa do cérebro, que na atualidade atingiu o *status* considerado epidêmico, segundo os dados do relatório da Organização Mundial da Saúde [67].

O déficit cognitivo associado à DA é um dos principais fatores relacionados à diminuição do desempenho ocupacional e redução da qualidade de vida dos pacientes. O indivíduo acometido pela DA apresenta progressiva degradação das funções cognitivas, incluindo atenção, memória, linguagem, percepção, orientação espacial e funções executivas [19]. As funções cognitivas apontadas por Corrêa e Silva [19] são descritas por Caixeta e colaboradores [18] e expostas a seguir:

Atenção: Consiste na capacidade de estabelecer foco nas atividades executadas durante uma tarefa. O déficit de atenção surge relativamente cedo na evolução da DA, geralmente após as disfunções na memória episódica e antes do surgimento de alterações na linguagem e visuoespacial.

Memória: A principal função cognitiva afetada pela DA. No estágio inicial o paciente perde a capacidade de aprender fatos novos, afetando gradualmente outras funções cognitivas.

Linguagem: A linguagem apresenta prejuízos, quando está ligada a problemas de comunicação, que pode ocorrer pela perda da memória de curto prazo, prejudicando a formação gramatical (ex: "parriga", ao invés de barriga) e semântica (ex: a troca de palavras casa por carro); redução da fala e esquecimento de palavras.

Percepção: A percepção é considerada afetada quando pacientes têm problemas na identificação de formas e combinações quando os distúrbios atingem os níveis de discriminação sensorial.

Orientação espaço temporal: A orientação espaço temporal está relacionada à habilidade de localizar e estabelecer relações espaciais entre os objetos. Os distúrbios visuoespaciais prejudicam tarefas de orientação linear, contagem de pontos, combinação de localizações espaciais, estimação cúbica, tempo.

Funções Executivas: As funções executivas são as habilidades que envolvem capacidades de planejamento, organização, sequenciamento, abstração, tomada de decisão, juízo crítico e estratégico.

A perda de memória e a desorientação espaço temporal são os primeiros sintomas percebidos por familiares e amigos dos pacientes "candidatos" a sofrer com a DA [9, 18]. Testes clínicos como Mini Exame do Estado Mental (MEEM) e o *Clinical Dementia Rating* (CDR), são frequentemente aplicados na avaliação do comprometimento cognitivo-demencial em idosos com a DA [62].

3.1 Mini Exame do Estado Mental - MEEM

O Mini Exame de Estado Mental (MEEM) é utilizado por profissionais de saúde, para auxiliar no diagnóstico e avaliar o progresso da degeneração das funções cognitivas, provocados pela doença de Alzheimer, devido à sua objetividade e simplicidade de aplicação [4].

A escala de pontuação do MEEM oscila em uma faixa de zero a trinta pontos divididos em 7 categorias, cada uma delas desenhada com o objetivo de avaliar funções cognitivas específicas: orientação para tempo (5 pontos), orientação para local (5 pontos), registro de 3 palavras (3 pontos), atenção e cálculo (5 pontos), lembrança das 3 palavras (3 pontos), linguagem (8 pontos), e capacidade construtiva visual (1 ponto)[4]. Bertolucci e seus colaboradores [15] adaptaram para a realidade brasileira o teste MEEM, que pode ser verificado na tabela 3.1.

Segundo Ávila [9] o limiar de 23 pontos indica que o paciente é passível de estar acometido com a DA, porém somente a aplicação do teste MEEM não pode diagnosticar a presença da DA no paciente. Exames laboratoriais e de imagem devem ser solicitados para confirmar ou não o diagnóstico. Após o diagnóstico da DA, a adoção do MEEM é uma forma não subjetiva de quantificar e acompanhar, a progressão do declínio das funções cognitivas do paciente.

A possibilidade de cuidadores acompanharem a evolução da doença permite a condução de ações e ou atividades paliativas, que objetivam retardar os prejuízos causados

pela DA. Médicos, terapeutas e cuidadores podem buscar estimular as funções cognitivas tentando minimizar as dificuldades do paciente.

Tabela 3.1: *Mini Exame do Estado Mental [15].*

Pontuação Máxima	Pontuação do Paciente	Tipo do Teste
5 pontos		Orientação Temporal Qual é o ano? (ano, semestre, mês, data, dia)
5 pontos		Orientação Espacial Onde estamos? (Estado, cidade, bairro, hospital, andar).
3 pontos		Memória Imediata Nomeie 3 objetos. Posteriormente pergunte os 3 nomes. Dê 1 ponto para cada resposta correta. Repita até o paciente aprender. Conte e anote as tentativas.
5 pontos		Atenção e Cálculo 7 seriado 1 ponto para cada acerto. Interrompa a cada 5 perguntas. Alternativamente soletre uma palavra de forma inversa.
3 pontos		Memória de Evocação Pergunte o nome dos 3 objetos ditos e repetidos anteriormente 1 ponto para cada acerto.
9 pontos		Linguagem - Mostre dois objetos (relógio e caneta) peça para nomeá-los 1 ponto para cada acerto. - Repita o seguinte: nem aquí, nem alí, nem lá. (1 ponto). - Comando em 3 estágios (pegue o papel com a mão direita, dobre ao meio e o coloque no chão (3 pontos para os acertos). - Leia e execute a ordem: "feche os olhos" - Escreva uma frase com sentido (1 ponto). - Copie o desenho (1 ponto).

3.2 Terapia de Reminiscência

Nesta seção são abordados conceitos e fundamentos da terapia de reminiscência, a descrição de uma sessão terapêutica com reminiscência apoiada e não apoiada por computador.

A Doença de Alzheimer vem se tornando um sério problema de saúde pública afligindo milhões de pessoas ao redor do planeta. Atualmente, sem uma forma de cura efetiva, a DA tem sido tratada por meios farmacológicos e não farmacológicos [42]. As intervenções não farmacológicas têm ganhado espaço em virtude da eficiência limitada dos fármacos e dos efeitos colaterais produzidos pela medicação nos pacientes [6, 42].

Dentre as terapias não farmacológicas, a terapia de reminiscência (TR) tem sido ministrada a pacientes com DA e outros tipos de demência [6, 11, 53]. O conceito de reminiscência consiste na recuperação das memórias pessoais de forma passiva, espontânea, não estruturada comum em todas as idades [44].

A potencialização das competências está centrada nas habilidades que restam nos pacientes com DA e a reabilitação atua como um atenuador da agitação, depressão e humor dos pacientes. O processo natural de memória do indivíduo pode ser ilustrado pela recordação de fatos pessoais, como o casamento, quando o indivíduo ao passar em frente a uma vitrine e avista um vestido de noiva.

A terapia de reminiscência como estratégia estruturada, direcionada pode ser aplicada, seja para potencialização de competências, seja para reabilitação de deficiências [16]. O processo de estruturação da reminiscência é mais que um mecanismo de sequenciamento e gatilho para simples recordação, ou seja, ela atua para permitir que uma pessoa possa alcançar algum objetivo psicossocial [80].

Comumente utilizada no tratamento de pacientes com demência, a TR busca sequenciar as memórias do passado até o presente, para impactar positivamente na cognição, bem-estar, combate a depressão, comunicação, variação do humor, qualidade de vida e na redução da sobrecarga dos cuidadores [22, 57].

A atenuação da agitação e melhora do humor é incentivo à comunicação tem sido conseguido com o uso da música [68]. Um exemplo de uso da música é aplicação na abertura das sessões de TR, com o intuito de tranquilizar e facilitar a comunicação.

Segundo Bohlmeijer e colaboradores [16], a comunicação não se resume apenas a narrativa dos fatos da vida. Muitas vezes um paciente com quadro de demência tem dificuldades de se expressar verbalmente (falada, escrita). A forma não verbal é importante para estabelecer a comunicação. Ao serem estimulados sensorialmente, pelos sons, imagens, cores, cheiros e sabores fortes lembranças podem ser despertadas.

A terapia de reminiscência estimula de forma intencional a memória dos pacientes, utilizando em cada sessão ou conjunto de temas diferenciados. Segundo Webster e

colaboradores [80], os temas devem estar inseridos em contextos (familiares, institucionais, culturais, sociais etc) comuns aos pacientes.

Temas relacionadas à família incluem lembranças da infância até a vida adulta (momentos com filhos, netos, relacionamentos amorosos etc); temas institucionais ligados a trabalho, aposentadoria; temas ligados à arte, música; e temas ligados a amigos e acontecimentos sociais [44].

Em uma sessão de TR convencional, seja coletiva ou individual, utilizando recursos computacionais ou não, os pacientes são conduzidos e observados por terapeutas, que ao menor sinal de mudança comportamental ruim (agitação, melancolia, raiva) podem intervir para motivar, acalmar, confortar o paciente.

3.2.1 Realização de uma Sessão de Reminiscência Convencional

Na realização de uma sessão de reminiscência convencional, o terapeuta precisa estar munido com informações sobre a vida e as relações sociais do paciente, para poder conduzir a TR com assuntos de interesse do paciente (e.g. culinária, fatos históricos, acontecimentos pessoais, entre outros), e utilizar como estímulo durante toda a sessão terapêutica [71]. Segundo Gonçalves e colaboradores [41], a ausência de padrões para a condução de uma sessão de TR prejudicam a avaliação da sua eficácia.

No formato de sessão proposto por Lopes e colaboradores [58], a primeira sessão é utilizada para organização e preparação do material das futuras sessões. Os eventos relacionados à vida do paciente são organizados em um esforço conjunto de paciente e terapeuta.

A grade temporal organizada facilita a condução das sessões de TR, conforme destacado em [58], permite:

- Organizar os eventos mais importantes de forma cronológica;
- Dar suporte visual ao paciente para a sua orientação visual;
- Registar os acontecimentos no tempo para o inserção de material de apoio às sessões pelo terapeuta; e
- Criar o arquivo de memórias com as lembranças recuperadas em cada sessão.

O paciente é encorajado a levar para as sessões fotografias, postais, cartas, objetos, etc. A cada sessão o arquivo de memórias do paciente (Caixa de memórias, cartaz fotográfico, livro de notas autobiográficas ou gravações) é incrementado.

Em todas as sessões coletivas, assim que a sessão se inicia, é importante que o terapeuta faça as apresentações, informe os objetivos da atividade de forma simples [8]. Nas sessões individuais é importante que paciente e terapeuta se apresentem e o terapeuta explique de forma simplificada os objetivos da sessão terapêutica.

O ambiente de realização das sessões de TR precisa ser calmo, evitando a desatenção permitindo a concentração, seja confortável, familiar ao paciente. O planejamento de cada sessão deve levar em consideração as habilidades cognitivas, ouvir a opinião do paciente, registar as sessões, falar com calma e claramente. A avaliação das sessões ŕealizada pelo paciente, considerando a satisfação, sensação de objetivos alcançados e os benefícios a saúde [58].

A condução e o andamento de uma sessão de TR sempre estão sujeitos a variação das condições psicológicas do paciente.

O tempo de duração de uma sessão de TR é um exemplo da variabilidade das condições do paciente, inicialmente uma sessão projetada para 20 a 30 minutos pode ser encurtada e finalizada antes do previsto.

Durante a condução de uma sessão de TR, conforme destaca Asttel e colaboradores [8], são comuns as interrupções da comunicação, ocasionada por lapsos de memória do paciente e de concentração. Estes lapsos são ocorrências comuns em um paciente com DA, gerando desgaste para paciente e terapeuta. O paciente pode se irritar e se agitar pelas constantes perdas da linha de raciocínio durante a sessão de TR. E o terapeuta precisa estar retomando o assunto esquecido ou procurar um novo tema de interesse, para a continuidade da sessão [7].

3.2.2 Realização de uma Sessão de Reminiscência Apoiada por Computador

A popularização e versatilidade da tecnologia de informação e comunicação (TIC) permitem a utilização do computador nos mais variados contextos. A adoção do computador, como ferramenta de apoio aos pacientes com DA colabora com o envolvimento e engajamento nas sessões terapêuticas de reminiscência [3, 52].

A adoção de aplicações computadorizadas no apoio a TR possibilita uma sessão mais leve para o cuidador e atraente para o paciente uma vez que, o uso de um *software* proporciona a liberdade de escolha dos assuntos a serem conduzidos e sempre que necessário, retomados durante uma sessão de reminiscência [8, 26].

Na realização de uma sessão de TR apoiada por computador, na sua primeira execução, segundo Astell e colaboradores [7], o paciente precisa ser avaliado, para determinar o nível de acomentimento das funções cognitivas prejudicadas pela DA. E após a avaliação e inserção dos artefatos (fotografias, músicas, vídeos) fornecidos por familiares, as sessões podem ser iniciadas.

A sessão apoiada por computador propriamente dita, inicia com o terapeuta e paciente sentados lado-a-lado em frente a um dispositivo *touch screen*. A escolha de um

assunto de preferência do paciente é feita em conjunto, então a sessão prossegue até o seu término ou interrompida por desejo do paciente.

As sessões caracterizam-se pelo diálogo entre paciente e terapeuta.

As iniciativas propostas [1, 8, 25, 26, 84] de utilização de recursos computacionais, nas sessões de terapia de reminiscência destacam que o "computador" facilita a organização, a manutenção e o acesso às informações, mas a sessão somente se realiza com o acompanhamento do terapeuta.

No entanto, a proposta é permitir a independência do paciente, para que ele seja capaz de realizar uma sessão de TR a qualquer momento e em qualquer local, sem o acompanhamento obrigatório de um terapeuta, cuidador ou familiar.

Atualmente, a evolução tecnológica dos dispositivos móveis permite vencer o problema espaço temporal uma vez que, é possível utilizá-los em todo lugar e a qualquer hora.

A maior dimensão das telas dos dispositivos móveis tem facilitado o desenvolvimento de aplicações preocupadas com a acessibilidade.

Conforme as constatações encontradas em [18, 22, 57, 68], a DA traz associada um quadro de depressão e mudança comportamental (irritabilidade, ansiedade, agitação).

Agregação de sensores (acelerômetro, giroscópio, microfone) aos dispositivos móveis e o desenvolvimento de algoritmos e artefatos de *software* possibilitam o monitoramento das pessoas em um ambiente.

Durante a realização de uma sessão de TR, os sensores integrados nos dispositivos móveis permitem monitorar e detectar a agitação e ansiedade, pela variação do tom de voz (microfone e algoritmos de reconhecimento de padrões) e captar os movimentos intensos dos membros superiores (giroscópio e acelerômetro).

Na ocorrência da mudança de comportamento detectada pela aplicação, o cuidador ou familiares podem ser avisados via mensagens e notificações, informando o estado comportamental do paciente. Músicas e sons predefinidos pelo paciente, cuidador e familiares podem ser executados para acalmar os ânimos do paciente, enquanto cuidadores ou familiares chegam para assistir o paciente.

Na busca da maior efetividade das sessões terapêuticas, *softwares* de apoio à TR têm sido desenvolvidos, atingindo resultados positivos e promissores [7]. No capítulo 4 são apresentadas aplicações de apoio à terapia de reminiscência.

3.3 Considerações Finais

Doença neurodegenerativa, a doença de Alzheimer pode surgir em pessoas de ambos os sexos e raça, independente da condição econômica, social e cultural.

Descrita por Por Alzheimer em 1906, a DA causa prejuízos à memória, linguagem, funções executivas, atenção, percepção, orientação espacial e temporal [18].

O quadro clínico de um paciente com doença de Alzheimer, na maioria das vezes traz associado um estado de depressão, irritabilidade, ansiedade, mau humor. Todos estes fatores associados conduzem o paciente a um estado de reclusão e afastamento do convívio com a sociedade.

O tratamento dos sintomas da DA pode ser realizado por terapias farmacológicas e não farmacológicas. As últimas têm conseguidos muitos adeptos, entre médicos e terapêutas, por não produzirem efeitos colaterais ao paciente e a baixa eficiência das terapias farmacológicas.

Dentre as terapias não farmacolócicas adotadas por médicos e terapeutas, a terapia de reminiscência é bastante utilizada, pois contribui para a redução da depressão, distúrbios do comportamento e os efeitos nas funções cognitivas.

A terapia de reminiscência consiste em uma estratégia psiquiátrica atuante no processo de recuperação de memórias recentes e passadas, relativas à vida do paciente [16].

As sessões convencionais de TR se apoiam em artefatos fornecidos por familiares e amigos do paciente, que ajudam a exercitar as funções cognitivas do paciente.

A popularização e versatilidade da tecnologia de informação e comunicação (TIC) permitem ferramentas computadorizadas de apoio aos pacientes com DA e contribuem para o envolvimento e engajamento nas sessões terapêuticas de reminiscência [3, 52], facilitando a condução das sessões terapêuticas de forma mais leve para o cuidador e atraente para o paciente uma vez que, o uso de um *software* proporciona a liberdade de escolha dos assuntos a serem conduzidos e sempre que necessário retomados durante uma sessão de reminiscência [8, 26].

CAPÍTULO 4

Trabalhos Relacionados

Neste capítulo são apresentados alguns exemplos de aplicações de apoio à saúde e bem-estar, para pessoas com doenças crônicas, em especial, aplicações terapêuticas voltadas à terapia de reminiscência.

4.1 Aplicativos de Apoio à Saúde e Bem-estar

Há uma diversidade de aplicações disponíveis no contexto da saúde na *Play Store* da *Google* [60] e *App Store* da *Apple* [23]. Alguns exemplos de doenças crônicas, segundo a Organização Mundial da Saúde, a DA atinge milhões de pessoas no mundo, obesidade, diabetes, câncer, cardiopatias, doença de Alzheimer.

4.1.1 Aplicativo de Controle à Obesidade MyPace

A plataforma *MyPace* [12] foi desenvolvida com a participação direta de nutricionistas e pacientes, principalmente considerando a prática de nutricionistas e a incorporação de princípios relevantes da teoria da mudança de comportamento, nas funcionalidades do *software* desenvolvidos para pacientes e nutricionistas.

Os recursos disponibilizados pelo *Mypace* são compostos por um portal desenvolvido para o nutricionista acompanhar a evolução do tratamento dos pacientes. Um aplicativo móvel relata ao paciente seu progresso no tratamento (alimentação, perda de peso etc) e o envia ao nutricionista.[46].

Os dados de progresso dos pacientes fornecem uma base de evidências, para a discussão na próxima consulta, objetivando facilitar o aconselhamento, a orientação e ação do nutricionista [12].

O *feedback* com as orientações e mensagens (predefinidas e não definidas) de incentivo do nutricionista é enviado ao paciente pelo portal na forma de notificações. O aplicativo móvel do paciente recebe as notificações com incentivos ou instruções [47].

4.1.2 Aplicativo de Controle do Diabetes *DeStress Assistant* (DeSA)

A aplicação descrita em [48] denominada *DeStress Assistant (DeSA)*, objetiva monitorar pacientes com diabetes de todas as idades em áreas remotas.

A aplicação possibilita o registro das doses diárias de glicose, o registro de atividades físicas diárias (por sensores de movimento ou pedômetro adicional), stress, peso, lembretes e relatórios médicos. Todos os dados coletados são armazenados no telefone.

A *interface* do aplicativo permite ao paciente a inserção de dados de glicose da forma manual e capturada direto do glucômetro. Os dados possibilitam ao paciente e ao médico acompanharem o histórico da taxa glicêmica, dosagem de insulina, nível nutricional diário. As informações sobre os níveis de glicemia são visualizados de forma gráfica, para facilitar o acompanhamento do tratamento.

Outra funcionalidade peculiar do aplicativo, segundo Isakovic e colaboradores [48], está na possibilidade de conectar dispositivos medidores de glicose pelo microfone do *smartphone* ou realizar as leituras de glicose com glucômeto, que se comunicam via *wifi* ou cabo com o dispositivo móvel.

4.1.3 Aplicativo de Acompanhamento do Câncer *Pain Squad*

A dor é um dos sintomas mais comuns e angustiantes relatados por adolescentes com câncer. Para motivar os pacientes a relatarem as dores sentidas, a aplicação móvel denominada *Pain Squad* [74], auxilia pacientes de 8 a 18 anos.

O aplicativo consiste em um jogo de investigação policial, no qual paciente é recrutado para realizar missões investigativas preenchendo relatórios relacionado às dores provocadas pela doença. A cada missão realizada o paciente recebe comendas de bravura e eficiência. Vídeos com atores de uma série popular da televisão canadense incentiva e parabeniza o paciente pelas realizações e prenchimento dos relatórios.

Dois protótipos, um de baixa fidelidade e um de alta fidelidade, foram desenvolvidos e testados, quanto o nível de usabilidade. A conformidade média foi de 88%, os participantes relataram diariamente, as dores sentidas com o tratamento durante o período de duas semanas [74].

4.1.4 Aplicativo Nike+ FluelBand no Monitoramento Cardíaco

Apesar da considerável evidência da eficiência dos exercícios físicos serem uma boa terapia para pacientes cardíacos congênitos estes são subutilizados [75].

Dispositivos comerciais para monitoramento de atividades físicas vêm se tornando populares, na pesquisa realizada por Stuart [75] o dispositivo *Nike+ FuelBand*

produzido pela *Nike*, consiste em um bracelete *gamificado*, que monitora a atividade física por medidas visuais (*display*) de calorias consumidas, metas alcançadas e passos por dia dos praticantes de atividades físicas.

Neste estudo, os dados medidos pelo dispositivo durante a atividade física foram transmitidos automaticamente para o *smartphone* por *Bluetooth*. A realização de exercícios prescritos é recompensada com elementos de gamificação.

Os troféus podem ser compartilhados com colegas pelas redes sociais. As recompensas representadas na forma de cartões e os demonstrativos dos resultados da atividade são enviados pelo bracelete para o *smartphone* do praticante da atividade física. Após nove meses de monitoramento, o paciente se manteve motivado demonstrando, que a gamificação tem sido benéfica [75].

Nas próximas seções são descritas aplicações construídas para apoiar terapias de reminiscência em pacientes idosos.

4.2 Sistema *Computer Interactive Reminiscence and Conversation Aid* - CIRCA

O sistema CIRCA (*Computer Interactive Reminiscence and Conversation Aid*) foi criado para dar suporte à comunicação entre pacientes com DA e seus cuidadores. Desenvolvido sobre os fundamentos da reminiscência, o CIRCA possibilita o estímulo e resgate das memórias passadas encadeando-as até o tempo presente, através de imagens, músicas sons, textos, utilizados para recontar a vida do paciente [7].

Na figura 4.1 é possível visualizar a *interface* do usuário com os temas (esportes, pessoas e eventos) e artefatos disponíveis (fotografias, vídeos e músicas) para utilização no apoio as sessões terapêuticas e botões com de controle, para reprodução dos artefatos.

As sessões terapêuticas podem ser desenvolvidas em grupo ou individualmente, no formato paciente cuidador. Astell e colaboradores [8], destacam dois motivos importantes do uso da hipermídia. O primeiro motivo é a liberdade do paciente em escolher, retomar e mover-se pelos temas, sem a exigência de uma ordem de navegação. O segundo motivo está ligado a possibilidade de vincular uma variedade de artefatos de mídia de modo dinâmico. Os artefatos podem ser combinados para produção de atividades motivantes e engajantes.

A liberdade de escolhas e retomadas aos temas de acordo com Astell e seus colaboradores [7], tira do cuidador a função da constante sustentação de conversas, permitindo um maior engajamento nas interações naturais com os pacientes com doença de Alzheimer.

Os principais pontos limitantes do CIRCA são:

- Baixa variedade de temas, não permitindo a inserção e substituição de assuntos pelo cuidador;
- Os temas são inseridos pelo desenvolvedor;
- O sistema não é adaptável às restrições cognitivas do paciente;
- Temas genéricos não personalizáveis; e
- Atividades de formato fixo;

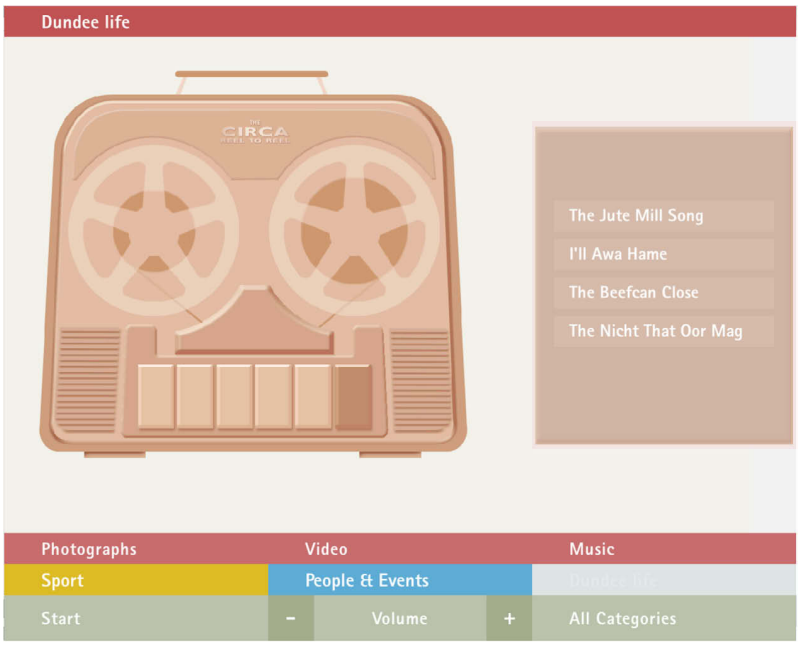

Figura 4.1: *Exemplo de Interface do CIRCA*

4.3 Vídeos Genéricos para Estímulo à Conversação

As pesquisas desenvolvidas por Davis e Shenk em [25], utilizam de tecnologias de baixo custo e baixa complexidade, na produção de vídeos pessoais e vídeos com assuntos de conhecimento público (fatos, históricos, *shows*, notícias do cotidiano), para apoiar sessões de TR, engajando pacientes e cuidadores no processo de sociabilização e comunicação.

Outro fator importante na adoção de vídeos com apelos genéricos segundo Davis e Shenk [25] são de que, vídeos genéricos geram os mais diversos comentários sobre uma ampla gama de tópicos, se comparados aos vídeos pessoais. Outra vantagem destacada é a ampliação do leque de possibilidades de diálogos desenvolvidos em sessões de TR em grupos. Apesar dos resultados preliminares promissores, na adoção de vídeos genéricos nas terapias de reminiscência, os autores apontam a necessidade de mais testes com um grupo maior e heterogêneo de pacientes. Os principais pontos de limitações na utilização de vídeos genéricos, no apoio a terapia de reminiscência são:

- A pouca interatividade com a *interface*;
- A facilidade de dispersão do paciente;
- A dificuldade na retomada dos temas;
- Atividade de formato fixo;
- O desgaste excessivo para pacientes como cuidadores; e
- Baixa variabilidade de temas em cada sessão.

4.4 Sistema *Memory Box*

O sistema *Memory Box* proposto por Davison e colaboradores [26], auxilia no tratamento da ansiedade, agitação e da depressão, sintomas psicológicos comuns em quadros de pacientes com doença de Alzheimer.

Desenvolvido com uma *interface* sensível ao toque, o *Memory Box* apresenta ícones mais intuitivos, com formato de botão. Cada um dos 4 botões da tela principal (figura 4.2) é rotulado indicando com um ícone e texto. O acesso a músicas, filmes, mensagens e fotografias, pode ser conseguido com um simples toque de botão na *interface*.

O sistema permite operação em dois níveis de interatividade. O modo simplificado permite ao paciente iniciar, parar e mudar de *stream* assistida. O nível mais complexo de interação o paciente pode selecionar uma música de uma coletânea de músicas disponíveis.

Diferente de outras propostas similares, o paciente em conjunto com seus familiares pode escolher os artefatos a inserir no sistema. A inserção pode ser realizada a distância ou localmente. E a personalização da *interface* e conteúdos pode ser feita para os níveis de restrições cognitivas do paciente [26].

As principais pontos limitantes do *Memory Box* são:

- Não é possível determinar quando o material foi assistido pelos pacientes, se no momento da inserção no sistema ou quando os pacientes estavam sozinhos em seus quartos; e
- Formato da atividade fixo;

Figura 4.2: *Exemplo de Interface do Memory Box*

4.5 Aplicação *MyBook*

A aplicação proposta por Abu e colaboradores [1], apresenta um livro digital de memórias, que combina a terapia da reminiscência (TR) e a terapia de estimulação cognitiva (TEC). A terapia de reminiscêcia é realizada através do uso de fotografias, agenda de atividades diárias. A terapia de estimulação cognitiva é aplicada com jogos de quebra-cabeça e jogos de cartas.

O *MyBook* é um livro de memórias digital personalizado, desenvolvido para trabalhar lapsos de memória. Criado para ser utilizado diariamente, para auxiliar o gerenciamento das atividades diárias da paciente.

Desenvolvido para a plataforma *Android* o aplicativo segundo Abu e colaboradores [1] possui uma *interface* simplificada e amigável.

A coleta e inserção dos artefatos e conteúdos na aplicação foram realizadas pelos desenvolvedores, com base nas preferências do paciente. A *interface* construída de modo amigável traz mensagens simples, ícones significativos, contraste de cores adequado ao usuário e comandos simplificados e diretos [1].

As atividades propostas no *MyBook* relacionadas às TR possibilitam ao usuário a construção da sua árvore familiar, na qual podem ser armazenados dados pessoais dos familiares, vídeos com depoimentos.

Na sessão galeria de fotos relacionadas as memórias da paciente é mantida, pelas fotos com atividades da família, viagens, entre outras imagens ligadas à vida do paciente. E nas atividades relacionadas à TEC é proposto o jogo da memória e quebra-cabeças. A figura 4.3 apresenta a estrutura organizacional dos componentes do aplicativo. As principais pontos limitantes do *MyBook* são:

- Impossibilidade de inserção de novos conteúdos;
- Atividades em formato fixo;
- Número de artefatos restrito em quantidade e variedade (músicas, vídeos, mensagens); e
- Sistema não adaptável às limitações progressivas da paciente.

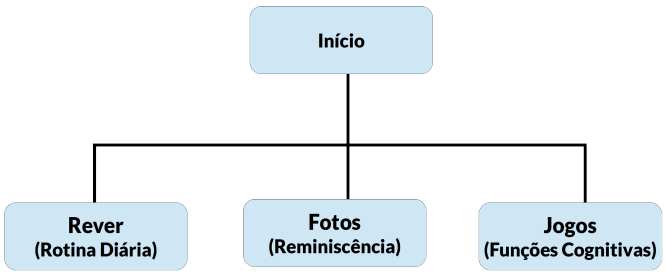

Figura 4.3: *Componentes da Aplicação MyBook*

4.6 Aplicação *ADcope*

O Adcope explora a capacidade tecnológica dos dispositivos móveis para auxiliar paciente com Alzheimer a potencializarem suas habilidades na realização de atividades da vida diária de forma independente [84].

A forma de atuação sobre a DA é focada em dois aspectos: ajudar o paciente lidar com a doença e abrandar a velocidade de declínio das habilidades cognitivas do paciente. Segundo Zmily e colaboradores [84], para que os aspectos elencados sejam atingidos, a tecnologia de comunicação (*near field communication* NFC) dos dispositivos móveis e *tags* NFC e a plataforma *Android* são utilizados.

O aplicativo é estruturado em três módulos (figura 4.4): módulo melhoria da qualidade de vida, módulo de exercícios para memória e módulos de suporte, que são apresentados a seguir.

4.6.1 Módulo Melhoria da Qualidade de Vida

Este módulo é composto por três funcionalidades [84].

1. **Carteira de memórias:** Composta de 30 imagens e frases sobre pessoas, lugares e eventos familiares. Os pacientes podem usar a carteira para melhorar suas conversação. O módulo permite ao usuário tirar fotos das pessoas, lugares e eventos familiares. Fotos de eventos e lugares podem ser etiquetadas com mensagens lembrete. As fotos das pessoas podem ser marcadas com amostras de voz da pessoa

fotografada. A imagem das pessoas é aberta automaticamente, quando a voz for detectada e reconhecida como uma amostra registrada;

2. **Calendário:** Emite notificações das atividades diárias que devem ser realizadas. Os eventos agendados podem se referir às pessoas ou lugares armazenados na carteira de memória, ajudando o paciente no reconhecimento da pessoa e/ou lugar que ela precisa ir; e
3. ***Tags* NFC:** Realiza a leitura de *tags* etiquetadas no ambiente, quando o dispositivo móvel se aproxima, exemplo a exibição do conteúdo de uma gaveta, sem necessitar abri-la.

4.6.2 Módulo Exercícios para Memória

Este módulo de acordo com Zmily e colaboradores [84], exercita a memória do paciente, para a retenção de informações por mais tempo. As funções deste módulo são descritas a seguir.

- **Treinamento da memória assistida por áudio:** O treinamento de memória assistida por áudio reproduz as informações biográficas, gravadas no módulo de *setup* e interroga o paciente sobre as informações. As respostas são direcionadas, simples e curtas, para o sucesso dos exercícios de recuperação de memória do paciente. E a simplicidade das respostas também permite o uso de técnicas de reconhecimento de voz, para validar automaticamente a correção das respostas;
- **Recuperação espaçada:** É executada uma fase de avaliação e uma fase de treinamento. Na fase de avaliação trechos de informações são apresentadas ao paciente, em um momento posterior o paciente é questionado sobre a informação apresentada. Havendo erro na resposta, o questionamento é repetido e o tempo de espera para um novo questionamento é reduzido. Este processo é repetido até o paciente recordar as informações, então o tempo de espera é registrado para iniciar questionamentos futuros. Na segunda fase com o tempo de espera entre apresentação de informação e questionamentos calibrado. O treinamento da memória é iniciado, com a apresentação das informações e o posterior questionamento. À medida que o paciente responde corretamente os questionamentos, o intervalo de tempo entre apresentação da informação, questionamento e resposta é ampliado.

4.6.3 Módulos de Suporte

O módulo de suporte descrito em [84] é composto pelo *setup* do aplicativo e a calibração do reconhecimento de voz. O módulo de configuração orienta o cuidador nas etapas de configuração, listadas a seguir.

- Gravador de áudio de dados biográfico;
- Gravador de áudio de questões e respostas biográficas;
- Recuperador de exercícios temporais;
- Inicializador da carteira de memória;
- *Tags* NFC; e
- Calendário.

As principais limitações elencadas relativas ao *ADcope*, mostradas a seguir:

- Flexibilidade de atividades; e
- Não incorpora todos os tipos de mídia.

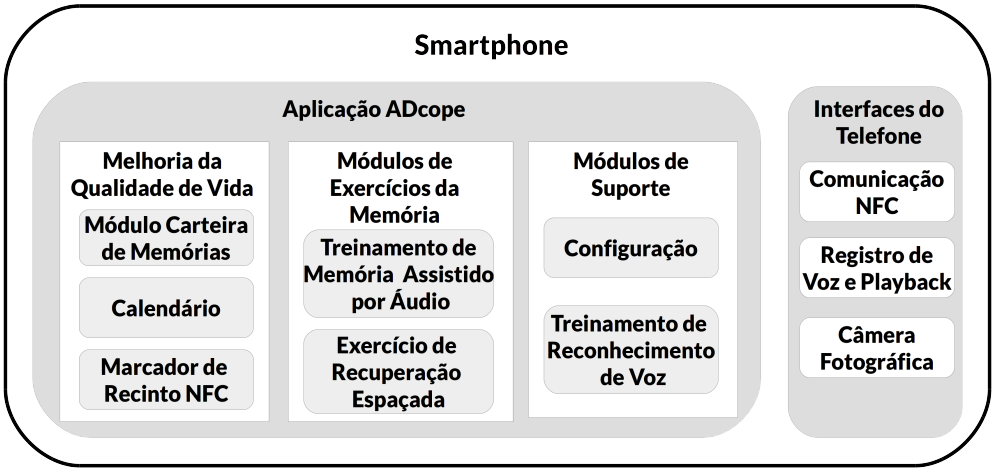

Figura 4.4: *Arquitetura da Aplicação ADcope [84].*

4.7 Comparativo entre os Trabalhos Relacionados

Ao considerar as recomendações de usabilidade e acessibilidade adaptadas de [42, 77] e as limitações das aplicações descritas na seção 2.2 e destacadas pelos próprios autores dos trabalhos relacionados, oito quesitos foram levantados e apresentados na tabela 4.1.

Para o item facilidade na retomada ou troca de temas e artefatos, todos os trabalhos possuem algum nível de retomada e troca de tema, exceto *Vídeos genéricos* que não possibilita a troca de tema durante a sessão.

A inserção de conteúdos nos trabalhos *Memory Box* são gerenciadas pelos familiares do paciente e no *ADcope* no qual os próprios pacientes fazem a gestão. Nos *Vídeos genéricos* não há o uma estrutura sistematizada para inserção dos conteúdos, porém os vídeos podem ser produzidos pelos familiares dos pacientes. E por último no

CIRCA a inserção de conteúdo no sistema é função do desenvolvedor e no *MyBook* não foi destacada pelo desenvolvedor.

Tabela 4.1: *Comparativo das Aplicações*

	CIRCA	**Memory Box**	**MyBox**	**Vídeos genéricos**	**ADcope**
Facilidade na retomada ou troca de temas e artefatos	Sim	Sim	Sim	Não	Sim
Inserção de conteúdo	Desenvolvedor	Familiar e paciente	Desenvolvedor	Não há um sistema	Paciente
Variabilidade de temas	Não	Sim	Não identificado	Sim	Sim
Customização para o tipo de restrição cognitiva	Não identificado	Sim	Não identificado	Não identificado	Não identificado
Interpretação dos ícones	Intuitiva	Intuitiva	Intuitiva	Sem ícones próprios	Sem ícones próprios
Interface de operação multissensorial	Não identificado	Não identificado	Não identificado	Não identificado	Sim
Mobilidade	Não	Não	Sim	Sim	Sim
Flexibilidade na construção das atividades	Não identificado	Não identificado	Não identificado	Não identificado	Não identificado

A variabilidade de temas é maior nos trabalhos *Memory Box*, vídeos genéricos e *ADcope*, por serem produzidos pelos familiares e/ou pacientes.

A customização do sistema para atender as restrições cognitivas é o ponto de destaque do *Memory Box*. A inserção de conteúdos pode ser feita por membros da família, pelo ícone de *prompt*, que possibilita a escolha, e o envio *online* de vídeos, fotos, sons para o dispositivo.

Em todos os trabalhos os ícones foram projetados para serem o mais intuitivo possível e facilitar o uso por parte do paciente, com exceção do *MyBook* e *Vídeos genéricos* não utilizam ícones.

No *ADcope* é considerada a possibilidade de interação multissensorial usando o toque na tela ou pelo reconhecimento de voz, ampliando a abrangência do público atendido.

A mobilidade é característica encontrada no *Mybook*, nos vídeos genéricos e *ADcope*. Esta possibilidade de movimentação do dispositivo permite aos pacientes realizem a sessão de terapia em casa ou durante viagens.

O diferencial deste trabalho está direcionado na flexibilização do desenvolvimento de atividades, construída pelo desenvolvedor, podendo ter colaboração do paciente. A possibilidade de utilização dos elementos de *gamificação*, na TR apoiada por dispositivos móveis, auxiliando na adesão ao tratamento, na redução dos sintomas provocados pela doença de Alzheimer sejam mais duradouras.

4.8 Considerações Finais

O capítulo apresenta exemplos de aplicativos móveis desenvolvidos, para a promoção do bem-estar, acompanhamento de tratamentos de doenças crônicas e aplicações de apoio à terapia de reminiscência tutoradas por computador.

O desejo das pessoas em mudar seus hábitos e comportamento relativos à saúde, na busca do bem-estar e acompanhamento da condição de saúde.

Na busca pelo bem-estar, a prática de exercícios físicos vem ganhando adeptos, pois pelos aplicativos móveis (e.g. *exergames*), as pessoas criam rotinas de exercícios físicos e acompanham sua evolução física a cada treinamento.

Outra vertente importante apoiada pelas aplicações móveis está relacionada ao acompanhamento do tratamento de doenças crônicas (obesidade, hipertensão arterial, câncer, doença de Alzheimer entre outras).

Esses aplicativos permitem aos profissionais de saúde acompanhar a adesão dos pacientes ao tratamento e às condições de saúde destas pessoas. Exemplos como o *Pain Squad* [74], para acompanhar crianças e adolescentes no relato das dores provocadas pelo tratamento do câncer, o *MyPace* [12], para o acompanhamento e mudança de hábitos alimentares no combate à obesidade, o *DeStress* [48] no controle da diabetes, com o monitoramento da glicemia, dosagens de insulina e nutrição.

Alternativas comerciais, como o *Nike+ FlueBand* [75] aplicados para acompanhar a atividade física de pacientes cardíacos. O dispositivo permite ao paciente e médico monitorar o consumo de calorias, caminhada etc.

O desenvolvimento das TICs adequadas à pessoa idosa constitui uma solução potencial de implementação. A terapia de reminiscência apoiada por computador segundo Astell e colaboradores [7, 8], contribui para uma sessão de TR mais eficaz e atraente para pacientes e terapeutas.

As sessões se tornam mais dinâmicas e fáceis de conduzir, o paciente tem maior liberdade e facilidade ao escolher o tema que pretende relembrar, aliviando a carga do terapeuta.

O sistema CIRCA realiza a TR de forma estruturada aproximando paciente e terapeuta, permitindo uma interação e comunicação mais efetiva [7, 8]. O material é organizado por temas e tipos de mídia (áudio, vídeo texto, imagens), o uso da hipermídia deixa o paciente livre para navegar pelos temas como desejar. A combinação dos artefatos, permite ao terapeuta a geração de sessões com atividades motivantes.

A opção de vídeos genéricos é utilizada para a reminiscência não estruturada, com vídeos genéricos e/ou pessoais produzidos por cuidadores ou familiares com imagens e recorte de outros vídeos. Esse tipo de TR dificulta a retomada de temas e a independência do usuário em navegar livremente pelos artefatos, a variabilidade de artefatos reduzidos prejudica o progresso da sessão de reminiscência.

O *MemoryBox*, é um sistema que permite a TR estruturada, fazendo alusão a uma caixa de memória convencinal. A *interface* do *MemoryBox* caracteriza-se pela simplicidade e significância. Os quatro botões de acesso aos artefatos presentes na *interface* são intuitivos [26].

O *MyBook* é um aplicativo móvel desenvolvido, para auxiliar a memória do paciente. O sistema apoia a terapia de reminiscência e a terapia de estimulação cognitiva. As atividades de agenda das atividades diárias e fotos sobre familiares apoiam a TR, e as atividades quebra cabeça e jogos com cartas apoiam a TEC.

O *ADCope* é um sistema multissensorial para plataforma móvel, que permite o paciente recuperar suas memórias(carteira de memórias) pelo *display* ou pelo reconhecimento da voz. O sistema utiliza imagens de locais e pessoas, para melhoria da conversação. O calendário ajuda o paciente a lembrar das atividades diárias, associados com pessoas ou lugares mantidos na carteira de memória do usuário. A funcionalidade *tags* NFC ajuda o paciente com DA lembrar o conteúdo de gavetas e armários, sem necessitar abri-los, os *QR codes* fixados pelo ambiente são lidos e as informações associadas mostradas no display do *smartphone*.

CAPÍTULO 5

Requisitos de *Gamificação* para Aplicações Voltadas para Pacientes com Doença de Alzheimer

Neste capítulo são descritos os requisitos a serem considerados no desenvolvimento da arquitetura de *gamificação* de aplicações voltadas a pacientes com doença de Alzheimer, tecnologias utilizadas e método de coleta de dados.

5.1 Materiais

Os recursos tecnológicos utilizados no desenvolvimento dos componentes de *gamificação* para aplicações terapêuticas *gamificadas* e o aplicativo exemplo foram:

- **Plataforma de desenvolvimento:** Foi utilizada a plataforma *Android Studio*, por ser uma ferramenta de código aberto, com todo o suporte para programação, compilação e testes de *interface* em diferentes tipos e modelos de dispositivos móveis, tanto virtuais (uso do emulador integrado), como em dispositivos reais;
- **Linguagem Java:** Linguagem de programação orientada a objetos, foi escolhida por ser nativa da plataforma de desenvolvimento *Android Studio* e compatível com o desenvolvimento de aplicações móveis;
- **Sistema Operacional *Android* 4.4 *KitKat*:** Foi escolhido por ser uma arquitetura aberta. A versão *KitKat* segundo dados da *Google*, é compatível com aproximadamente 80% dos dispositivos móveis ativos que utilizam *Android*;
- **Ferramenta *Astah Community*:** Foi utilizada a ferramenta *Astah Community* versão estudantil, para modelagem dos diagramas utilizados no projeto;
- **Banco de Dados *SQLite* versão 3.0:** Foi utilizado o banco de dados *SQLite*, para o desenvolvimento da base de dados que persiste dados sobre o paciente e sua desenvoltura na utilização da aplicação exemplo. A preferência por este banco de dados se justifica por ser compacto, livre e recomendado pelos desenvolvedores *Android*;

- **Dispositivo móvel *Samsung Galaxy S5 mini*:** O dispositivo escolhido possui suporte para a versão 4.4 *KitKat* e superiores fornecido pelo fabricante, capacidade de memória interna e externa compatível com as necessidades; e
- **Dispositivo móvel LG G Pad 8.0:** O dispositivo possui suporte para a versão de partida do sistema operacional e superiores fornecidas pelo fabricante, *display* de 8 polegadas, que possibilita boa visualização e manuseio do dispositivo.

5.2 Métodos

A fase de levantamento dos dados envolveu a pesquisa de material bibliográfico, levantamento dos requisitos funcionais, para o desenvolvimento da arquitetura e aplicação exemplo.

5.2.1 Elementos de *Gamificação*

Foram realizadas pesquisas nas bases de periódicos da IEEE, ACM, Springer, ScienceDirect, SciELO, PubMed e Google Scholar na busca de trabalhos que apresentassem dados e informações sobre a utilização dos elementos de *gamificação*, como facilitador da mudança comportamental ligadas a saúde.

A primeira *string* de busca foi elaborada no idioma inglês *Gamification AND (health OR healthcare)*. Foi detectada a existência de trabalhos publicados relacionados ao desenvolvimento de aplicações criadas para manutenção da forma física (*fitness*), monitoramento da saúde (pressão arterial, obesidade, diabetes, entre outas), para crianças, jovens adultos, e idosos. Uma segunda busca nas mesmas bases de periódicos foi realizada com uma *string* mais restrita, para refinar e encontrar trabalhos sobre a *gamificação* de aplicações criadas para idosos com doença de Alzheimer. A *string* de busca elaborada no idioma inglês *(Serious game OR Gamification) AND (elderly OR older people) AND Alzheimer*.

5.2.2 Doença de Alzheimer

A busca na literatura médica (livros base de periódicos PubMed) para coleta de dados e informações sobre a doença de Alzheimer e determinação de personas (apêndice D), que representem os possíveis pacientes com DA apoiados pelas aplicações *gamificadas* com finalidades terapêuticas.

5.2.3 Restrições Naturais da Idade

O levantamento de relatos de terapeutas ocupacionais em vídeo sobre as limitações impostas pela idade e a observação de idosos na família, orientou a procura por dados relativos sobre a perda das habilidades motoras, cognitivas e sensoriais nos idosos. Com estes dados foram compilados, gerando uma lista de restrições imposta aos idosos pelo processo de envelhecimento e apresentados no apêndice E.

5.3 Definição do Elementos de *Gamificação*

Após identificar as limitações naturais impostas pela idade e pela doença de Alzheimer, mais precisamente nas funções cognitivas dos idosos, foram identificados 17 componentes de *gamificação* que podem contribuir com a melhora dos resultados das terapias não farmacológicas, para pacientes com doença de Alzheimer nas fases inicial e moderada.

Em especial, a utilização dos componentes de *gamificação* em aplicações móveis desenvolvidas para apoiar sessões terapêuticas de reminiscência (seção 3.2). Para a determinação dos componentes de *gamificação* foram realizados os seguintes passos:

- **Passo 01:** Buscar o conceito de *gamificação*, a definição dos elementos de *gamificação* propostos na literatura (Kapp, Werbach e Hunter)(capítulo 2);
- **Passo 02:** Busca na bibliografia médica, sobre os efeitos, causas da DA, exames clínicos de detecção da doença, formas de tratamento terapêuticos (capítulo 3); e
- **Passo 03:** Identificar entre as doenças consideradas crônicas, como a *gamificação* tem contribuído para a mudança de comportamento relacionado à saúde e bem-estar, com objetivo de facilitação da adesão ao tratamento e redução dos sintonas de doenças neurodegenerativas como a doença de Alzheimer [24, 64].

A relação de elementos de *gamificação* identificados, como componentes com potencial de contribuir na melhora dos sintomas cognitivos causados pela DA, motivação e adesão as terapias. A tabela 6.1 traz a relação dos componentes de *gamificação* aplicáveis na construção de aplicações terapêuticas e a tabela 6.2 estabelece a relação dos elementos de *gamificação* com o estímulo das funções cognitivas, estímulo as relações sociais e estímulo emocional dos pacientes com doença de Alzheimer.

5.4 Perfis dos Usuários

No Brasil a Lei nº 10.741/2003 cria o Estatuto do Idoso, em seu artigo primeiro considera idoso o cidadão brasileiro com idade igual ou superior a sessenta anos. A

manifestação da doença de Alzheimer, quando ocorre, geralmente atinge idosos com idade acima dos 65 anos em ambos os sexos [9, 18].

Outro usuário importante é a figura do cuidador. O cuidador pode ser representado por um familiar, enfermeiro ou terapeuta, que acompanha o paciente durante a sessão terapêutica.

5.5 Limitações do Usuário Idoso

As limitações funcionais presentes no usuário idoso podem surgir ocasionadas pelo envelhecimento natural, a existência de doenças crônicas não transmissíveis causadoras de mudanças fisiológicas agudas, como a doença de Alzheimer, acidente vascular cerebral, fraturas etc.

Denominam-se limitações funcionais as restrições na realização de ações físicas e mentais fundamentais para a vida diária, quando comparadas as funções funcionais de pessoas do mesmo sexo e faixa etária [65].

No paciente idoso diagnosticado com doença de Alzheimer, além da doença as limitações funcionais podem ser agravadas, quando associadas a problemas de baixa visão, audição e restrições motoras.

O processo de perda da visão segundo Macedo e colaboradores [59], tem início na faixa etária dos 40 a 50 anos. A redução gradual e irreversível da visão diminui a capacidade de acomodar ou de focalizar objetos próximos.

A perda da acuidade visual é um fator que dificulta o uso de dispositivos e *interfaces*. Os relatos de idosos jogadores colhidos por Cota e Ishitani [21] confirmam as dificuldades de utilização de dispositivos e *interfaces*. *"A interface deve evitar pequenos elementos, controles de tela perto um do outro e muita informação em torno da tela são inadequados para os idosos"* [21].

A audição é outro sentido bastante afetado com o processo de envelhecimento, tornando-se socialmente incomodo a partir dos 50 anos [78]. O grau de perda auditiva pode variar de leve a profunda. Segundo Etcheverria e colaboradores [34] esta perda dificulta a percepção da comunicação falada e dos sons do ambiente.

Veras [78] constatou que na comunicação falada, o uso de frases claras inseridas no contexto são mais inteligíveis que palavras isoladas. O contexto da frase impõe uma redução das alternativas possíveis de semelhanças, entre os sons das palavras ampliando a compreensão.

Diante das limitações funcionais e as restrições cognitivas impostas pela DA, o projeto de *interfaces* acessíveis e de fácil utilização contribui para a adesão dos idosos utilizarem aplicativos no apoio as atividades diárias, conforme os resultados apresentados nas pesquisas [21, 42].

5.6 Elemento Mínimos de *Gamificação* para Aplicações Voltadas para Doença de Alzheimer

Nesta seção são apresentados os três níveis da hierarquia dos elementos de jogos (seção 2.1) relevantes para a *gamificação* com enfoque em aplicações terapêuticas elaboradas para pacientes com DA. Os elementos mínimos de *gamificação* implementados nesta dissertação são propostos por Berndt em [14].

5.6.1 Dinâmicas

A dinâmica representa os elementos de jogos com nível de abstração mais elevado presentes no processo de *gamificação*. Seus elementos são:

Narrativa

Forma estruturada de representar estórias sobre experiências personalizadas de modo atraente e dinâmico [45]. Neste trabalho são adotadas as duas formas mais comuns de utilizar narrativas em jogos, as narrativas embutidas e as emergentes. A narrativa embutida representa o roteiro do jogo e não pode ser alterada pelo jogador. A narrativa emergente é criada pelo jogador a medida que há a interação com o jogo [81].

A narrativa deve estar voltada para o resgate das atividades de interesse do paciente, utilizando elementos relacionados à sua vida, sempre com o objetivo de despertar boas emoções, lembranças prazerosas, enquanto realiza a atividade. Ao ser utilizada em projetos de aplicações *gamificadas*, quando bem estruturadas as narrativas podem auxiliar o paciente a relembrar fatos ocorridos e atividades realizadas recentemente, além de apoiar a estruturação verbal do discurso mediante o resgate de palavras, muitas vezes esquecidas ou aplicadas de forma não correta no contexto de um diálogo.

Consequentemente, o estímulo a iniciativa é possibilitado pelas narrativas, a medida que o paciente pode ser instigado a realizar as atividades programadas, respeitando o tempo para executá-las.

Regras

Contribuem para orientar e conduzir a realização das atividades e evitar a tentativa do usuário ludibriar o sistema, na busca de recompensas e pontos, enfraquecendo a efetividade dos benefícios da aplicação. As regras norteiam a realização de uma atividade, para que exista um fluxo de começo, meio e fim, devendo ser expostas de modo simples e objetivo. Nas aplicações é importante que o paciente seja sempre supervisionado por familiares e/ou cuidadores. Aplicações desenvolvidas para esses pacientes podem permitir a interação de um ou mais usuários (e.g. paciente e cuidador).

O início da aplicação, preferencialmente, deve ser realizado pelo paciente, podendo haver sugestão do cuidador. Uma atividade deve ser maleável, sem limites máximos de duração, sempre levando em conta a disponibilidade do paciente.

Relacionamento

Ocorre entre pessoas, entre pessoas e personagem virtual (avatar), ou entre ambas durante a execução de uma atividade. Ao se relacionar, seja virtualmente ou presencialmente, o paciente pode demonstrar seus feitos e progressos aos pares, criando um sentimento de independência e utilidade. Tal demonstração eleva a motivação, iniciativa, desperta o interesse pelas atividades e contribui diretamente com a redução da ansiedade e agitação.

Neste trabalho o relacionamento corresponde ao contato físico e direto com o cuidador (terapeuta, familiar) ou via redes sociais, com grupo de amigos, familiares e outros pacientes.

Progressão

Demonstra o nível de eficiência atingido pelo paciente durante a execução de uma atividade e o quanto mais pode ser melhorado. Ao progredir e demonstrar esse progresso aos pares o paciente eleva a confiança e autoestima.

A progressão tutorada (níveis e conquistas) auxilia a orientação no espaço e no tempo, possibilitando ao paciente e cuidador uma visão evolutiva da trajetória, com ponto de partida, posicionamento atual e objetivo a ser conquistado. Duas visões de progresso podem ser mantidas e reportadas, uma para o paciente admitindo oscilações e outra real para o cuidador com a evolução da doença.

Emoção

Despertar o prazer, a felicidade em realizar a atividade. O uso de elementos (imagens, sons, textos) que gerem boas emoções contribui diretamente para a redução da agitação do paciente, e o despertar do interesse pelas atividades a realizar. O sentimento de felicidade torna o paciente mais sociável, com o desejo de partilhar suas emoções e conquistas com seus pares [10].

Os níveis de emoção a serem despertados nos pacientes devem ser comedidos. O aconselhável é procurar orientação médica para definição dos níveis de emoção a despertar.

5.6.2 Mecânicas

A mecânica agrupa os elementos de jogos em um nível de abstração intermediário, que corresponde os processos básicos que geram as ações de engajamento do usuário.

São elementos dessa categoria:

Desafios

Estão ligados às habilidades preservadas do paciente, pois desafios inalcançáveis somente irão trazer frustrações e recusas na realização da atividade. Ao utilizar desafios de forma ponderada, as atividades realizadas tornam-se mais atraentes e motivadoras, despertando o interesse do paciente em executá-las.

Ao vencer os desafios impostos e progredir, desperta-se o desejo de partilhar as conquistas alcançadas, gerando um sentimento de prazer e realização, que reduz a agitação e a depressão do paciente. Os desafios estão fortemente relacionados com o objetivo terapêutico definido pelo terapeuta e/ou cuidador.

Competição

O conceito de *handcap* ou *handicapping* contribui na redução da disparidade entre competidores. Segundo Rocha e colaboradores [69], *handcap* é o recurso usado para auxiliar competidores menos experientes competirem com competidores mais experientes.

O objetivo do *handicap* é proporcionar aos competidores ainda inexperientes chances de vitória. Nas situações de competição disputas desequilibradas tendem a ser desmotivantes, tanto nas situações de derrota ou vitória.

O ato de competir desperta o interesse e motiva o paciente a realizar suas atividades de forma mais precisa e como consequência amplia sua concentração, organização (espacial e temporal) e recuperação de fatos e eventos ocorridos.

Cooperação

É considerado o elemento motivador ao contribuir com um grupo na realização das atividades o paciente sente-se importante e necessário. Esse sentimento auxilia na motivação e no interesse em cumprir com maior qualidade as atividades, refletindo diretamente na sua atenção, orientação e recuperação de informações.

O papel do cuidador é importante na orientação do paciente frente a dificuldades de execução de uma atividade. As redes sociais constituem outro modo de contribuir na realização de uma atividade uma vez que, dicas de realização de atividades podem ser partilhadas por todos os membros da rede social.

Recompensas

Ao ser recompensado pelas suas ações, o paciente torna-se mais motivado, interessado e feliz em realizar suas atividades. O sentimento de felicidade reduz a agitação e a depressão do paciente, além de torná-lo mais sociável.

As recompensas precisam ter significado para produzir resultados efetivos ao paciente. A ocorrência de uma recompensa pode acontecer após a conclusão parcial ou não de uma atividade, ou de tempos em tempos de modo aleatório.

Feedback

As respostas sempre positivas, incentivando as realizações do paciente deve possuir uma linguagem simples com termos familiares ao paciente. Ao informar periodicamente os resultados das ações, o *feedback* permite orientar a tomada de decisões, a noção de tempo e espaço do paciente.

As respostas ao paciente ocorrem no momento que as atividades são realizadas e quando detectado uma demora na realização da atividade. As mensagens devem ser de incentivo e com um tom que destaque os feitos do paciente.

5.6.3 Componentes

Os componentes agrupam os elementos de jogos vistos com explicitude nas aplicações *gamificadas*. Esses elementos encontram-se na base da pirâmide de abstração. Seus elementos incluem:

Avatar

O uso de avatares representando o paciente, colaboradores, competidores facilita a interação do usuário com o ambiente digital da aplicação, o estabelecimento de relações sociais, o agrupamento e recuperação de informações sobre as pessoas.

A representação em terceira pessoa no ambiente digital auxilia a orientação do paciente em relação ao ambiente e objetos virtuais.

Insígnias

São símbolos representativos dos objetivos alcançados pelo paciente ao realizar com sucesso uma atividade.

A atribuição de representação simbólica das conquistas pode ser utilizada como elemento motivador, ampliador dos interesses dos pacientes, em continuarem a realizar suas atividades, além da possibilidade de socialização entre os membros de um grupo social.

A obtenção de uma insígnia ocorre quando um objetivo é completado, mesmo que parcialmente. As insígnias podem corresponder a símbolos, mensagens de carinho e afeto gravadas por familiares, amigos do paciente, sendo renovadas a cada realização da atividade.

Conquistas

São importantes para registrar a evolução do paciente na realização de suas atividades. Fornece subsídios para um *feedback* de incentivo, mesmo com realizações parciais. As conquistas podem auxiliar na motivação e interesse em realizar as atividades.

As realizações do paciente são registradas pela aplicação e poderão ser enviadas para o grupo social (familiares, amigos) do paciente.

Gráfico Social

Representam a participação do paciente em um grupo de usuários. Os gráficos sociais colaboram com o sentimento de independência, importância e utilidade, ajudando a combater o quadro depressivo dos pacientes.

A representação gráfica dos relacionamentos traz o paciente como o elemento central da rede, com todas as ramificações partindo dele. A comunicação com qualquer um dos membros da rede pode ser feita de modo síncrono ou assíncrono.

Níveis

Aumento do grau de dificuldade das atividades de modo não linear, ou seja, nem sempre crescente de forma a respeitar o tempo de execução e os limites do paciente.

Os níveis são importantes para indicar a evolução do paciente na realização de suas atividades e como forma de demonstrar seus feitos e capacidade de realização.

A medida que o paciente conclui uma atividade predefinida, dependendo o grau de acerto, o próximo nível pode exigir mais habilidades do paciente ao cumprir uma atividade. Na existência de um alto grau de dificuldade na realização da atividade proposta, as dificuldades devem ser reduzidas.

Pontos

Possibilita quantificar (percentual, absoluto) o quanto já foi realizado pelo paciente, e o que falta para finalizar a atividade. Sabendo o quanto falta para atingir fim de uma atividade, o paciente pode se motivar a continuar a realizá-la.

Porém, o desejo de conseguir os pontos para atingir o fim de uma atividade pode levar o usuário a tentar burlar o sistema da aplicação e não cumprir todos os objetivos da atividade.

A medida que o paciente completa as atividades propostas, os pontos conseguidos podem liberar atividades predefinidas, novas e diferentes.

Missões

Representam as atividades que devem ser realizadas pelos pacientes. As missões podem auxiliar na inciativa do paciente, uma vez pertencente a um grupo, é sua responsabilidade atuar para a evolução do todo e demonstrar a capacidade de realização.

As missões devem ser maleáveis para se adaptarem ao estado emocional do paciente, uma vez que, a atividade pode funcionar no primeiro momento. Em um segundo instante pode não ser tão interessante realizá-la.

A importância das missões está em preservar e potencializar as habilidades físicas e cognitivas restantes no paciente. O cuidador deve conduzir o paciente a realizar as missões (atividades) do modo mais instigante e prazeroso possível.

A escolha dos componentes de *gamificação* utilizados na construção de uma atividade está relacionada as funções cognitivas a serem trabalhadas na a atividade. A tabela 6.1 lista os 17 componentes de *gamificação* propostos e a justificativa do uso em uma atividade.

5.7 Elementos de *Gamificação* na Estimulação de Pacientes com Doença de Alzheimer

Nas três frentes de pesquisa ligadas ao combate ao Alzeimer (inibição do surgimento da doença, a cura dos pacientes e alívio dos sintomas), a *gamificação* está enquadrada, como recurso tecnológico de suporte a tratamentos não medicamentosos de combate aos sintomas da doença de Alzheimer.

Atividades terapêuticas computadorizadas *gamificadas* podem estimular as funções cognitivas (estímulo cognitivo), o relacionamento social (estímulo social) e as boas emoções (estímulo emocional), propiciando momentos de bem-estar aos pacientes e cuidadores.

A tabela 6.2 mostra a relação entre os elementos de *gamificação* propostos e os estímulos cognitivos, sociais e emocionais.

O trio de componentes *avatar*, *feedback* e narrativas atuam em colaborar com a linguagem do paciente, no momento que este pode ler e ouvir as mensagens de apoio e incentivo retornadas pelas atividades.

As funções executivas são trabalhadas no momento da realização de atividades exigindo planejamento, tomada de decisões no cumprimento das missões sem ferir as regras. Ao superar os níveis dos desafios propostos na atividade, o paciente exercita a sua

capacidade de organização, planejamento, avaliação das soluções abstraídas e a decisão crítica sobre as alternativas de realização das atividades.

O competir e o cooperar durante as missões estimulam o foco, com o objetivo de vencer individualmente ou auxiliar o grupo a conquistar os objetivos definidos em uma atividade. O ato de competir e cooperar exige o resgate das experiências do paciente, ao planejar, julgar e decidir ações, que devem ser tomadas no cumprimento de uma atividade.

O reconhecimento da capacidade de realização e das conquistas (insígnias, conquistas, recompensas), pelo grupo social gera a sensação de inclusão, o sentimento de alegria e satisfação no paciente em sempre progredir.

5.8 Considerações Finais

No Brasil, o cidadão idoso, conforme o Estatuto do Idoso, são pessoas com idade igual ou superior a 60 anos. O envelhecimento naturalmente reduz as habilidades físicas, cognitivas e sensoriais das pessoas.

No paciente idoso diagnosticado com doença de Alzheimer, além da doença as limitações funcionais podem ser agravadas, quando associadas a problemas de baixa visão, audição e restrições motoras.

Os elementos de *gamificação* descritos englobam os três níveis da hierarquia dos elementos de jogos dinâmica, mecânica e componentes. Os 17 elementos são listados a seguir:

- Narrativa Embarcadas: Orientam globalmente o usuário sobre o funcionamento da atividade (exemplo: "...nesta atividade tu deves escolher a palavra que se encaixa nas frases...");
- Narrativa Emergente: Orientar ações locais relativas a atividade (exemplo: selecione uma das alternativas);
- *Feedback*: Incentivo ao usuário em momentos de desmotivação e erros na atividade (exemplo: Falta pouco, vamos continuar tentando), nos momentos de êxito parabenizar pelas conquistas (e.g Parabéns!! Tu fostes muito bem);
- *Avatar*: Representação virtual de uma pessoa na atividade (exemplo: instrutor);
- Regras: Definem formas de evitar tentativas de burlar o sistema, pontuação a ser atribuída na conclusão de uma atividade, e determinam o que deve ser conquistado para mudança de nível etc;
- Relacionamento: Orientam a forma de interação na atividade (avatar-paciente, paciente-pessoa externa etc);
- Progressão: Representa duas visões evolutivas paciente, as conquistas pela realização das ações na atividade (pontos, insígnias etc), evolução da doença representada

pelo mini exame de estado mental disponibilizado para profissionais de saúde e familiares;
- Emoção: Objetiva remeter o paciente a lembraças, que gerem boas emoções;
- Desafio: Representa a superação de barreiras para realização da atividade (exemplo: arrumar a mesa do café da manhã);
- Competição: Orienta a disputa entre o paciente e competidores reais ou virtuais;
- Cooperação: Permite a parceria do paciente em colaborar com outras pessoas para a realização de atividades;
- Recompensas: Define os elementos para premiar o esforço e a capacidade do paciente (exemplo: mensagens reconhecendo a feito, imagens etc);
- Insígnias: Represnta simbolicamente a realização do usuário. é importante que as insígnias tenham sentido para o paciente;
- Conquistas: Representa as premiações recebidas durante a realização da atividade;
- Gráfico Social: Representa todas as ligações de amizade do paciente;
- Níveis: Define o grau de dificuldade para a realização das atividades;
- Pontos: Quantificação das realizações do paciente (exemplo: receber condecoração, influenciar os níveis de dificuldades); e
- Missões: Define as metas a serem alcançadas com o cumprimento da atividade.

Atividades terapêuticas apoiadas por *software gamificados* podem estimular as funções cognitivas (estímulo cognitivo), o relacionamento social (estímulo social) e as boas emoções (estímulo emocional), propiciando momentos de bem-estar aos pacientes e cuidadores.

Os estímulos cognitivos provocados pelos componentes (*feedback, avatar e narrativas*) influenciam na manutenção da linguagem do paciente.

As atividades terapêuticas *gamificadas* com o objetivo de estimular sociabilização dos pacientes com DA, podem conter na sua construção elementos para proporcionar a competição, cooperação e desafios. O ato de competir, cooperar exige dos pacientes a concentração (atenção), memorização (memória), planejamento, organização (funções executivas).

A satisfação e a alegria são sentimentos que acalmam, melhoram o humor, aliviam a depressão. Os componentes de *gamificação* (Narrativas, Recompensas, Insígnias, Conquistas, Progressão) podem estimular boas emoções, ao serem utilizados, como mecanismos de reconhecimento e elogios, as realizações e esforços do paciente.

CAPÍTULO 6

Proposta de Arquitetura de *Gamificação* para Aplicações Voltadas para Pacientes com Alzheimer

Neste capítulo é descrita a arquitetura de *gamificação* proposta para a construção de aplicações terapêuticas, para pacientes com doença de Alzheimer nos estágios leve a moderado.

6.1 Arquitetura dos Componentes de *Gamificação*

A organização dos componentes de *gamificação* foi estruturada em quatro módulos.

- **Módulo Comunicação:** Composto por componentes que realizam a comunicação das aplicações, sempre que uma interação do paciente acontece no ambiente da aplicação;
- **Módulo Social:** É formado por componentes, que permitem a troca de experiência com outros usuários;
- **Módulo Regulação:** Contempla os componentes que estabelecem as regras de funcionamento das aplicações e flexibilizam ou endurecem as dificuldades de realização das atividades, mediante a variação das habilidades do paciente; e
- **Módulo Objetivos:** Composto por componentes que monitoram, registram e têm a função de motivar pacientes na realização das atividades terapêuticas propostas em uma aplicação.

Os módulos foram modelados fundamentados no paradigma de programação orientado objetos e padrões de projeto *Abstract Factory*.Além dos componentes integrantes da arquitetura de *gamificação*, foram modelados componentes (habilidades, ações) auxiliares, para gerenciar e registrar a interação do paciente com as aplicações.

O modelo de dados relacional foi utilizados para modelar e projetar a base de dados das aplicações. Os dados correspondem às habilidades cognitivas, sensoriais

e físicas do paciente, que determina a flexibilização ou endurecimentos dos níveis de dificuldades impostos, para a realização das atividades previstas na aplicação.

Os dados foram coletados, baseados na observação do comportamento dos idosos, entrevista com neurologista e revisão da literatura.

Tabela 6.1: *Componentes de gamificação utilizados nas construção das atividades*

Componente	**Justificativa**
Avatar	Interação visual e audível com o paciente.
Feedback	Seleção das narrativas, conforme resultado das ações realizadas pelo paciente junto a aplicação.
Narrativas	Conjunto de narrativas embutidas e emergentes a serem utilizadas pela aplicação.
Insígnias	A medida que o paciente realiza as ações de forma correta este recebe uma condecoração pelas suas realizações.
Recompensas	Utilizada como forma de estímulo as recompensa são oferecidas para incentivar o paciente a persistir no momento de dificuldade na realização das ações previstas na atividade.
Regras	Controla as ações que o paciente pode realizar durante a execução da atividades (ex: repetir a ação, desistir da ação) e a atribuição da pontuação resultante das ações realizadas pelo paciente (ex: pontos em caso de acerto na primeira tentativa, pontos nas próximas tentativas, recompensas).
Pontos	Utilizados para informar ao paciente, o quanto foi conquistado com a realização das ações da atividade e servir como parâmetro na atribuição de insígnias aos feitos do paciente.
Progressão	Comparativo entre os resultados obtidos na realização da mesma atividade em outros momentos terapêuticos. Para comparação do progresso do paciente na realização da atividade será utilizado o MEEM, que pode ser acompanhado pelo cuidador.
Níveis	Controle do tempo de resposta em que o paciente necessita para realizar as ações da atividade. Este tempo de resposta pode variar, conforme os níveis de habilidade do paciente se alteram.
Relacionamentos	Identificar as pessoas que o paciente mantêm mais relações, sejam relações de cooperação ou competição.
Conquistas	Representar visualmente as recompensas previstas na atividade.
Cooperação	Estimular o paciente a interagir com outras pessoas, através da colaboração na realização das atividades.
Competição	Manter o equilíbrio de condições e habilidade entre os participantes de uma atividade.
Desafios	Isntigar o paciente a realizar as ações propostas em uma atividade da forma mais precisa e eficiente.
Missões	Orientar os pacientes sobre os objetivos a alcançar, a ordem e as ações que devem ser realizadas, como encerrar uma atividade.
Gráfico Social	Mostrar para o pacientes a sua rede de relacionamento com outras pessoas.

Tabela 6.2: *Funções Estimuladas pela gamificação*

Componentes de *Gamificação*	Estímulo Cognitivo	Estímulo Social	Estímulo Emocional
Avatar	Linguagem		
Feedback	Linguagem		
Narrativa	Linguagem		Alegria, Satisfação
Pontos	Atenção		
Níveis	Funções Executivvas		
Recompensas			Alegria, Satisfação
Cooperação	Atenção, Memória, Funções Executivas	Utilidade, capacidade	
Competição	Atenção, Memória, Funções Executivas	Capacidade	
Missões	Atenção, Funções Executivas		
Gráfico Social		Inclusão	
Insígnias			Alegria, Satisfação
Conquistas			Alegria, Satisfação
Emoções	Memória		
Regras	Funções Executivas		
Relacionamento		Inclusão	
Progressão			Alegria, Satisfação
Desafios	Atenção, Memória, Funções Executivas		

6.2 Arquitetura de *Gamificação*

A arquitetura de *gamificação* objetiva facilitar o processo de desenvolvimento de aplicações *gamificadas* para apoiar as sessões terapêuticas de pacientes diagnosticados com doença de Alzheimer. Na figura 6.1 são mostrados todos os artefatos de *software*, arquivos de suporte e atores envolvidos no desenvolvimento de aplicações *gamificadas* para suporte às sessões terapêuticas de pacientes com Alzheimer.

O desenvolvimento de aplicações terapêuticas é realizado pelo ator desenvolvedor, por meio da colaboração do ator cuidador (familiar ou profissional de saúde). A colaboração do cuidador consiste no fornecimento de dados relativos as condições físicas e cognitivas do paciente, artefatos de entrada relacionados a história do paciente (música, imagens, vídeos, textos, preferências) e objetivo terapêutico da aplicação a ser desenvol-

vida.

As informações repassadas ao desenvolvedor pelo cuidador são inseridas na Arquitetura de *gamificação*, por meio de uma ferramenta de autoria, gerando como saída um arquivo XML (*Extensible Markup Language*) correspondente a *atividade xml*.

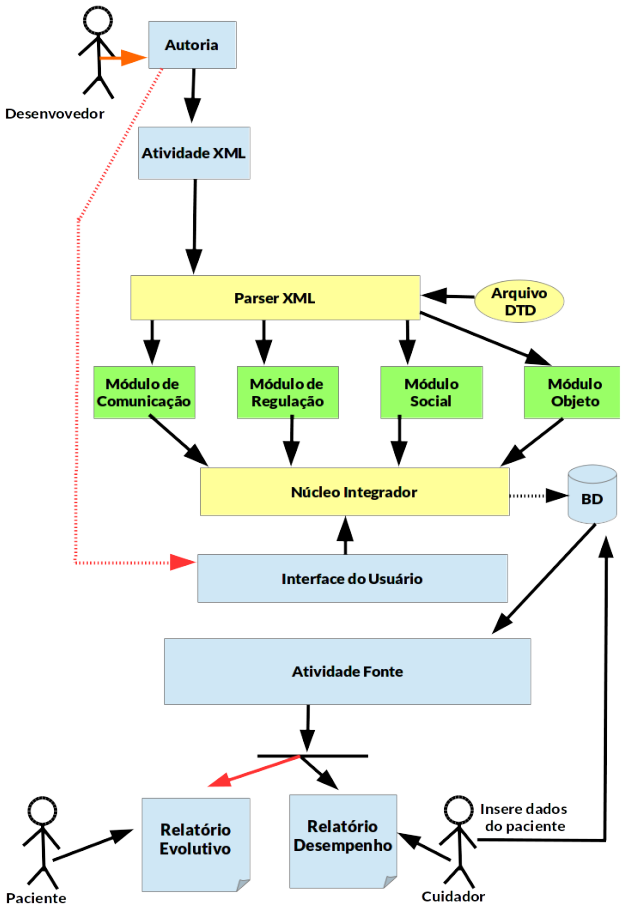

Figura 6.1: *Arquitetura de gamificação.*

A *atividade xml* é composta pelos artefatos de entrada, componentes de *gamificação* que farão parte da atividade terapêutica a ser entregue para o paciente. Após ter sido gerada a *atividade xml* é utilizada pelo artefato de *software parserxml*.

O processamento realizado pelo *parserxml* verifica a corretude do código *xml*, por meio das definições contidas no arquivo *atividade.dtd* seleciona os componentes de *gamificação* contidos no arquivo *atividades xml*.

A busca e seleção dos componentes de *gamificação* são realizadas nos módulos comunicação, regulação, social e objeto, a tabela 6.3 relaciona os componentes de *gamificação* contidos em cada um dos módulos.

As *interfaces* das aplicações terapêuticas são escolhidas pelo desenvolvedor durante o processo de autoria. Os componentes de *gamificação* selecionados e as *interfaces* das aplicações terapêuticas são integradas, por meio do núcleo integrador.

O núcleo integrador, também é resposável pela geração das tabelas da base de dados. O banco de dados é composto por entidades responsáveis por manter dados ligados a relacionamentos, gráfico social, evolução, desempenho e habilidades do paciente. Sendo responsabilidade do cuidador e ou terapeuta manter os dados referentes as habilidades ainda presentes no paciente.

Os relatórios produzidos pela atividade dividem-se em dois tipos: relatório evolutivo e relatório desempenho atividade. O relatório evolutivo traz informações relacionadas à evolução do nível de comprometimento das funções cognitivas do paciente. Esse relatório é de acesso exclusivo do cuidador, do médico e do terapeuta. O relatório de desempenho está relacionado às conquistas do paciente durante a realização de uma atividade. Esse relatório é acessível a paciente e cuidador. O fluxo de funcionamento da arquitetura de *gamificação* (figura 6.2) para a geração da atividade é realizado em quatro passos até a utilização pelo paciente:

1. O desenvolvedor constrói o arquivo XML;
2. *Parser xml* aplica o arquivo DTD e gera a *atividade fonte*;
3. Núcleo integrador integra os elementos de *gamificação*, *interface do usuário* e *artefatos de entrada*; e
4. Atividade disponibiliza relatórios para paciente e cuidador/terapeuta.

Tabela 6.3: *Módulos deGamificação*

Módulo de *Gamificação*	**Componentes**
Comunicação	Avatar, *Feedback*, Narrativas
Regulação	Regras, Habilidade, Perfil
Social	Gráfico social, Cooperação, Competição, Relacionamentos
Objetivo	Missões, Emoções, Pontos, Desafios, Recompensas, Conquistas, Progressão, Insígnias, Níveis

6.3 Descrição do XML das Atividades

O arquivo XML da atividade é gerado por uma ferramenta de autoria disponibilizada ao desenvolvedor. Porém, o desenvolvimento desta ferramenta está fora do escopo do trabalho, ficando como trabalhos futuros.

A estruturação do XML é construída em 2 níveis, atividade e subatividade. A atividade é representada pela *tag Root_activity*, que é formada por uma ou mais subatividades. A *tag Sub_Activity* engloba uma subatividade que está organizada em 4 grupos, *gamificação*, artefatos de entrada, ações e relatório.

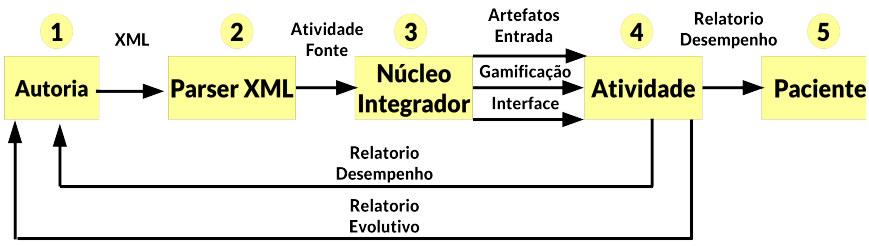

Figura 6.2: *Fluxo de funcionamento da arquitetura de gamificação.*

Os componentes de *gamificação* da arquitetura proposta estão representados pelas *tags* XML, que se agrupam dentro da *tag gamification*, conforme exemplificado na figura 6.3.

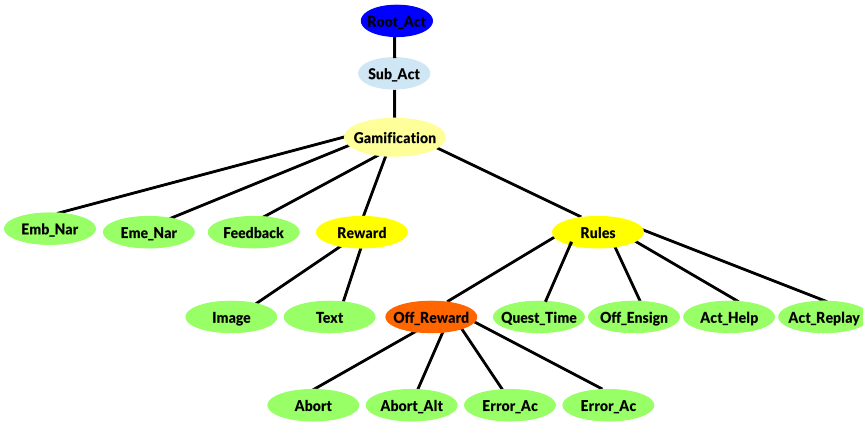

Figura 6.3: *Árvore XML exemplo dos componentes de gamificação de uma atividade.*

Os artefatos de entrada estão agrupados na *tag* XML *Input* e são representados por vídeos, imagens, sons e textos utilizados na criação da atividade. Esses artefatos representam a memória do paciente, dados de localização e dados de atividades da vida real (eventos a participar, cuidados pessoais, cuidados com a casa etc), como o exemplo da figura 6.4.

As ações representam a interação do paciente com a atividade. Uma forma de interação pode ser representada pela resposta de questões sobre fatos passados e eventos (festas, formaturas, recepções etc) a participar. Um segundo modo de interação com uma atividade é realizar o *checklist* de tarefas agendadas relacionadas à casa, à higiene pessoal, ligações telefônicas a realizar, envio de mensagens de *email* etc. A terceira maneira

de interação com uma atividade é o recebimento de lembrete (medicamentos, visita de pessoas, despesas a saldar, etc), conforme exemplemplificado na figura 6.5.

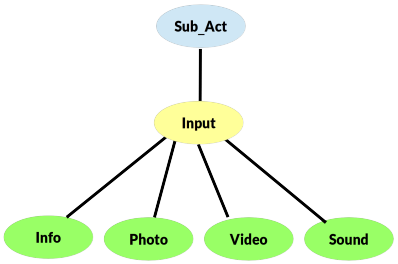

Figura 6.4: *Árvore XML exemplo dos artefatos de entrada de uma atividade.*

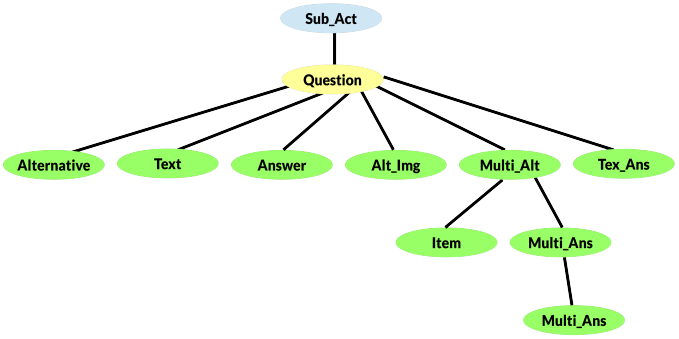

Figura 6.5: *Árvore XML exemplo das ações de uma atividade.*

Os relatórios gerados e integrantes das aplicações são direcionados para pacientes e cuidador/familiar. Nos relatórios direcionados ao paciente estão contidos dados sobre o desempenho na realização da atividade. Nos relatórios direcionados ao cuidador/familiar dados sobre o Exame de estado mental atual do paciente são disponibilizados.

6.4 Modelo de Componentes

O modelo de componentes integrante da arquitetura proposta está organizado em quatro módulos de componentes: módulo comunicação, módulo regulação, módulo objetivos e módulo social. Nas seções a seguir, serão descritos os módulos e componentes

de *gamificação*. A descrição de todos os métodos dos componentes de *gamificação* está no apêndice C.

6.4.1 Módulo Comunicação

O módulo de comunicação está encarregado de emitir mensagens textuais, sonoras e visuais ao usuário, relativas a interação do mesmo com a aplicação. A comunicação direta com o usuário é realizada por um avatar. Segundo Baylor [13], o uso de *avatar* pode impactar na motivação e mudança de atitude, contribuindo na realização de atividades no ambiente virtualizado.

A composição do módulo comunicação é integrada pelos componentes de *feedback*, narrativa e avatar. Na figura 6.6 é apresentada a composição do módulo de comunicação e detalhadas nas seções componente Narrativas, componente *feedback* e componente avatar.

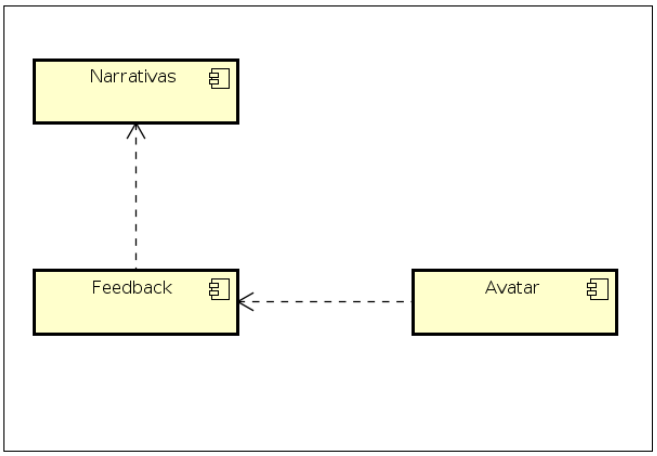

Figura 6.6: *Representação do módulo comunicação.*

Componente Narrativas

O componente narrativas permite o acesso às narrativas dos tipos embutido e emergente descritas, na seção 5.6.1, item narrativas. Esse componente é responsável por acessar a lista de mensagens a ser externada ao usuário, mediante a sua interação com a atividade.

O componente narrativas utiliza os princípios de usabilidade (padronização da funcionalidade e da informação, compatibilidade, consistência).

O componente narrativas disponibiliza um método c*onstrutor* e o método *getNarrative* que retorna as narrativas. A assinatura do método *getNarrative* inclui o retorno do tipo *ArrayList<String>* e um parâmetro de entrada do tipo *ParserNarrative* (tipo pertencente ao *ParserXML*) descritos nas tabelas 6.4 e 6.5.

As narrativas estão definidas como constantes públicas estáticas do tipo inteiro, na *classe narrative* descritas na tabela 6.6.

O método *getNarrative* é acionado no início da atividade e retorna as narrativas do tipo embutida e tipo emergente. A narrativa embutida é utilizada para orientar o usuário sobre o funcionamento da atividade. As narrativas emergentes são retornadas mediante as ações de interação do usuário com a atividade.

A figura 6.7 mostra a sequência de funcionamento do componente narrativas e a relação com o componente *feedback* e a atividade.

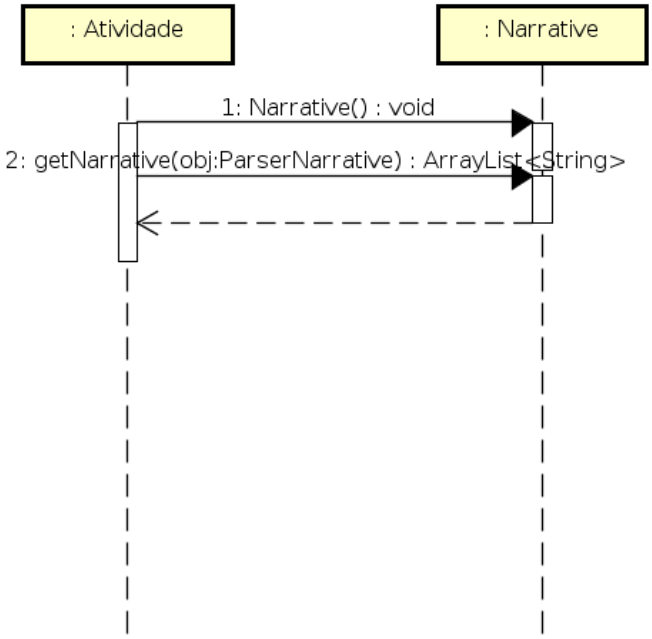

Figura 6.7: *Exemplo de utilização do componente narrativas.*

Tabela 6.4: *Componente narrative método construtor*

	Descrição
Nome Método:	*Narrative*
Propósito:	Criar uma instância do componente *narrative.*
Descrição:	Esse método cria uma instância do objeto representante do componente *narrative*.
Assinatura:	*Narrative(void)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Componente *Feedback*

A função desse componente é permitir o retorno das mensagens ao usuário, mediante a interação com as atividades propostas pela aplicação.

Tabela 6.5: *Componente narrative método getNarrative*

	Descrição
Nome Método:	*getNarrative*
Propósito:	Buscar narrativas de ambas as categorias embarcada ou emergente.
Descrição:	Esse método retorna um objeto do tipo *ArrayList<String>* contendo narrativas embarcadas e emergentes.
Assinatura:	*ArrayList<String>getNarrative(<ParserNarrative obj>)*
Escopo:	*Public*
Retorno:	*ArrayList<String>*
Parâmetros:	*ParserNarrative obj*
obj	contêm a referência as narrativas lidas pelo módulo *ParserXML*.

Tabela 6.6: *Descrição das constantes pertencentes ao componente narrative.*

Constante	Valor	Descrição
EMERGENTINCORRECT	0	Indica uma narrativa emergente em caso de ação incorreta
EMERGENTCORRECT	1	Indica narrativa emergente em caso de ação correta
EMERGENTHELP	2	Indica narrativa emergente para solicitação do primeiro auxílio
EMERGENTNEWHELP	3	Indica narrativa emergente para solicitação do segundo auxílio ou mais auxílios
EMERGENTABORT	4	Indica narrativa emergente para casos de desistência da atividade

O componente *feedback* realiza o acesso às mensagens mantidas pelo componente narrativas (seção 5.6.1 item componente narrativa) e as envia para o componente *avatar* (seção 5.6.3 item componente avatar) a mensagem adequada, para ser reproduzida ao usuário.

O componente *feedback* disponibiliza nove métodos públicos, sendo um construtor, quatro métodos de configuração estética da narrativa, um método que verifica existência da instância do avatar, um método de exibição das narrativas, método de seleção da narrativa e um método de envio de narrativas ao *avatar*.

Alguns dos métodos disponibilizados pelo componente *feedback* estão descritos nas tabelas 6.7 e 6.8. A totalidade dos métodos encontra-se no apêndice C.3

O componente *feedback* utiliza os princípios de usabilidade (consistência, ajuda) e princípios de acessibilidade (visibilidade, padronização da funcionalidade e da informação, uso equitativo).

O método *AvatarExist* é acionado para verificação da existência da instância do componente *avatar*. Caso esta instância exista, o método *SendMessageAvatar* realiza o

envio da mensagem de *feedback* ao *avatar*.

Os métodos de configuração de texto e tempo de exibição das narrativas são utilizados para alterar a configuração visual e duração das mensagens.

A não utilização dos métodos de configuração implica na utilização da configuração padrão (cor texto=vermelho, tamanho da fonte, 25, cor do fundo = ciano e duração da exibição = 10 segundos).

Os métodos de configuração estética do *feedback* devem ser utilizados após a instância do componente *feedback*, se as configurações padrão não forem satisfatórias.

A figura 6.8 mostra o fluxo de funcionamento em alto nível do componente *feedback* e a relação com os componentes narrativa, *avatar* e a atividade.

Integrante da mecânica dos jogos, o *feedback* é representado pelo componente *feedback*, que atua no direcionamento das mensagens e informações adequadas aos propósitos da atividade em curso.

Tabela 6.7: *Componente feedback método construtor*

	Descrição
Nome Método:	*Feedback*
Propósito:	Criar uma instância do componente *feedback*.
Descrição:	Esse método cria uma instância do objeto representante do componente *feedback*.
Assinatura:	*Feedback(void)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.8: *Componente feedback método SeekNarrative*

	Descrição
Nome Método:	*SeekNarrative*
Propósito:	Buscar narrativas de ambas as categorias embarcada ou emergente.
Descrição:	Esse método retorna um objeto do tipo *String* contendo uma narrativa embarcada ou emergente.
Assinatura:	*<String>SeekNarrative(<ArrayList<String> obj>,<int tipo>,<int ordem>)*
Escopo:	*Public*
Retorno:	*String*
Parâmetros:	*ArrayList<String> obj, int tipo, int ordem*
obj	contêm a referência as narrativas lidas pelo módulo *ParserXML*.
tipo	Tipo da narrativa (0 - Embarcada, 1, 2, 3, 4 -Emergente).
ordem	Corresponde a 1ª,2ª, 3ª,..., narrativa do tipo especificado

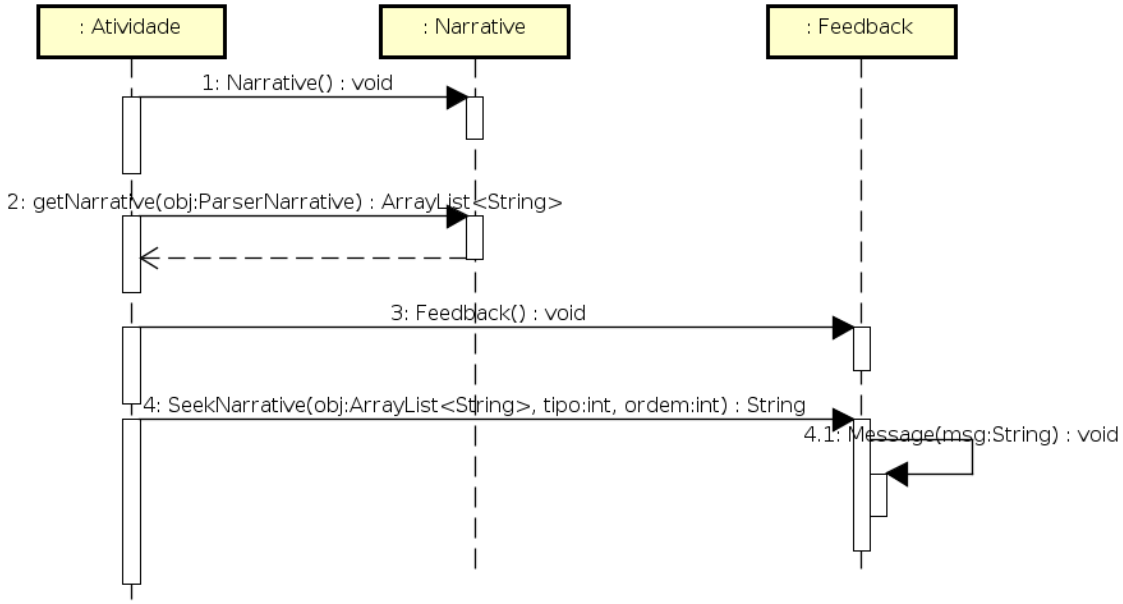

Figura 6.8: *Exemplo de utilização do componente feedback.*

Componente Avatar

A função desse componente é possibilitar a comunicação da aplicação com o usuário, por meio de mensagens de texto, mensagem falada e visual.

O componente *avatar* utiliza os princípios de usabilidade (consistência, visibilidade, padronização da funcionalidade e informação) e princípios de acessibilidade (controle, padronização da funcionalidade e informação) (seção 2.2).

O componente é acionado como resultado da interação do usuário com a atividade. O *avatar* é utilizado como canal de retorno das mensagens de *feedback* disparada pelo componente *feedback* (seção 5.6.2 item componente *feedback*).

A segunda forma de atuação do avatar ocorre no início da atividade, quando as instruções de realização e objetivo da tarefa são explanadas para o usuário.

Alguns dos métodos disponibilizados pelo componente *avatar* estão descritos nas tabelas 6.9 e 6.10. A totalidade dos métodos encontra-se no apêndice C.2

O método *SpeakText* recebe a mensagem enviada pelo componente *feedback* e aciona o sintetizador de voz, com o idioma padrão, para leitura da narrativa.

O componente avatar pode ser configurado para estar adaptado as preferências do usuário. Dentre as configurações, é possível redefinir o idioma do sintetizador de voz utilizando o método *setLanguage*.

As cores, de fundo, texto, tamanho de fonte e o tempo de exibição das mensagens e elementos visuais são configuráveis, para a melhor visualização do usuário. Os métodos *setBackgroundColor*, *setFontColor*, *setSizeFont* e *Timemsg* são utilizados para alterar a configuração.

A representação gráfica do *avatar* é definida pelo método *setImage*, que permite

a escolha da imagem desejada do usuário. O sintetizador de voz é ou não acionado, conforme o status definido pelo método *setSpeak*. O status de ativo ou inativo do sintetizador é verificado pelo método *isSpeak*.

Os métodos de configuração do *avatar* podem ser utilizados após a instância do componente *avatar*. Caso não sejam utilizados, as configurações padrão são adotadas.

A figura 6.9 mostra o fluxo de funcionamento do componente *avatar* e seu relacionamento com outros componentes de *gamificação* e a atividade.

Integrante dos componentes dos jogos o componente *avatar* proposto atua como um indivíduo virtual, que assiste o paciente na realização das atividades. Nos momentos que o paciente se encontra em dificuldades na realização das ações o avatar pode ser invocado para expressar os auxílios e motivações.

Tabela 6.9: *Componente feedback método construtor*

	Descrição
Nome Método:	*Avatar*
Propósito:	Criar uma instância do componente *avatar*.
Descrição:	Esse método cria uma instância do objeto representante do componente *avatar*.
Assinatura:	*Avatar(void)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.10: *Componente avatar método SpeakText*

	Descrição
Nome Método:	*SpeakText*
Propósito:	Realiza a leitura e a fala do texto de uma narrativa.
Descrição:	Esse método é encarregado de realizar a leitura e síntese de voz das mensagens de texto mantidas na lista de narrativas.
Assinatura:	*<void>SpeakText(String msg>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*String msg*
msg	Parâmetro que recebe a mensagem a ser falada.

6.4.2 Módulo Social

O módulo social agrupa tanto os componentes que permitem a troca de experiências entre usuários como também, compara suas habilidades relativas a outros usuários.

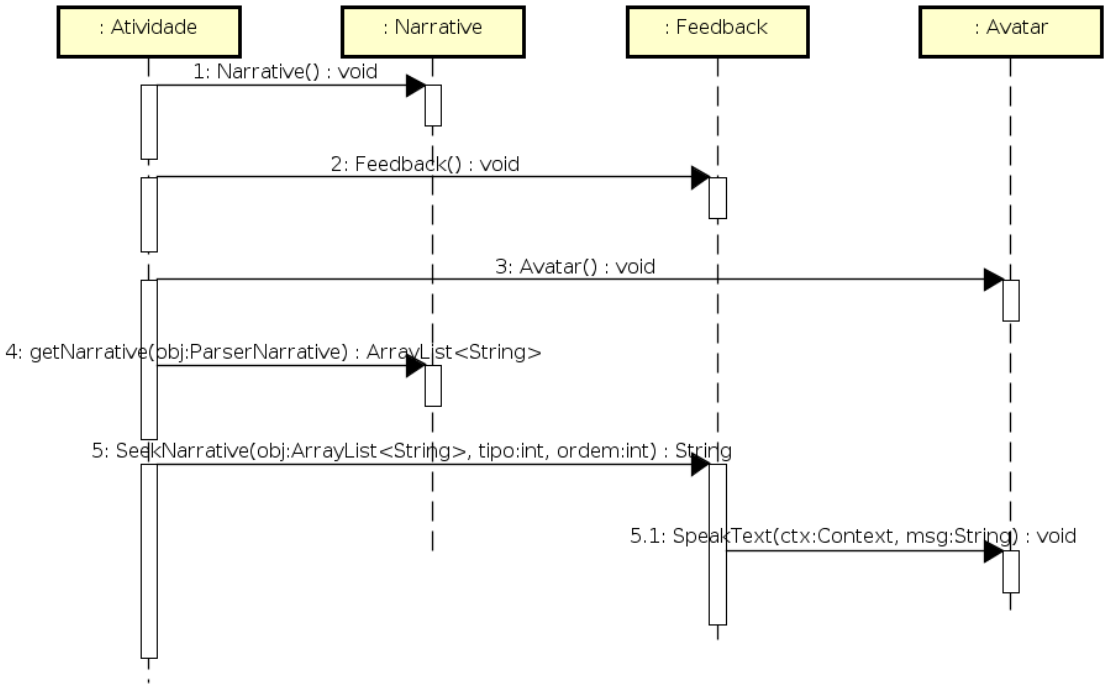

Figura 6.9: *Exemplo de utilização do componente avatar.*

Os componentes que integram esse módulo correspondem aos componentes gráfico social, cooperação, relacionamentos e competição. A composição do módulo social é mostrada na figura 6.10.

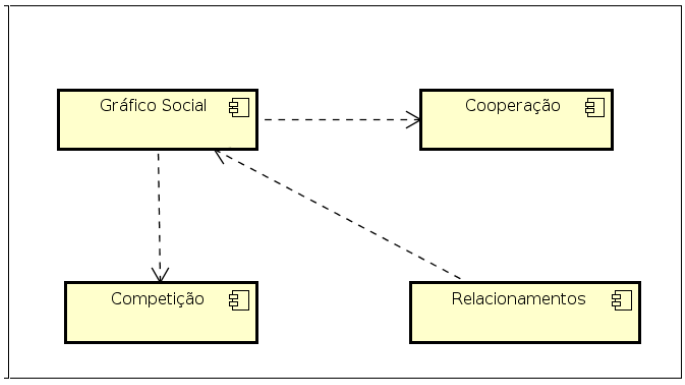

Figura 6.10: *Módulo Social.*

Componente Relacionamentos

A função desse componente é registrar os dados das pessoas que fazem parte das relações do paciente. O método construtor e o método *getRelationship* do componente estão descritos Nas tabelas 6.11 e 6.12. Os outros métodos do componente são descritos no apêndice C.6.

Os métodos *upDateRelationship*, *insertRelationship* e *getRelationship* são utilizados para manutenção dos relacionamentos do usuário. Esses métodos são utilizados pelo cuidador e ou paciente, para criar a rede de relacionamentos.

O acesso aos métodos de manutenção dos relacionamentos são acessíveis independentemente da atividade elaborada pelo cuidador.

A figura 6.11 mostra o fluxo de funcionamento do componente relacionamentos.

Integrante na dinâmica dos jogos, o relacionamento refere-se à forma de interação entre os usuários (companheiros, adversários). O componente relacionamentos apoia a construção da rede social do usuário, registrando dados das relações.

Tabela 6.11: *Componente relacionamento método construtor*

	Descrição
Nome Método:	*Relationship*
Propósito:	Criar uma instância do componente *relationship*.
Descrição:	Esse método cria uma instância do componente *relationship*.
Assinatura:	*Relationship(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.12: *Componente relacionamento método getRelationship*

	Descrição
Nome Método:	*getRelationship*
Propósito:	Retorna uma ocorrência de relacionamento.
Descrição:	Esse método retorna uma ocorrência na base de dados relativo aos relacionamentos do usuário.
Assinatura:	*<Relationship>getRelationship(<String name, String kinship>)*
Escopo:	*Public*
Retorno:	*Relationship*
Parâmetros:	*String name, String kinship*
name	Parâmetro utilizado para localizar uma pessoa da rede social do usuário
kinship	Parâmetro utilizado para localizar o tipo de parentesco da pessoa da rede social do usuário.

Gráfico Social

A função desse componente é projetar as relações do usuário (seção 5.6 item relacionamentos) com pessoas que compartilham dados registrados sobre as conquistas oriundas de suas ações dentro das atividades.

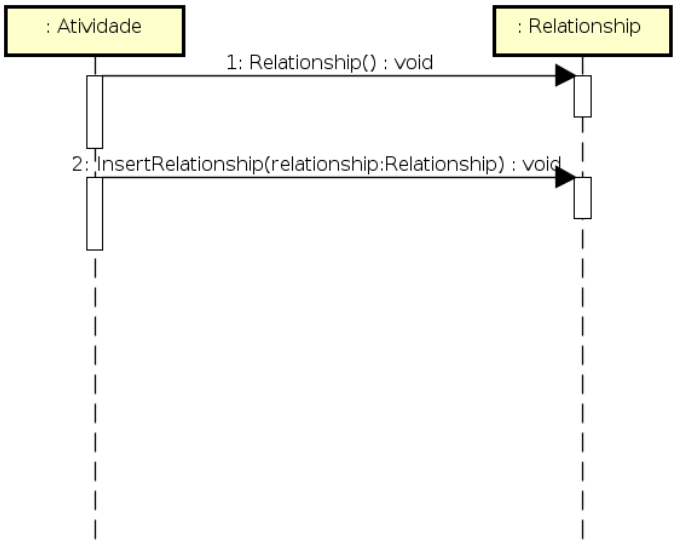

Figura 6.11: *Exemplo de utilização do componente relacionamentos.*

Os métodos construtor e *DisplaySocialgraph* oferecidos pelo componente gráfico social estão descritos nas tabelas 6.13 e 6.14. A relação completa dos métodos oferecidos pelo componente gráfico social estão descritos no apêndice C.7.

Os métodos *upDateSocialGraph*, *insertSocialgraph* e *getSocialgraph* são utilizados para manutenção da rede social do usuário. Esses métodos são acionados sempre que o usuário coopera ou compete com outra pessoa.

O método *DisplaySocialgraph* é utilizado pelo usuário e ou cuidador para visualizar a rede social construída. Esse método é independente da atividade construída pelo cuidador.

A figura 6.12 mostra o fluxo de funcionamento do componente gráfico social e seu relacionamento com outros componentes de *gamificação*. Componente integrante dos jogos o gráfico social permite a interação e visualização dos companheiros participantes de um jogo. O componente grfico social permite a visualização da rede social do usuário. A interação entre os membros da rede ocorre de forma assíncrona, por meio de mensagens de *email*.

Tabela 6.13: *Componente gráfico social método construtor*

	Descrição
Nome Método:	*Socialgraph*
Propósito:	Criar uma instância do componente *socialgraph*.
Descrição:	Esse método cria uma instância do componente *socialgraph*.
Assinatura:	*Socialgraph(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

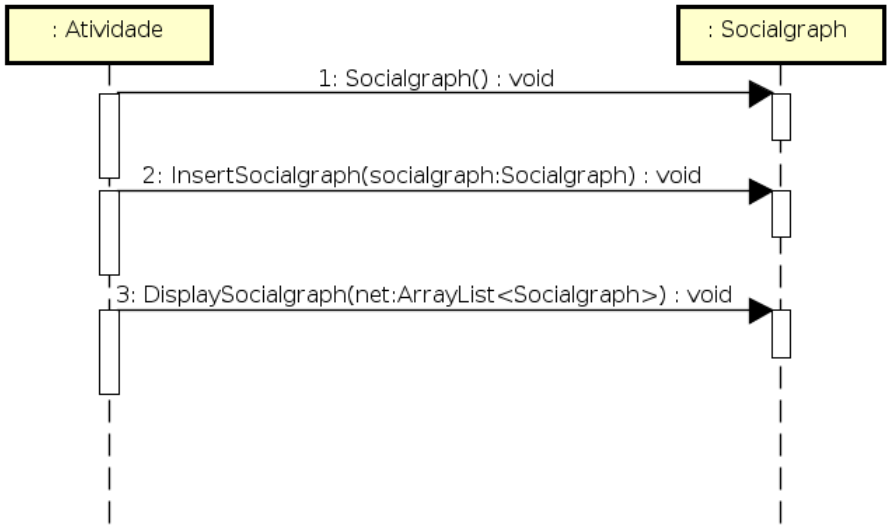

Figura 6.12: *Exemplo de utilização do componente gráfico social.*

Tabela 6.14: *Componente gráfico social método DisplaySocialgraph*

	Descrição
Nome Método:	*DisplaySocialgraph*
Propósito:	Mostra a rede social do usuário.
Descrição:	Esse método mostra a rede social do usuário.
Assinatura:	*<void>DisplaySocialgraph(ArrayList<Socialgraph> socialgraph>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*ArrayList<Socialgraph> socialgraph*
socialgraph	Parâmetro contêm a rede social do usuário.

Componente Competição

A função desse componente é definir os níveis de dificuldade (pontos para mudança de nível, tempo de resposta para as ações) atribuídos aos competidores virtuais durante a realização de uma atividade. Em casos de disparidade de habilidades dos competidores, o mecanismo de *handcap* (seção 5.6.2 item competição) é adotado, para equilíbrar as forças entre competidores.

Os métodos constructor e *setDifficult* definidos no componente competição são descritos nas tabelas 6.15 e 6.16. Os outros métodos integrantes do componente competição estão descritos por completo no apêndice C.4.

O costrutor do componente competição recebe como parâmetro um objeto do tipo *ParserCompetition* gerado pelo *ParserXML* que define o nível de dificuldade a ser atribuído ao competidor do paciente.

Os métodos com prefixo *set* e *get* redefinem e retornam respectivamente os dados relativos aos níveis de dificuldade das atividades a serem realizadas pelos competidores.

Os níveis de dificuldade possíveis aplicados ao competidor são definidos na tabela 6.17.

Os métodos disponibilizados pelo componente competição são utilizados se a atividade elaborada pelo cuidador incluir a a *tag competition* e o nível de dificuldade atribuído ao competidor.

A figura 6.13 mostra o fluxo de funcionamento do componente competição e seu relacionamento com outros componentes de *gamificação*.

Integrante da mecânica dos jogos, a competição é representada pelo componente competição, que define os níveis de dificuldade para os competidores do paciente durante uma atividade. Esse componente equipara o nível dos competidores, para permitir disputas mais equilibras.

O desequilíbrio entre os competidores leva a perda da motivação e do interesse em competir.

Tabela 6.15: *Componente competição método construtor*

	Descrição
Nome Método:	*Competition*
Propósito:	Criar uma instância do componente *competitionh*.
Descrição:	Esse método cria uma instância de um objeto da classe *Competition*, que controla os níveis de dificuldade aos competidores do usuário da atividade.
Assinatura:	*Competition(<ParserCompetition parsercompetition>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*ParserCompetition parsercompetition*
parsercompetion	Parâmetro que define o nível de dificuldade a ser utilizado no mecanismo de *handcap*.

Tabela 6.16: *Componente competiçãol método setDifficult*

	Descrição
Nome Método:	*setDifficult*
Propósito:	Define e redefine o nível de dificuldade para o competidor.
Descrição:	Esse método define e redefine o nível de dificuldade para realização de uma atividade por parte do competidor.
Assinatura:	*<void>setDifficulty(<int difficult>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*int difficult*
difficult	Parâmetro contêm o nível de dificuldade para o competidor.

Tabela 6.17: *Constantes relativas aos níveis de dificuldade atribuídos as atividades realizadas pelo competidor*

Constante	Valor	Descrição
EASY	0	Código relativo a uma atividade com nível fácil de dificuldade.
INTRMADIATE	1	Código relativo a uma atividade com nível intermediário de dificuldade.
DIFFICULT	2	Código relativo a uma atividade com nível difícil de dificuldade.

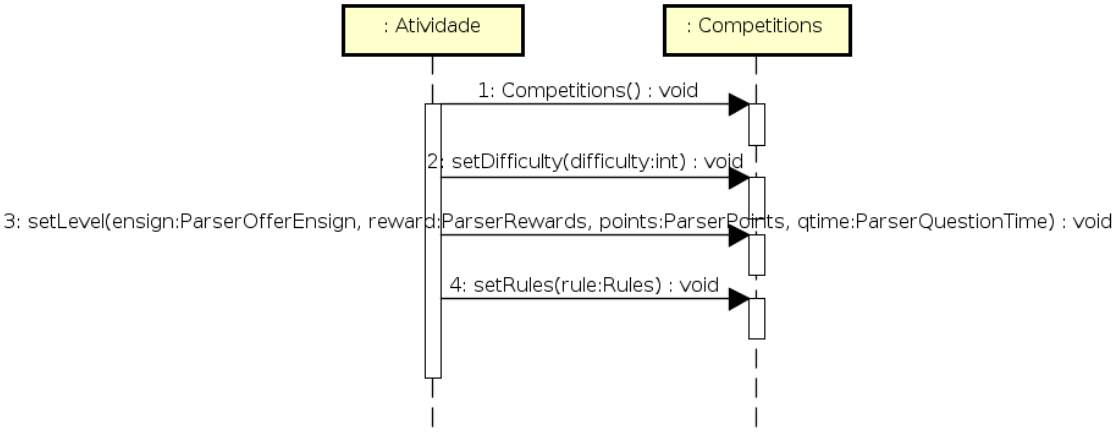

Figura 6.13: *Exemplo de utilização do componente competição.*

Componente Cooperação

A função desse componente é permitir ao usuário enviar dicas de como realizar atividades. As dicas são enviadas para usuários da rede social do paciente.

A cada cooperação realizada pelo usuário, o componente *feedback* (seção 5.6.2 item componente *feedback*) é acionado, juntamente com o componente insígnias (seção 5.6.3 item insígnias), para premiar o usuário pela sua colaboração.

O envio de questões problema e de possíveis soluções descobertas pelos pacientes é realizado por email. Os endereços de *emails* estão previamente cadastrados, gerenciados pelo componente relacionamentos (5.6.1 item relacionamentos).

As tabelas 6.18. descrevem o método construtor e o método *CreateMessage* fornecidos pelo componente cooperação. A relação completa dos métodos do componente cooperação estão descritos no apêndice C.5.

Os métodos com prefixo *set* e *get* definem e retornam respectivamente, os dados relativos ao integrante das relações sociais que receberam auxílio do usuário.

O método *sendMessage* envia para o *email* escolhido pelo usuário, os dados de cooperação das atividades. E o método *createMessage* cria a mensagem de cooperação a ser transmitida.

Os métodos do componente cooperação são utilizados pelo usuário na forma de

um formulário a ser evocado durante a atividade. A figura 6.14 mostra o fluxo de funcionamento do componente cooperação e seu relacionamento com outros componentes de *gamificação*.

Integrante da mecânica dos jogos, a cooperação é representada pelo componente cooperação e em conjunto com o componente gráfico social possibilitam ações colaborativas entre os usuários da rede de relacionamento.

A possibilidade de colaborar na solução de atividades de outros usuários e criar discussões em torno do problema levantado estimula o raciocínio, a atenção, o exercício da memória, a busca de novos conhecimentos por parte dos usuários.

O estreitamento dos laços entre os integrantes da rede social do usuário possibilita a troca de informações sobre a DA, organização de atividades presenciais.

Tabela 6.18: *Componente cooperação método construtor*

	Descrição
Nome Método:	*Cooperation*
Propósito:	Criar uma instância do componente *cooperation*.
Descrição:	Esse método cria uma instância de um objeto da classe *Cooooperation*.
Assinatura:	*Cooperation(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.19: *Componente cooperação método CreateMessage*

	Descrição
Nome Método:	*CreateMessage*
Propósito:	Criar uma a mensagem de cooperação.
Descrição:	Esse método cria a mensagem a ser enviada ao último contato ligado a rede social do usuário, que recebeu cooperação.
Assinatura:	*CreateMessage(<String> msg)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*String msg*
msg	Parâmetro que contêm os dados variáveis da mensagem a ser transferida via *email* ao último contato do usuário.

6.4.3 Módulo Regulação

O módulo regulação reúne componentes com a função de controlar os níveis de dificuldade de uma atividade, conforme a perda das habilidades cognitivas do paciente diminui. O módulo regulação é composto pelos componetes, habilidades, perfil e regras descritos nas três próximas subseções.

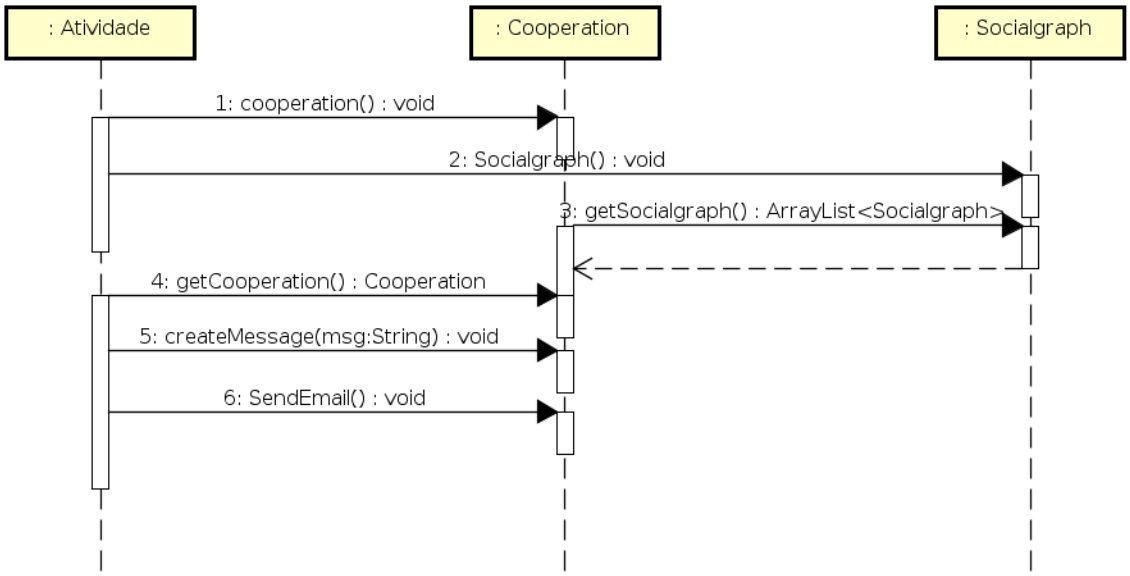

Figura 6.14: *Exemplo de utilização do componente cooperação.*

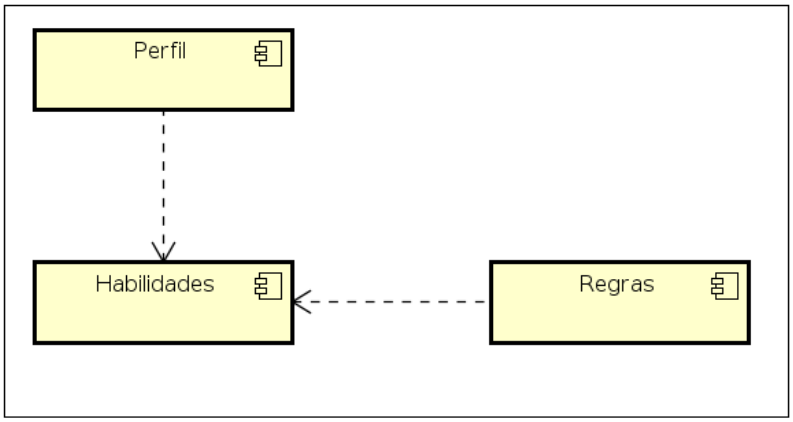

Figura 6.15: *Módulo Regulação.*

Componente Regras

A função desse componente é definir as regras e o fluxo de execução da aplicação. O componente regras, por exemplo, permite definir quais as ações do usuário dentro de uma atividade serão pontuadas e o valor desta pontuação.

O fluxo de execução permite o controle de quais ações o usuário pode realizar, qual a sua ordem de execução. As tabelas 6.20 e 6.21 descrevem os métodos construtor e o método *CheckActionAbort* do componente regras. A relação completas dos métodos do componente regras estão descritas no apêndice C.16.

Os métodos com prefixo *Check* verificam se as ações do usuário estão dentro das regras e se as ações resultam na mudança de nível dentro da atividade. Os métodos de checagem devem ser utilizados após as ações do usuário.

Os métodos com sufixo *increment* contabilizam as ações realizadas pelo usuário durante uma atividade. E os métodos com sufixo initial zeram a contagem de ações dos

usuários durante uma atividade.

A figura 6.16 mostra o fluxo de funcionamento do componente regras e seu relacionamento com os outros componentes de *gamificação*.

Integrante da dinâmica dos jogos, o componente regras limita a atuação do usuário. O componente regras desenvolvido permite a configuração de regras diferentes em cada atividade, bem como, o grau de liberdade.

Tabela 6.20: *Componente regras método construtor*

	Descrição
Nome Método:	*Rules*
Propósito:	Criar uma instância do componente *rules*.
Descrição:	Esse método cria uma instância de um objeto da classe *Rules*.
Assinatura:	*Rules(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.21: *Componente regras método CkeckActionAbort*

	Descrição
Nome Método:	*CheckActionAbort*
Propósito:	Verificar a ação realizada pelo usuário.
Descrição:	Esse método verifica a ação realizada pelo usário, se esta foi uma ação de desistência.
Assinatura:	*<boolean>CheckActionAbort(<void>)*
Escopo:	*Public*
Retorno:	*boolean*
Parâmetros:	*void*

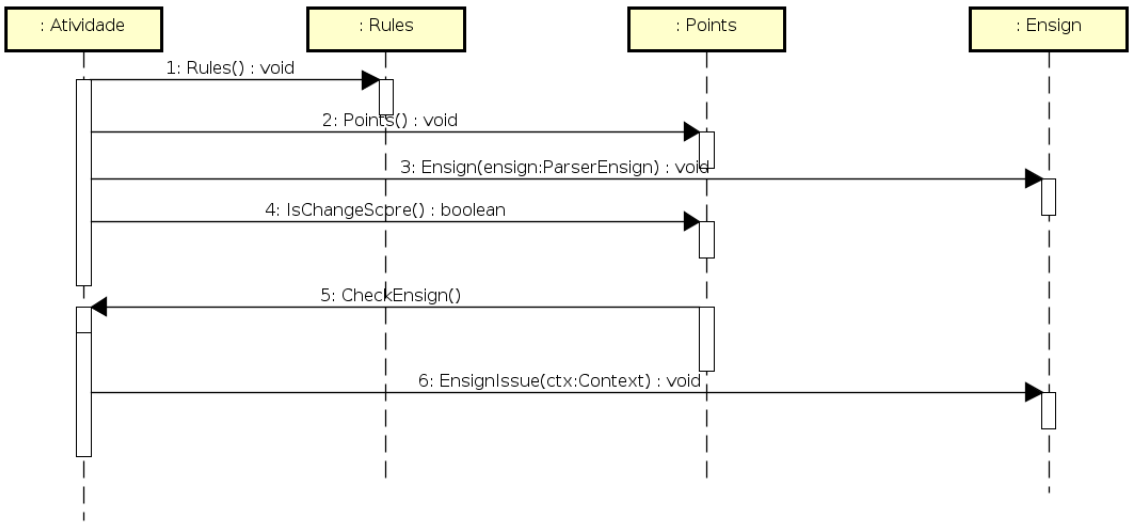

Figura 6.16: *Exemplo de utilização do componente regras.*

Componente Habilidades

É o componente de suporte aos dados relativos à evolução das habilidades restantes nos usuários. O componente verifica a duração e a forma de execução das atividades realizadas pelo paciente. Os dados coletados são persistidos na base de dados e posteriormente o relatório, com os resultados do paciente são enviados ao cuidador.

Os métodos *InsertAction* e *seekAction* são descritos, respectivamente, nas tabelas 6.22 e 6.23. A relação completa dos métodos habilidades estão descritos no apêndice C.17.

Os dados coletados pelo método *SeekAction* referem-se aos acertos, erros, solicitação de auxílio e desistências que possam vir a acontecer durante a realização de uma atividade. É função do método *InsertAction* persistir os dados na base de dados.

Os dados registrados são disponibilizados para o cuidador através do método *DisplayAction* na forma de um relatório. A figura 6.17 ilustra o funcionamento do componente habilidades.

Tabela 6.22: *Componente habilidades método InsertAction*

	Descrição
Nome Método:	*InsertAction*
Propósito:	Persiste o resultados das ações do usuário no banco de dados.
Descrição:	Esse método registra os dados relativos a execução de uma atividade na base de dados.
Assinatura:	*<void>InsertAction(<Rules rule>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*Rules rule*
rule	Parâmetro que contêm o resultado da execução das ações do usuário em uma atividade.

Tabela 6.23: *Componente habilidades método SeekAction*

	Descrição
Nome Método:	*SeekAction*
Propósito:	Retorna os dados relativo as ações do usuário.
Descrição:	Esse método retorna os dados relativos as ações do usuário realizadas durante uma atividade.
Assinatura:	*<Rules>SeekAction(<void>)*
Escopo:	*Public*
Retorno:	*Rules*
Parâmetros:	*void*

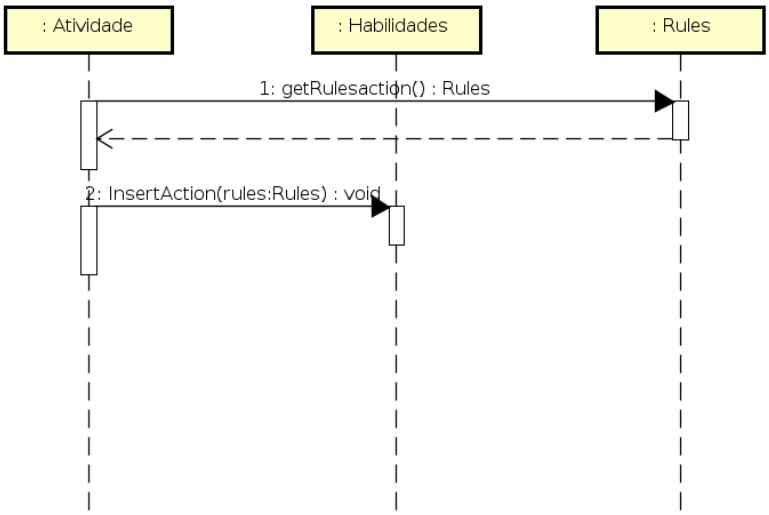

Figura 6.17: *Exemplo de uso do componente habilidades.*

Componente Perfil do Usuário

É o componente de suporte aos usuários do aplicativo, sendo os usuários de dois tipos: pacientes e cuidadores (seção 5.4 item perfil do usuário). O componente perfil do usuário identifica o usuário e o direciona para dois possíveis caminhos de utilização da aplicação: o caminho do paciente e o caminho do cuidador.

No caminho do paciente, o usuário tem à sua disposição somente a possibilidade de visualizar os artefatos (imagens, vídeos, sons,mensagens) e ações relativas as atividades terapêuticas estabelecidas pelo cuidador. No caminho cuidador, o usuário realizará a alimentação e manutenção do aplicativo com dados de perfil do usuário, atividades, artefatos sobre a vida do paciente, as habilidades físicas e cognitivas ainda presentes no paciente.

Os dados estão persistidos no banco de dados do aplicativo. Esse componente se relaciona com os componentes habilidades 6.4.3 item componente habilidades), 6.4.1 item componente *feedback*), sendo que o componente *feedback* realiza somente a leitura de dados disponibilizados pelo componente perfil do usuário.

6.4.4 Módulo Objetivos

Neste módulo encontram-se agrupados os componentes que proporcionam o acompanhamento das atividades e o registro das ações do usuário durante uma sessão terapêutica.

Os componentes integrantes desse módulo são apresentados na figura 6.18 e descritos em alto nível nas seções a seguir:

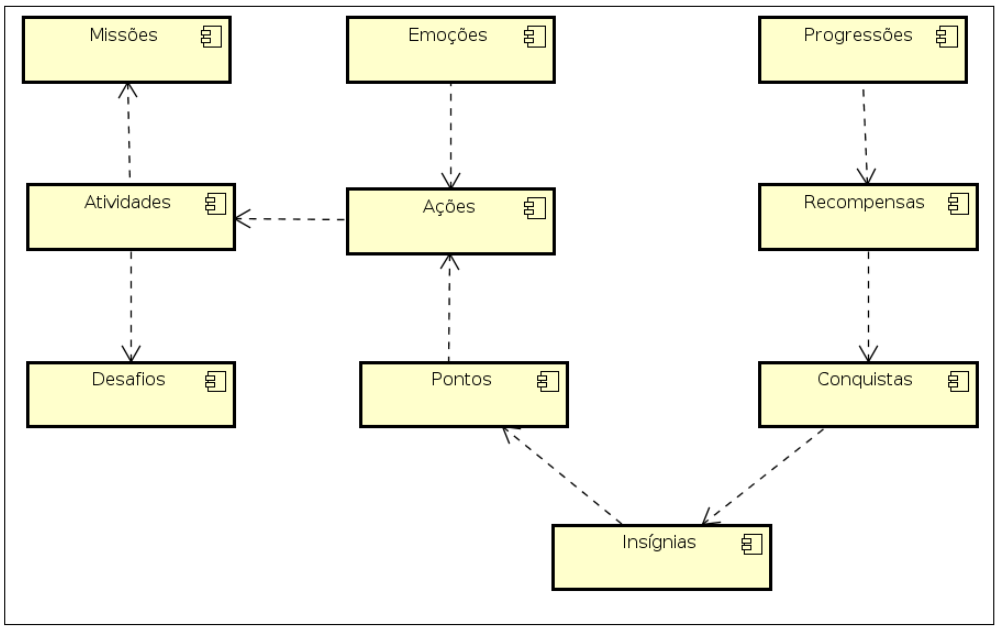

Figura 6.18: *Módulo Objetivos.*

Componente Progressão

A função desse componente é comparar a evolução do usuário em relação aos resultados obtidos em sessões anteriores. Os dados coletados (tempo de execução de uma atividade, erros na realização na atividades, solicitação de auxílio ao avatar, data das sessões, e desistências) por esse componente são persistidos em uma base de dados.

Há duas visualizações distintas baseadas nesses dados: a primeira representando a evolução dos feitos do usuário na forma de *feedbacks* (seção 5.6.2 item componente *feedback*) e a segunda com a evolução da doença disponibilizada ao cuidador.

A progressão do usuário é visualizada também pela evolução dos pontos, conquistas e níveis durante a realização das missões.

Os métodos construtor e *MEEMCalculator* disponibilizados pelo componente progressão são descritos nas tabelas 6.24 e 6.25. A relação completa dos métodos disponibilizados pelo componente *progressão* estão listados no apêndice C.9.

O método *MeemCalculator* é invocado ao término de cada atividade realizada pelo usuário. O cálculo do índice MEEM é persistido em base de dados.

O método *getProgressPatient* retorna ao usuário as conquistas e condecorações conseguidas durante a execução da atividade. As conquistas poderão ser visualizadas pelo usuário a qualquer momento. A figura 6.19 mostra o fluxo de funcionamento do componente progressão e seu relacionamento com outros componentes de *gamificação*.

Integrante da dinâmica dos jogos, o componente progressão passa a ideia de evolução dos usuários dentro da atividade. O componente progressão desenvolvido atua em dois níveis de evolução.

O primeiro nível de evolução possibilita ao paciente verificar como está o seu caminhar dentro da atividade, através da pontuação e das conquistas. O segundo nível de evolução está relacionado ao avanço da DA.

Ao verificar seu progresso dentro da atividade, o usuário pode se sentir motivado em melhorar seu desempenho em relação a execuções anteriores.

Tabela 6.24: *Componente progressão método construtor*

	Descrição
Nome Método:	*Progress*
Propósito:	Criar uma instância da classe *progress*.
Descrição:	Esse método cria uma instância de um objeto da classe *Progress*.
Assinatura:	*Progress(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.25: *Componente progressão método MeemCalculator*

	Descrição
Nome Método:	*MeemCalculator*
Propósito:	Calcular o índice MEEM do usuário.
Descrição:	Esse método permite calcular o índice MEEM relacionado a evolução da saúde do usuário.
Assinatura:	*<Progress>MeemCalculator(<void>)*
Escopo:	*Public*
Retorno:	*Progress*
Parâmetros:	*void*

Componente Pontos

A função desse componente é contabilizar a pontuação atingida pelo usuário a cada ação realizada, mesmo que uma ação seja realizada parcialmente.

Esse componente disponibiliza cinco métodos: o construtor, para instanciar o componente, dois métodos de incremento da pontuação e dois métodos de obtenção do *status* da pontuação. Nas tabelas 6.26 e 6.27 estão listados os métodos construtor e *Score*.

A relação completa de métodos do componente pontos (5.6 item pontos) está descrita no apêndice C.8.

Os métodos *Incrementcorrect* e *Incrementerror* são utilizados no incremento da pontuação, após a realização de uma ação proposta pela atividade. O método *IsChangeScore* retorna o *status* de mudança ou permanência da pontuação, este *status* é utilizado para a atribuição de recompensas, insígnias, mudança de nível.

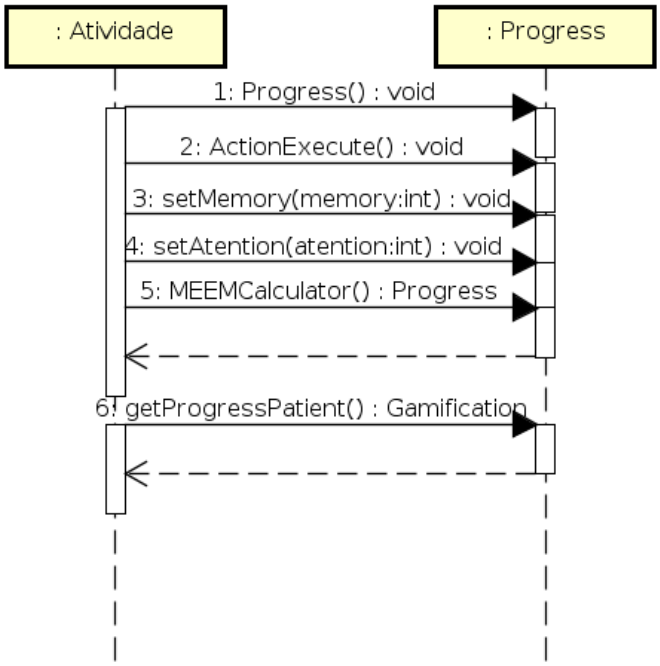

Figura 6.19: *Exemplo de uso do componente progressão.*

O método *Score* retorna a pontuação total do usuário conquistada no decorrer da atividade. O método *Score* é utilizado para atualizar visualmente o placar, quando este estiver presente no display.

A figura 6.20 mostra o fluxo de funcionamento do componente pontos e seu relacionamento com outros componentes de *gamificação*.

Integrante dos componentes dos jogos os pontos representam uma forma de progresso, assim como os níveis. O componente pontos desenvolvido atua no registro da pontuação dentro da atividade.

A pontuação aliada ao nível tende a causar um sentimento de satisfação no usuário, pois representa a capacidade de vencer as dificuldades impostas pela atividade.

Tabela 6.26: *Componente pontos método construtor*

	Descrição
Nome Método:	*Points*
Propósito:	Criar uma instância da classe *points*.
Descrição:	Esse método cria uma instância de um objeto da classe *Points*.
Assinatura:	*Points(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.27: *Componente pontos método Score*

	Descrição
Nome Método:	*Score*
Propósito:	Retorna a pontuação do usuário.
Descrição:	Esse método retorna a pontuação conseguida pelo usuário na realização de ações executadas dentro de uma atividade.
Assinatura:	*<int>Score(<void>)*
Escopo:	*Public*
Retorno:	*int*
Parâmetros:	*void*

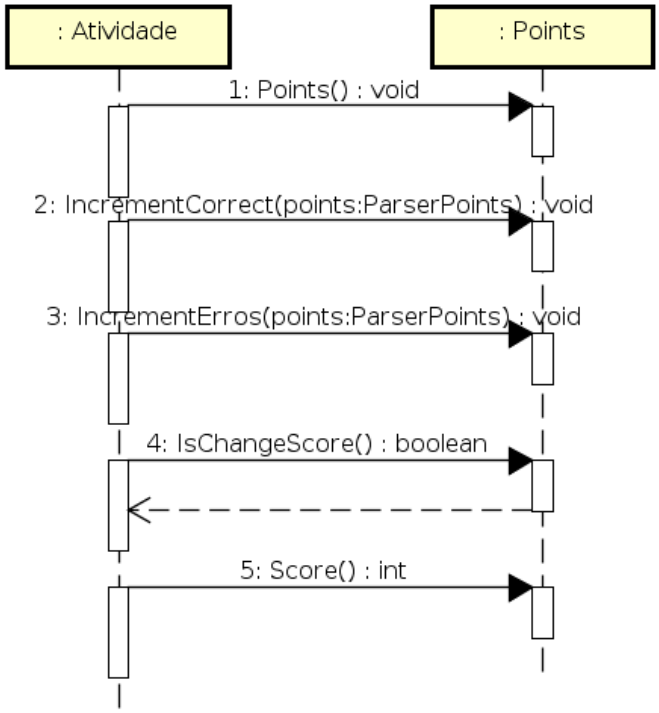

Figura 6.20: *Exemplo de uso do componente pontos.*

Componente Níveis

A função deste componente é definir os níveis de dificuldade e o número de níveis a serem conquistados pelo usuário.

A cada nível conquistado pelo usuário a dificuldade para realização das ações da atividade se altera gradativamente.

Os métodos construtor e *UpLevel* integrantes do componente níveis são descritos nas tabelas 6.28 e 6.29. A relação completa dos métodos do componente níveis são descritos no apêndice C.10. Os métodos, cujos nomes iniciam com os prefixos *Up* e *Down* atuam diretamente nos níveis de exigência atribuída as ações executadas pelo usuário. Os níveis iniciais de exigência são definidos pelo XML criado pelo cuidador e mantidos no objeto *ParserXML*. Os percentuais usados na elevação ou redução dos níveis

de dificuldade são definidos na classe *LevelData* e manipulados pelos métodos *sets* do objeto *levels* atributo da classe *Levels*. Os métodos da classe *Levels* são evocados após a verificação das respostas proferidas pelo usuário durante a realização das ações de uma atividade. Os métodos da classe *Leveldata* são descritos Nas tabelas 6.30 e 6.31. A relação completa dos métodos da classe *LevelData* são descritos no apêndice C.10.1.

A figura 6.21 mostra o fluxo de funcionamento do componente níveis e seu relacionamento com outros componentes de *gamificação*.

Assim como os pontos, o componente níveis integra os componentes dos jogos. O componente níveis desenvolvido gerencia os níveis de exigência necessários para a realização da atividade.

Esse componente pode influenciar de modo positivo quando o usuário se dá conta da sua capacidade de vencer as dificuldades impostas na atividade.

Tabela 6.28: *Componente níveis método construtor*

	Descrição
Nome Método:	*Levels*
Propósito:	Criar uma instância da classe *Levels*.
Descrição:	Esse método cria uma instância de um objeto da classe *Levels*.
Assinatura:	*Levels(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.29: *Componente níveis método UpLevel*

	Descrição
Nome Método:	*UpLevel*
Propósito:	Ampliar o nível da atividade.
Descrição:	Esse método amplia o nível de dificuldade da atividade.
Assinatura:	*<void>UpLevel(<int level>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*int level*
level	Parâmetro que determina o novo nível de dificuldade da atividade.

Componente Recompensas

A função desse componente é definir os tipos de recompensas a serem oferecidas ao usuário. A relação de recompensas é mantida em uma lista encadeada e oferecida ao usuário quando este realiza suas atividades com êxito.

Tabela 6.30: *Componente LevelData método construtor*

	Descrição
Nome Método:	*LevelData*
Propósito:	Criar uma instância da classe *LevelData*.
Descrição:	Esse método cria uma instância de um objeto da classe *LevelData*.
Assinatura:	*Levels(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.31: *Componente Leveldata método setLevelid*

	Descrição
Nome Método:	*setLevelid*
Propósito:	Define o número que identifica o nível.
Descrição:	Esse método define e redefine o *id* do nível de uma atividade.
Assinatura:	*<void>setLevelid(<int id>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*int id*
id	Parâmetro que define o identificador que representa o número do nível da atividade.

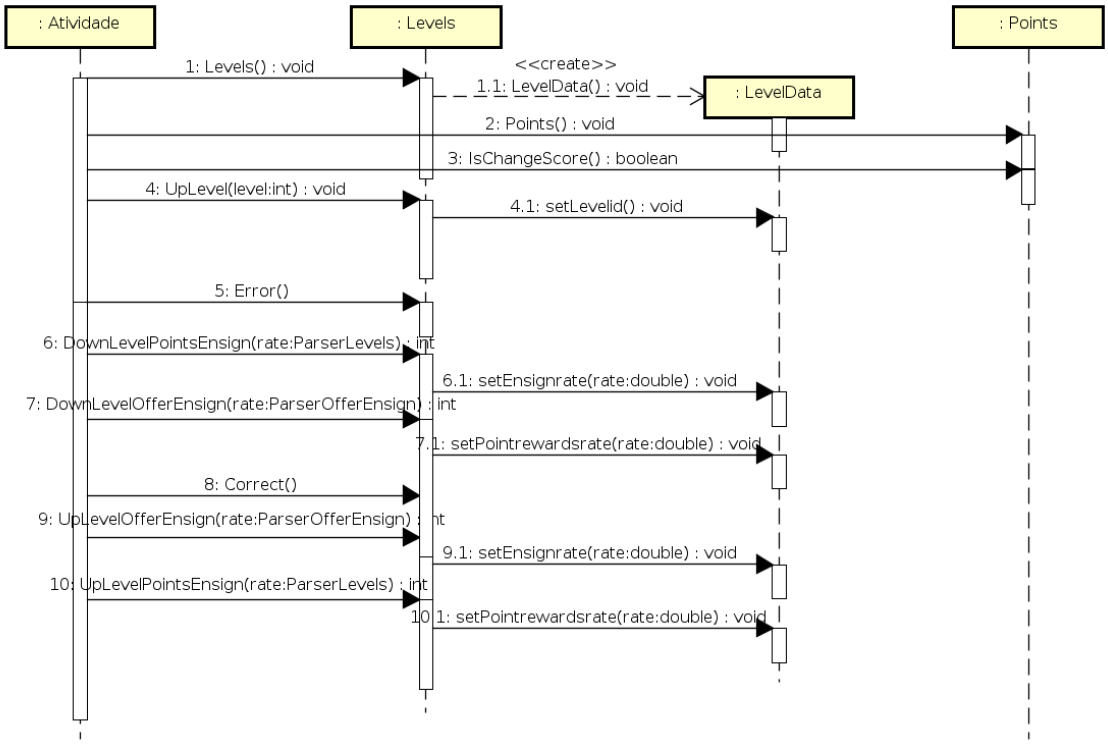

Figura 6.21: *Exemplo de uso do componente níveis.*

Os métodos construtor e *SeekReward* disponibilizados pelo componente recompensas são descritos nas tabelas 6.32 e 6.33. Os demais métodos pertencentes ao compo-

nente recompensas estão descritos no apêndice C.15.

O construtor da classe *Rewards* recebe como parâmetro um objeto do tipo *ParserReward*, com a lista de recompensas. O objeto do tipo *ParserReward* é gerado na etapa de *parser* pelo *ParserXML*.

O método *RewardIssue* é executado após a conquista de uma recompensa, conseguida com a pontuação obtida pelas ações realizadas durante a atividade.

O método *SeekReward* é executado logo após a instância do objeto *reward*, que contém as recompensas. O uso das recompensas é sequencial, sendo oferecidas a cada ação (abortar, erro).

As recompensas devem fazer sentido para o usuário para que possam surtir efeitos benéficos ao usuário.

A figura 6.21 mostra o fluxo de funcionamento do componente recompensas e seu relacionamento com outros componentes de *gamificação*.

Integrante da mecânica dos jogos, o componente recompensas indica quais os benefícios a serem oferecidos ao usuário. O oferecimento de recompensas desperta um sentimento de satisfação pelo reconhecimento do empenho e da capacidade de realização do usuário.

Tabela 6.32: *Componente recompensas método construtor*

	Descrição
Nome Método:	*Rewards*
Propósito:	Criar uma instância da classe *Rewards*.
Descrição:	Esse método cria uma instância de um objeto da classe *Rewards*.
Assinatura:	*Rewards(<ParserReward reward>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*ParserReward reward*
reward	Parâmetro contém o objeto com as recompensas disponíveis a serem oferecidas na realização da atividade.

Componente Insígnias

Este componente representa visualmente as condecorações alcançadas pelo usuário. No componente regras (seção 5.6.1 item componente regras) são estabelecidas como e quando uma insígnia pode ser oferecida ao usuário.

O componente insígnias utiliza os princípios de usabilidade (visibilidade, fácil aprendizado, consistência, padronização da funcionalidade e informação) (seção 2.2).

A cada conquista atingida registrada pelo componente conquistas (seção 5.6.3 item componente conquistas) e os pontos atribuídos pelo componente pontos (seção 5.6.3 item componente pontos), uma insígnia é oferecida ao usuário.

Tabela 6.33: *Componente recompensas método SeekReward*

	Descrição
Nome Método:	*SeekReward*
Propósito:	Busca as recompensas a serem oferecidas.
Descrição:	Esse método retorna a lista de recompensas a serem oferecidas mediante as ações realizadas pelo usuário durante uma atividade.
Assinatura:	*ArrayList<String>SeekReward(<ParserReward obj>)*
Escopo:	*Public*
Retorno:	*ArrayList<String>*
Parâmetros:	*ParserReward obj*
obj	Parâmetro que contêm as recompensas a serem oferecidas ao usuário.

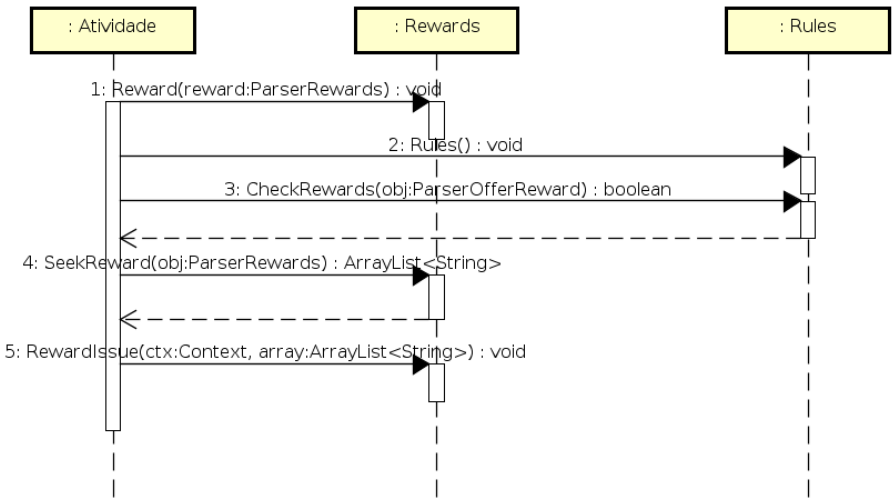

Figura 6.22: *Exemplo de uso do componente recompensas.*

As tabelas 6.34 e 6.35 contêm os métodos construtor e *SeekEnsign* respectivamente. A relação completa dos métodos do componente insígnias estão descritos no apêndice C.14.

Os métodos *SeekEnsign* e *EnsignIssue* são utilizados para resgatar e exibir, respectivamente as condecorações conseguidas pelo usuário mediante as suas ações dentro de uma atividade.

A figura 6.23 mostra o fluxo de funcionamento do componente insígnias e seu relacionamento com outros componentes de *gamificação*.

Componente integrante dos jogos, as insígnias/condecorações representam visualmente os feitos do usuário. O componente ensígnias desenvolvido permite a visualização da imagem de uma insígnia, representando o reconhecimento dos esforços do usuário em vencer dificuldades, a capacidade de realização. Este reconhecimento produz um sentimento de satisfação e afirmação da capacidade de realizar do usuário.

Tabela 6.34: *Componente insígnias método construtor*

	Descrição
Nome Método:	*Ensign*
Propósito:	Instanciar um objeto do tipo insígnias.
Descrição:	Esse método cria uma instância do objeto insígnias, com as recompensas a serem oferecidas ao usuário.
Assinatura:	*Ensign(<ParserEnsign ensign>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*ParserEnsign ensign*
ensign	Parâmetro contêm o objeto com as insígnias disponíveis a serem oferecidas na realização da atividade.

Tabela 6.35: *Componente Insígnias método SeekEnsign*

	Descrição
Nome Método:	*SeekEnsign*
Propósito:	Busca a insígnia a ser oferecida ao usuário.
Descrição:	Esse método busca as insígnias a serem oferecidas com a realização das ações propostas na atividade.
Assinatura:	*<void>SeekEnsign(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

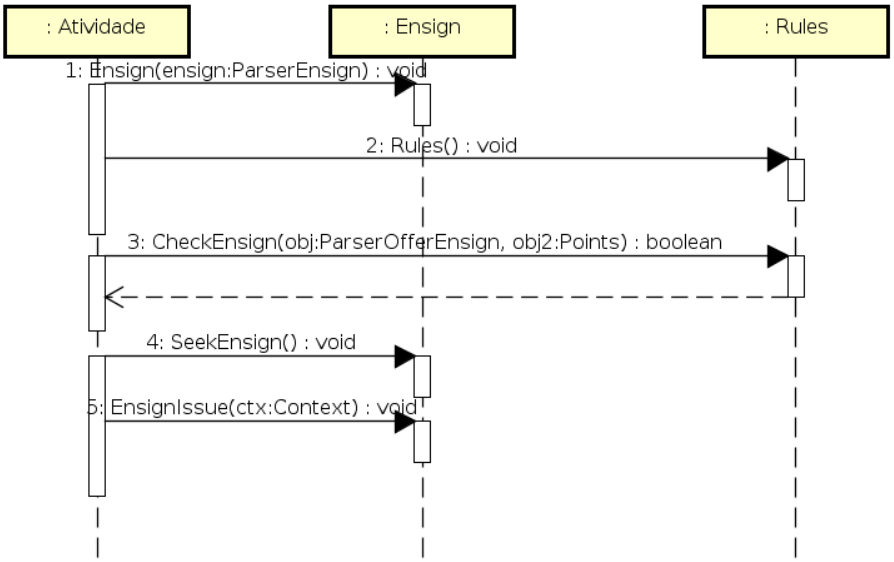

Figura 6.23: *Exemplo de uso do componente insígnias.*

Componente Conquistas

Este componente representa visualmente as conquistas relacionadas às recompensas a serem oferecidas ao usuário relativos aos seus feitos, além de permitir o compartilhamento ou não de *score*, conquistas e insígnias.

O componente conquistas utiliza os princípios de usabilidade (visibilidade, fácil aprendizado, consistência, padronização da funcionalidade e informação) (seção 2.2).

As tabelas 6.36 e 6.37 descrevem respectivamente, os métodos construtor e *SendScore* disponibilizados pelo componente conquistas. A relação completa dos métodos do componente conquistas são descritos no apêndice C.13.

Os métodos com prefixo *set* e *get* redefinem e retornam respectivamente, as permissões de compartilhamento das conquistas dos usuários. Os métodos *set* interferem diretamente nos métodos com prefixo *send*, pois redefinem a possibilidade de envio das conquistas a outros usuários. Os métodos *get* permitem o acesso as conquistas e permissões de compartilhamento entre o paciente e sua rede de colaboradores.

Os métodos com prefixo *send* são acionados sempre que o usuário desejar compartilhar seus feitos com outros usuários da sua rede de relacionamento (componente gráfico social seção 5.6.3).

A figura 6.24 mostra o fluxo de funcionamento do componente conquistas e seu relacionamento com outros componente de *gamificação*.

Componente integrante dos jogos, as conquistas representam o conjunto de recompensas pela realização das atividades. O componente conquistas desenvolvido permite visualizar as conquistas e definir, que conquistas podem ser compartilhadas.

A possibilidade de reconhecimento das realizações e o compartilhamento dos feitos do usuário tendem a motivá-lo a querer realizar suas atividades com maior perfeição.

Tabela 6.36: *Componente conquistas método construtor*

	Descrição
Nome Método:	*Achievement*
Propósito:	Criar instância do componente conquistas.
Descrição:	Esse método cria uma instância do objeto conquistas.
Assinatura:	*Achievement(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Componente Missões

Esse componente é utilizado para fornecer ao usuário todos os parâmetros de funcionamento de uma atividade a ser realizada, as funções cognitivas trabalhadas, em quantas etapas se divide a atividade, tempo de duração da atividade, participantes, funções de cada participante.

O componente missões utiliza os princípios de usabilidade consistência, compatibilidade, padronização da funcionalidade e da informação) (seção 2.2).

Tabela 6.37: *Componente conquistas método SendScore*

	Descrição
Nome Método:	*SendScore*
Propósito:	Compartilha a pontuação do usuário.
Descrição:	Esse método compartilha a pontuação conquistada pelo usuário durante a realização de uma atividade.
Assinatura:	*<int>sendScore(<Context ctx, Points point>)*
Escopo:	*Public*
Retorno:	*int*
Parâmetros:	*Context ctx, Points point*
ctx	Parâmetro que contêm o contexto do *activity* de uma atividade.
point	Parâmetro que contêm os pontos conquistados pelo usuário.

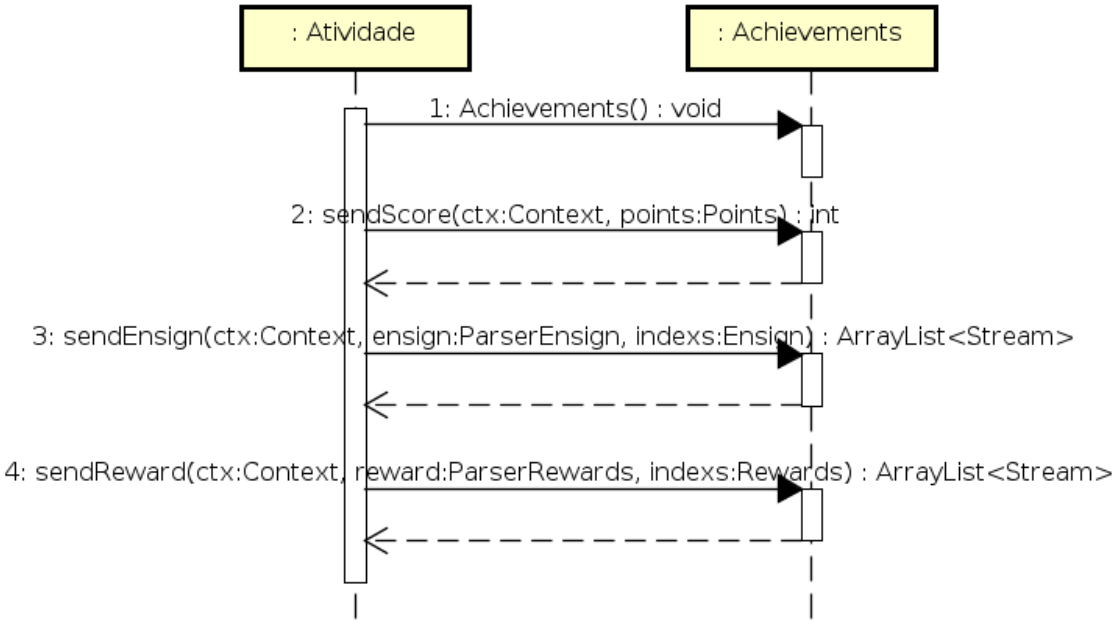

Figura 6.24: *Exemplo de uso do componente conquistas.*

As missões são orientadas por narrativas (embutidas e emergentes seção 5.6.1 item narrativas).

Os métodos construtor e *setMissionObjective* são descritos nas tabelas 6.38 e 6.39. A relação completa de métodos disponibilizados pelo componente missões estão no apêndice C.11.

Os métodos com prefixo *set* e *get* possibilitam a definição e a recuperação da missão das atividades. As narrativas embutidas são passiveis de serem redefinidas pelo método *setMission*, depos da instância das narrativas da atividade.

A figura 6.25 mostra o fluxo de funcionamento do componente missões e seu relacionamento com outros componentes de *gamificação*.

Elemento integrante dos componentes dos jogos, a missão é representada pelo componente missões, define as ações que o usuário deve executar na realização da

atividade.

Esse componente tem a função de orientar as ações do usuário. A orientação ocorre pelas mensagens retornadas ao usuário a cada etapa da atividade em curso.

A cada nova missão a ser cumprida pelo usuário, este pode se sentir desafiado e instigado a superar seus próprios limites.

Tabela 6.38: *Componente missões método construtor*

	Descrição
Nome Método:	*Missions*
Propósito:	Criar instância do componente missões.
Descrição:	Esse método cria uma instância do objeto missões.
Assinatura:	*Missions(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.39: *Componente missões método setMissionObjective*

	Descrição
Nome Método:	*setMissionObjective*
Propósito:	Define o objetivo da atividade.
Descrição:	Esse método define o objetivo da missão a ser realizada dentro de uma atividade pelo usuário.
Assinatura:	*<void>setMissionObjective(<String objective>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*String objective*
objective	Parâmetro que define o texto de orientação sobre o objetivos a serem atingidos na atividade.

Componente Desafios

A função desse componente está ligada ao nível inicial de dificuldade das atividades. Os níveis de dificuldade são definidos conforme as habilidades restantes no usuário definidas no componente habilidades (seção 5.6 item componente habilidades).

Os desafios têm o grau de dificuldade variante aumentando ou diminuindo, conforme os acertos e erros do paciente na realização das atividades. Os níveis de dificuldade das atividades são representados pelo tempo de espera para a resposta das ações, intervalo de tempo entre uma ação e outra, pedidos consecutivos de ajuda e desistências.

As tabelas 6.40 e 6.41 descrevem respectivamente os métodos construtor e *setErrorLimit* disponibilizados pelo componente desafios. A relação completa dos métodos do componente desafios estão descritos no apêndice C.12.

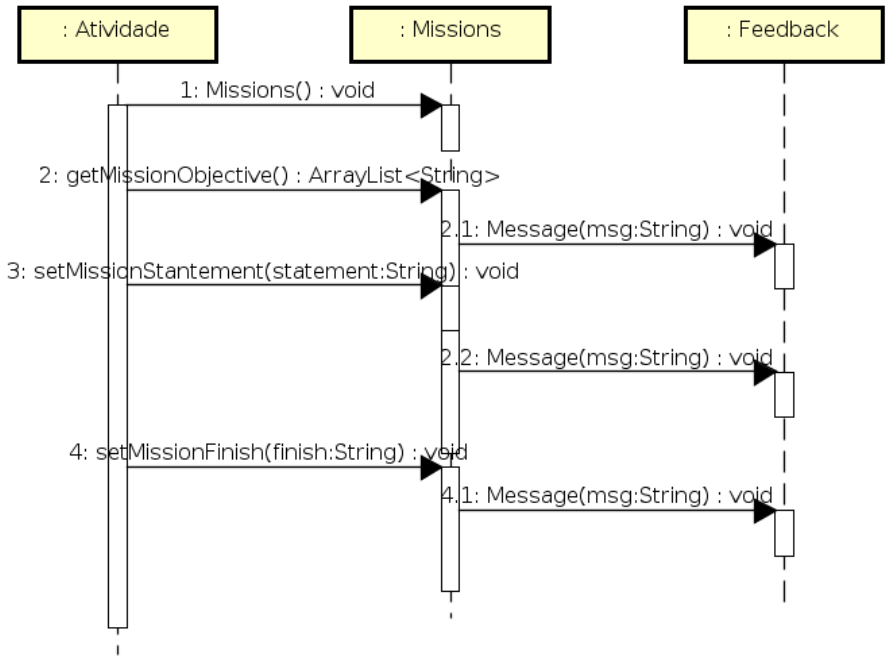

Figura 6.25: *Exemplo de uso do componente missões.*

A figura 6.26 mostra o fluxo de funcionamento do componente desafios e seu relacionamento com outros componentes de *gamificação*.

Integrante da mecânica dos jogos o desafio é representado pelo componente desafios, que traz as barreiras a serem vencidas.

No contexto dessa arquitetura o objetivo dos desafios é manter pelo maior tempo possível o paciente exercitando suas funções cognitivas. Associado com a missão, o desafio pode estimular o usuário a pensar e desenvolver novas estratégias para realização da atividade.

Tabela 6.40: *Componente desafios método construtor*

	Descrição
Nome Método:	*Challenges*
Propósito:	Criar instância do componente desafios.
Descrição:	Esse método cria uma instância do objeto desafios.
Assinatura:	*Challenges(<void>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*void*

Tabela 6.41: *Componente desafios método setErrorlimit*

	Descrição
Nome Método:	*setErrorLimit*
Propósito:	Definir e redefinir o limite de erros permitidos na atividade.
Descrição:	Esse método define e redefine o limite máximo aceitável de erros cometidos pelo usuário dentro da atividade.
Assinatura:	*<void>setErrorlimit(<int erro>)*
Escopo:	*Public*
Retorno:	*void*
Parâmetros:	*int error*
error	Parâmetro contêm o limite aceitável de erros.

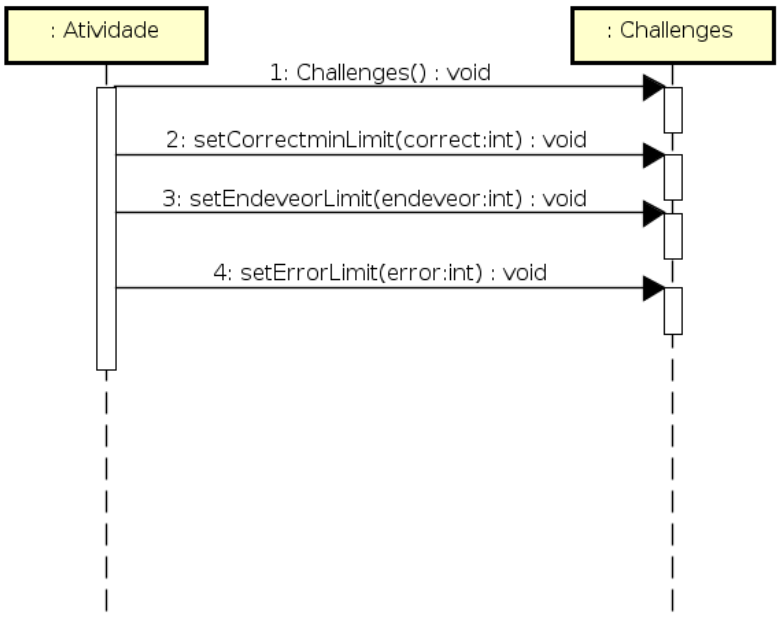

Figura 6.26: *Exemplo de uso do componente desafios.*

CAPÍTULO 7

Ferramentas Empregadas

Neste capítulo são apresentadas as tecnologias utilizadas no desenvolvimento dos módulos de *gamificação* e aplicação teste do estudo de caso.

7.1 Biblioteca *SQLite*

Desenvolvido por D. Richard Hipp em 2000, o *SQLite* consiste em uma biblioteca de *software* que implementa banco de dados relacionais. Uma ferramenta *open source* compacta, o *SQLite* foi desenvolvido para persistir dados gerenciados por aplicações que rodam em dispositivos móveis. Cada aplicativo roda seu próprio banco de dados em um processo *standalone*. Por padrão não é permitido o compartilhamento dos dados do banco entre as aplicações instaladas no dispositivo móvel ou via acesso remoto [63].

Atualmente, o *SQLite* encontra-se na versão 3.0. É uma biblioteca de acesso a banco de dados SQL (*Structured Query Language* - Linguagem de Consulta Estruturada), que suporta diversas tabelas, visualizações, índices e procedimentos, todos reunidos em um único arquivo na memória do dispositivo.

Disponível nas plataformas 32 e 64 bits. O *SQLite* possui uma biblioteca de funcionalidades que pode variar entre 300Kb e 500Kb, conforme a plataforma utilizada. Em execução a exigência de memória do *SQLite* é de 4Kb na pilha de processos e um *heap* de 100Kb, tornando-se ideal para dispositivos com baixa capacidade de armazenamento [72].

A tipagem de dados dinâmica utilizada no *SQLite* está associada ao valor do dado e não a estrutura de um campo. O sistema de tipos dinâmico do SQLite é compatível com os sistemas de tipos estáticos, garantindo às instruções SQL de bancos de dados de tipagem estática funcionem da mesma maneira no SQLite [73].

A tabela 7.1 apresenta os nomes dos tipos de dados comuns em banco de dados SQL tradicionais e a compatibilidade com os tipos do *SQLite*.

Tabela 7.1: *Compatibilidade de tipos SQLite.*

Tipos de Dados Padrão	Tipos de Dados *SQLite*	Definição do Tipo
NULL	NULL	Valor nulo
INT, INTEGER, TINYINT, SMALLINT, MEDIUMINT, BIGINT, UNSIGNED BIG INT, INT2 ,INT8	INTEGER	Valor inteiro sinalizado, armazenado em 1,2, 3, 4, 6 ou 8 bytes
CHARACTER(20) ,VARCHAR(255), VARYING CHARACTER(255), NCHAR(55), NATIVE CHARACTER(70), NVARCHAR(100), TEXT, CLOB	TEXT	Valor *string* armazenado no formato (UTF-8, UTF-16BE ou UTF-16LE)
BLOB (tipo não especificado)	BLOB	Objeto de dados binário
REAL, DOUBLE, DOUBLE PRECISION, FLOAT	REAL	Valor ponto flutuante armazenado 8-byte IEEE
NUMERIC, DECIMAL(10,5), BOOLEAN, DATE, DATETIME	NUMERIC	Converte uma entrada de texto em INTEGER ou REAL

7.2 Plataforma *Android*

A plataforma *Android* caracteriza-se por ser aberta e flexível, com a capacidade de utilização em qualquer dispositivo. Projetada sobre o sistema operacional *Linux*, *Kernel* versão 3.4, o *Android* constitui-se em um poderoso *framework* de aplicações, concebido em quatro camadas de *software* (figura 7.1) [5].

Segundo os desenvolvedores do *Android* [31] a estrutura em camadas adotada flexibiliza o versionamento da plataforma e a incorporação de novos dispositivos. A estrutura em camadas pode ser vista na figura 7.1.

A camada de nível zero: Estão localizados programas que gerenciam a memória dos dispositivos, *drivers de hardware* (câmera, *display* etc), comunicação (*Bluetooth, Wi-fi* etc), configuração de segurança, gerenciamento de energia, gerenciamento de processos.

A camada de nível um: Dois grandes grupos de *software* da plataforma *Android* estão localizados no nível um, as bibliotecas e rotinas de tempo de execução do *Android*. As bibliotecas são responsáveis por definir a forma de tratamento dos mais variados tipos de dados, biblioteca gráfica *OpenGL*, gerenciador de banco de dados*SQLite*, rotinas escritas em C e C++ utilizados pelo sistema operacional e disponibilizada aos desenvolvedores de aplicação.

As rotinas de tempo de execução do *Android* se subdividem em bibliotecas de núcleo Java e a máquina virtual do *Android Dalvik Virtual Machime* (DVM).

A biblioteca de núcleo java propicia o suporte ao desenvolvimento de aplicativos para o *Android*, disponibilizando pacotes com classes escritas na linguagem Java, que serão processadas pela DVM.

As aplicações desenvolvidas são executadas em processos identificados por *IDs - Identities* únicos gerados pelo sistema operacional. No interior de cada processo é instanciada uma cópia da DVM, a aplicação do usuário, os dados e recursos necessários para a execução da aplicação.

A camada de nível dois: Corresponde a camada de *framework* de aplicações, esta camada é responsável pelo gerenciamento das aplicações básicas dos dispositivos. As permissões de acesso aos recursos de *hardware, software* e dados são gerenciadas nessa camada.

Na camada de nível três: Corresponde a camada de aplicação, neste nível da arquitetura encontram-se as aplicações desenvolvidas para os usuários dos dispositivos. Exemplos como *e-mail*, navegadores, aplicativos realizadores de chamadas telefônicas, envio e recebimento de SMS, como tantos outros estão na camada de aplicação.

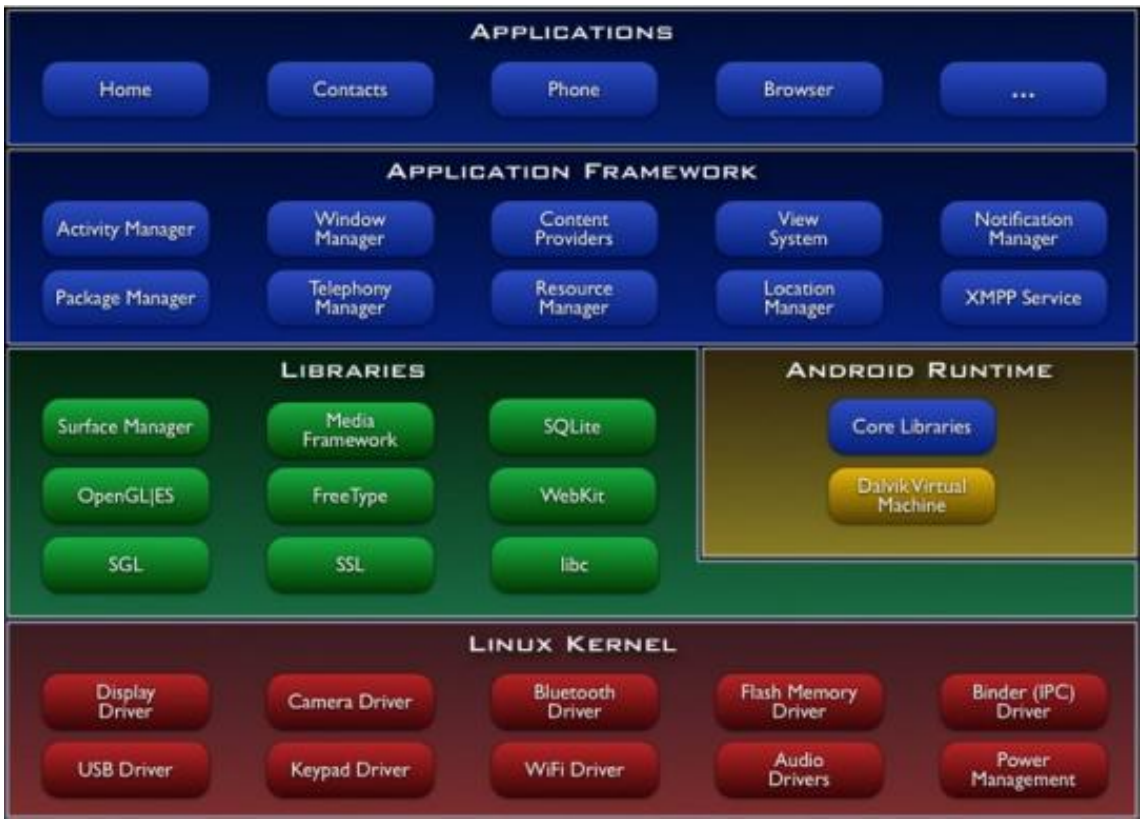

Figura 7.1: *Plataforma Android*

7.2.1 Versionamento da Plataforma *Android*

Desde o surgimento da versão 1.0 do *Android SDK - (Software Development Kit)*, a plataforma *Android* tem sido desenvolvida de forma rápida pelos projetista e desenvolvedores da *Google* [5]. Segundo dados do *Google* [32], atualmente o *Android* encontra-se na versão 6.0 batizada *Marshmallow* e a *Application Programming Interface* (API) nível vinte e três.

A tabela 7.2 mostra dados sobre versionamento da plataforma e o percentual de dispositivos utilizando cada versão da plataforma *Android* recentemente.

No desenvolvimento dos módulos de componentes e da aplicação teste adotaremos a API versão nível 19, versão 4.4 *Kit Kat*, como identificado na tabela 7.2, aproximadamente 80% dos dispositivos ativos utilizam a versão *Kit Kat* ou superior.

Tabela 7.2: *Abrangência do versionamento sobre o número de dispositivos (01-08-2016).*

o	**Versão**	**Apelido**	**API**	**Abrangência**
2.2		*Froyo*	8	0,1%
2.3.3 - 2.3.7		*Gingerbread*	10	1,7%
4.0.3		*Ice Cream Sandwich*	15	0,8%
4.0.4		*Ice Cream Sandwich*	15	0,8%
4.1.x		*Jelly Bean*	16	6.0%
4.2.x		*Jelly Bean*	17	8.3%
4.3		*Jelly Bean*	18	2.4%
4.4		*KitKat*	19	29,2%
5.0		*Lolipop*	21	14,1%
5.1		*Lolipop*	22	21,4%
6.0		*Marshmallow*	23	15,2%

7.3 Padrões de Projeto

Originado da arquitetura, o termo *Padrões* possou a ser utilizado na Engenharia de *Software* após serem observados e propostos Erich Gamma e seus colegas em 1995, com a publicação de um catálogo contendo vinte e três padrões de projeto conhecidos como catálogo GoF - (*Gang of Four*) [40].

Os padrões de projeto podem ser definidos como um conjunto de soluções para os problemas frequentes no desenvolvimento de *software* que permitem serem resolvidos pelas mesma soluções utilizando caminhos diferentes [38, 40].

A identificação de qual padrão de projeto aplicar na solução de um problema, segundo Fowler [38], ocorre pela observação e a experiência com a resolução de problemas semelhantes, que ocorrem durante o desenvolvimento de *software*.

Partindo deste princípio, Fowler e Freeman [38, 39] observaram o surgimento e a consolidação de um vocabulário comum entre os grupos de desenvolvedores. A utilização de um vocabulário comum possibilita uma comunicação mais eficiente, permitindo uma compreensão imediata e precisa do problema, e as possíveis soluções propostas.

Os padrões de projeto segundo Gamma e colaboradores [40] podem ser classificados a partir de critérios de finalidade e escopo. A finalidade define o que o padrão faz: um padrão pode ter finalidade de criação, estrutural ou comportamental. O escopo direciona o padrão a relacionamentos entre classes ou objetos.

A tabela 7.3 mostra a classificação dos vinte e três padrões de projeto propostos em [40], relativos a escopo e finalidade. Em relação ao escopo a tabela 7.3 dois conjuntos de padrões um voltado a classes e outro voltado a objetos. A finalidade divide os padrões múltiplos em três conjuntos: criação, estrutural e comportamental.

Nos padrões de criação no escopo de classe, o processo de criação de objetos é realizado nas subclasses. No de escopo de objeto o processo é postergado para outros objetos. Os padrões estruturais no escopo de classe utilizam a herança para compor classes, enquanto os de escopo de objeto descrevem estruturas para relacionar objetos. E os padrões comportamentais no escopo de classe utilizam a herança para descrever algoritmos e fluxos de controle, enquanto os voltados para objetos descrevem como um grupo de objetos se relaciona e compartilha a execução de tarefas [40].

Tabela 7.3: *Padrões de Projeto GoF [40].*

		Propósito		
		Criação	Estrutural	Comportamental
Escopo	Classe	*Factory Method*	*Adapter*	*Interpreter* *Template Method*
	Objeto	*Abstract Factory* *Builder* *Prototype* *Singleton*	*Adapter* *Brige* *Composite* *Decorator* *Facade* *Flyweight* *Proxy*	*Chain Responsibility* *Command* *Iterator* *Mediador* *Memento* *Observer* *State* *Strategy* *Visitor*

7.3.1 Padrão *Abstract Factory*

O padrão de projeto *Abstract Factory* fornece uma *interface* de criação para famílias de objetos relacionados ou dependentes sem a necessidade de explicitar suas classes concretas [39].

O grupo de objetos (família) criados a partir do padrão *Abstract Factory* conforme define Gamma e colaboradores [40], geram uma conjunto de objetos que se relacionam para atingir um objetivo.

A responsabilidade de implementação dos objetos são encapsuladas e isoladas das aplicações cliente, que manipula as instâncias através de *interfaces* abstratas.

Segundo Freeman e colaboradores [39], o baixo acoplamento entre os objetos possibilita a alteração das classes-produto sem interferir no código da aplicação e pouca dependência de implementações concretas.

O uso do padrão *Abstract Factory* é recomendado quando forem identificadas características, como as listadas a seguir:

- Existir a independência na forma de criação dos objetos (produtos);
- Configurar os sistemas na forma de uma família de diversos objetos (produtos); e
- Garantir o uso conjunto dos objetos pertencentes ao grupo (família).

Porém, apesar das vantagens de se utilizar o método *Abstract Factory* [40], o padrão dificulta o suporte a novos objetos, devido a fixação do conjunto de objetos que podem ser criados. Suportar novos tipos de objeto exige a extensão da *interface* resposável pela fabricação dos objetos (produtos), pois envolve a mudança da classe *AbstractFactory* e todas as subclasses.

CAPÍTULO 8

Exemplos de Cenários de Utilização dos Elementos de Gamificação

Neste capitulo serão apresentados alguns cenários de utilização das atividades *gamificadas*, voltadas para pacientes diagnosticados com a doença de Alzheimer nos níveis inicial a moderado.

8.1 Descrição do Paciente

O perfil de paciente considerado nessa dissertação está baseado no estudo de caso apresentado por Ávila [9]. A paciente é uma senhora com idade 73 anos, aposentada, viúva e quando estava na ativa desempenhava as funções de secretária bilíngue e professora de piano. A paciente vive em sua própria residência acompanhada de um cuidador.

Uma pessoa considerada com alto nível de escolaridade, após exames de sangue, eletroencefalograma e aplicação do teste Mini Exame do Estado Mental (MEEM) (seção 3.1) foi diagnosticada com doença de Alzheimer na fase inicial.

A paciente tem apresentado lapsos de memória recente, desorientação espaço temporal em alguns momentos e desatenção na realização de atividades da vida diária tais como: pagamento de contas, esquecimento de compromissos, objetos deixados em locais não apropriados.

A cuidadora lhe faz companhia seis dias por semana, de segunda a sábado, nos finais de semana um familiar acompanha a paciente, auxilia nas atividades domésticas e a acompanha em passeios e viagens. A descrição detalhada dos perfis dos pacientes encontra-se no apêndice D.

8.2 Atividades do Cotidiano da Aplicação 01

As atividades diárias da paciente iniciam com a compra do pão na padaria próxima à residência. O café da manhã é preparado pela paciente, com supervisão da sua cuidadora.

Após tomar o café da manhã e a organização da cozinha, a paciente recolhe o jornal local deixado em sua porta. As noticias são lidas em voz alta pela paciente, para serem partilhadas pela cuidadora.

Ao término da leitura do jornal, a paciente gosta de ouvir música clássica, relembrar dos bailes e viagens realizadas com seu esposo e cuidar de seu orquidário.

O almoço é realizado no restaurante próximo à residência da paciente. Ambas se dirigem ao restaurante a pé. Após o almoço a paciente e a cuidadora passeiam pelas imediações visitando a praça e o museu da cidade.

O jantar é preparado pela cuidadora e servido nas primeiras horas da noite, após assistir ao telejornal de sua preferência a paciente se recolhe a seus aposentos.

Aos finais de semana, parentes e amigos costumam visitar a paciente, que são recepcionados com a mesa farta de bolos, biscoitos feitos pela paciente e sua cuidadora.

8.3 Atividade Terapêutica da Aplicação 01

A atividade terapêutica desenvolvida está fundamentada na terapia de reminiscência (seção 3.2), cujo objetivo é trabalhar a capacidade de orientação espaço temporal da paciente, e a realização dos seus afazeres diários. A aplicação que representa a atividade é composta por uma sequência de imagens referentes às atividades diárias (Preparação do café da manhã, o deslocamento da residência até à padaria de costume, preparação do almoço etc), de pessoas de convívio da paciente, lugares por onde viajou com pessoas de seu convívio.

Os componentes de *gamificação* utilizados nesta atividade estão descritos na tabela 8.1.

8.3.1 Artefatos de Entrada Aplicação 01

Os artefatos de entrada para a construção da aplicação consistem de fotografias, vídeos e textos.

As fotografias e vídeos representam pessoas, viagens, festas e eventos vividos pela paciente, parentes e amigos. As imagens são carregadas em uma lista, contendo fotografias e vídeos, juntamente com dados que descrevem os acontecimentos e quem são as pessoas mostradas nas imagens. Os eventos também podem ser classificados como a realização de atividades da vida diária, preparar refeições, realizar compras entre outras atividades.

Um conjunto de fotografias ligadas às atividades da vida diária é utilizado para auxiliar o resgate e memorização do processo de preparação de alimentos, horário

das refeições, familiares e amigos que costumam ou costumavam apreciar as refeições reunidos, gerenciamento e organização da casa.

O formato dos arquivos de vídeo mantidos e utilizados na atividade têm a extensão *mp4* e o formato das fotografias utilizados na atividade têm a extensão *jpg*.

As imagens podem ser inseridas pelo cuidador na aplicação de modo a propiciar o encadeamento das lembranças da paciente até o tempo presente, a atenção, linguagem, funções executivas.

8.4 Descrição da Aplicação 01

A aplicação está dividida na área do paciente e na área do cuidador. A área do paciente é acessada pela *interface* principal da aplicação, onde paciente realiza a atividade programada pelo cuidador. A figura 8.1 apresenta a *interface* principal da da aplicação de uma aplicação exemplo gerada com componentes de *gamificação* da arquitetura proposta;

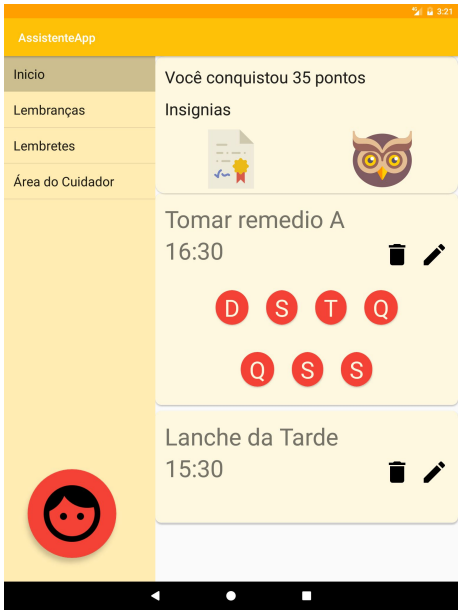

Figura 8.1: *Interface Principal da Aplicação Exemplo.*

A área do cuidador é acessada pela autenticação do nome do cuidador e senha. Na figura 8.2 o cuidador visualiza os dados relacionados a evolução do MEEM e dados do paciente.

Na segunda etapa da criação da atividade, o cuidador juntamente com o desenvolvedor dentro do ambiente de autoria desenvolve exercícios para estimulação cognitiva relativas às imagens e às habilidades cognitivas a serem trabalhadas.

As ações (exercícios) são pares pergunta e resposta respondidos pela paciente, pela seleção de uma única resposta correta e a confirmação, seleção de vários itens de

resposta e a confirmação, a resposta textual e a confirmação. A confirmação é realizada com a pressão do botão responder.

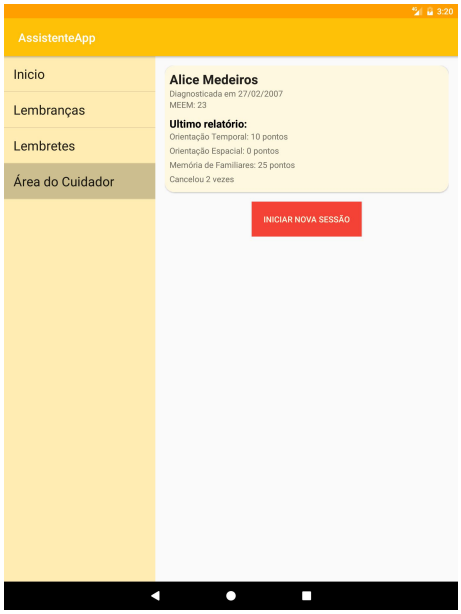

Figura 8.2: *Área do Cuidador.*

As questões são criadas pelo cuidador ou escolhidas de uma lista de questões já predefinidas no ambiente de autoria. A cada questão formulada, há a necessidade de salvá-la pressionando o botão salvar.

8.4.1 Formas de Interação da Atividade com o Paciente

A interação entre paciente e aplicação ocorre em dois sentidos: aplicação para paciente e paciente para aplicação.

No sentido aplicação paciente, a comunicação ocorre sempre que existir a necessidade de explicar o funcionamento da atividade, parabenizar pelos feitos da paciente, incentivar a participação e a continuidade na realização das atividades.

Na figura 8.3 pode ser visualizado os componentes *avatar*, *feedback* e narrativa, os quais são representantes desse sentido de comunicação.

No sentido paciente aplicação, a comunicação ocorre quando as ações propostas pela atividade são realizadas pela paciente.

As ações realizadas pela paciente consistem em responder as questões elaboradas na área do cuidador. As questões estão relacionadas aos artefatos de entrada, também escolhidos pelo desenvolvedor e cuidador dentro da área de autoria.

A cada questão apresentada na fase de *quiz*, a paciente tem a possibilidade de selecionar a resposta e pressionar o botão responder, desistir de responder a questão

pressionando o botão desistir, ou solicitar a explicação da ação pressionando o botão explicar.

Figura 8.3: *Atuação dos componentes avatar, narrativas e feedback.*

A cada resposta da paciente e pressão de um dos três botões da *interface* (responder, desistir e explicar), a aplicação se comunica com a paciente dando retorno audível, textual e visual as respostas.

8.4.2 Exercícios Propostos na Atividade

Os exercícios realizados pela paciente constituem na resposta dos questionamentos relativos às imagens, vídeos e execução de atividades ligadas ao dia a dia. A interação pode ocorrer de forma a permitir múltiplas escolhas às respostas, uma única escolha como resposta ou resposta textual.

Fluxo de Execução da Atividade

Fase 1:

- passo 1: Os objetivos da atividade são apresentados e explicados na forma textual e audível a paciente; e
- passo 2: imagens na forma de fotografias e vídeos contendo pessoas queridas a pacientes são apresentadas.

Fase 2:

- passo 1: Os objetivos da atividade são apresentados e explicados na forma textual e audível a paciente; e

- passo 2: Após visitar e revisitar as imagens das pessoas queridas, a paciente responda questões ligadas as suas preferências e das pessoas vistas, relacionada com alguma atividade diária (exemplo: preparo do café da manhã).

Fase 3:

- passo 1: Os objetivos da atividade são apresentados e explicados na forma textual e audível à paciente;
- passo 2: Questões relacionadas à localização temporal e espacial da padaria (proximidade de quais locais, horário funcionamento); e
- posso 3: Perguntas ligadas aos produtos comprados na padaria, realização de cálculo relativo ao valor da compra, troco.

Fase 4:

- passo 1: Os objetivos da atividade são apresentados e explicados na forma textual e audível a paciente; e
- passo 2: Questões relacionadas ao preparo do café da manhã (itens utilizados, tempo de preparo de cada item do café, ordem de preparação etc).

Ao desenvolver a atividade o desenvolvedor cria uma atividade principal, que engloba todas as subatividades e as ações de cada subatividade 6.3.

A atividade principal pode ser composta apenas de ações, não sendo dividida em subatividades. Em cada atividade ou subatividade é permitido a criação e utilização de apenas uma narrativa embutida.

O desenvolvedor, por meio da ferramenta de autoria escolhe:

- O tipo de atividade (principal ou subatividade);
- O nome da atividade;
- A narrativa embutida;
- O ícone que representa o avatar;
- O *feedback* é audível ou não audível;
- As Regras para obtenção:
  - Da pontuação para obtenção das insígnias; e
  - Da duração (tempo) da atividade.
- Os ícones representantes das insígnias (presentes na galeria de imagens da aplicação ou na galeria de imagens do dispositivo); e
- A pontuação do MEEM a ser atingida pelo paciente utilizada na progressão.

Na criação das ações de cada atividade ou subatividade, o desenvolvedor baseado nos dados fornecidos pelo cuidador/familiar escolhe:

- As narrativas emergentes para as possibilidades de acerto das questões, desistência de uma questão, mensagem de auxílio;
- As regras para obtenção:
  - Da pontuação para cada acerto de questão na primeira tentativa e nas outras tentativas;
  - Do tempo de espera para a resposta do paciente (níveis);e
  - Da bonificação pela conclusão das ações antes do tempo estipulado.

8.5 Modelando a Aplicação Móvel de Apoio a Terapia de Reminiscência Fase 1

Nesta primeira fase, o cuidador pode definir os artefatos de entrada, as *interfaces* da aplicação, os elementos de *gamificação*, exercícios para o paciente e os relatórios produzidos aplicação.

8.5.1 Etapa 1: Escolha dos Artefatos de Entrada

Nesta etapa, o desenvolvedor auxiliado pelo paciente e cuidador escolhem os artefatos (imagens, vídeos, textos e músicas) utilizados durante a terapia de reminiscência. Os artefatos devem ser fornecidos previamente por familiares e amigos do paciente.

Os artefatos escolhidos são categorizados em listas, sendo uma lista para os vídeos, uma lista para as imagens, uma lista para textos, uma lista para músicas e sons.

A cada lista de artefatos criada é necessário confirmar a criação, pressionando o botão *criar lista*. A ação de criação da lista gera uma sequência de *tags* XML, mostrado no código XML 8.1 a seguir.

A área de inserção de artefatos de entrada é demarcada pelas *tags* <*Input*></*Input*>. No interior da *tag input* estão listadas as *tags* representantes dos artefatos utilizados na aplicação terapêutica.

A *tag* <*Video*> é composta por dois atributos obrigatórios *id* e *src*. O atributo *id* mantém a ordem de inserção dos artefatos, sendo a ordem iniciada pelo índice 1. O atributo *src* está associado ao nome do arquivo de vídeo integrante da lista.

As *tags* <*Photo*>, <*Information*> e <*Sound*> possuem os mesmos atributos da *tag* <*Video*> e seguem a mesma lógica de funcionamento.

Código 8.1 Fragmento XML dos Artefatos de Entrada

```
1 <Input>
2         <Photo id="1" src="filho.jpg"/>
3         <Video id="1" src="casamento.mp4"/>
4         <Sound id="1" src="piano.mp3"/>
5 </Input>
```

8.5.2 Etapa 2: Escolha das Interfaces da Aplicação

Nesta etapa, o desenvolvedor, cuidador e paciente de forma conjunta podem escolher as *interfaces*, as quais serão visualizados os artefatos de entrada selecionados na etapa 1.

O número de *interfaces* utilizadas depende de quantas *activitys* a aplicação irá conter. Uma *activity* consiste em uma tela que permite ao usuário interagir com a aplicação (fotografar, enviar e-mail etc). Cada atividade possui uma janela que exibe a *interface* do usuário [31].

As *interfaces* são escolhidas pela seleção de múltiplos modelos predefinidos de *layouts*, para as janelas de interação com o usuário. Após a seleção o botão *salvar layouts* é pressionado.

8.5.3 Etapa 3: Escolha dos Elementos de *Gamificação* da Aplicação

Após a definição das *interfaces* da aplicação, os componentes de *gamificação* e suas configurações são definidos e inseridos na aplicação.

A aplicação teste é composta por 9 componentes de *gamificação* escolhidos pelo cuidador e inseridos pelo desenvolvedor.

A cada componente de *gamificação* selecionado é necessária a inserção no arquivo XML é realizada pressionando o botão *inserir componente*.

Na sequência é exemplificado o processo de inserção e configuração dos componentes de *gamificação* da aplicação de apoio a terapia de reminiscência.

O cuidador tem a possibilidade de selecionar os componentes de *gamificação* de uma lista predefinida, para o projeto da aplicação *gamificada*. Após a escolha, a configuração dos atributos e salvamento dos componentes, uma sequência de *tags* XML é gerada.

A área de inserção dos componentes de *gamificação* é demarcada pelo par de *tags* <*Gamification*><*/Gamification*>. No interior da *tag gamification* estão listadas as *tags* representantes dos componentes de *gamificação* utilizados na aplicação terapêutica.

O componente narrativa é representado por dois pares de *tag*. O primeiro par de *tags* <*Emb_Narrative*></*Emb_Narrative*> representa as narrativas embarcadas (seção 5.6 narrativa) e entre o par de *tags* é inserido o texto relativo às informações de funcionamento da atividade contida na aplicação terapêutica.

O segundo par de *tags* <*Eme_Narrative*></*Eme_Narrative*> representa as narrativas emergentes (seção 5.6) e os atributos *id* e *type* são obrigatórios. O atributo *id* mantém a ordem de inserção dos artefatos, sendo a ordem iniciada pelo índice número 01. O atributo *type* identifica os tipos de narrativas emergentes retornadas como *feedback* para o paciente (tabela 6.6). Entre o par de *tags* é inserido o texto relativo as mensagens utilizadas, como resposta as interações do paciente com a atividade. Um exemplo do código XML gerado para as narrativas pode ser visto na sequência de código 8.2.

Código 8.2 Fragmento XML das Narrativas

```
1  <Emb_Narrative>
2          <Message>Narrativa Embarcada</Message>
3  </Emb_Narrative>
4  <Eme_Narrative id="1" type="1">
5          <Message>Muito Bem, Meus Parabens!</Message>
6  </Eme_Narrative>
7  <Eme_Narrative id="2" type="1">
8          <Message>Congratulacoes, hoje seu
9          desempenho esta excelente.</Message>
10 </Eme_Narrative>
11
```

A *tag* <*Feedback*> é composta por dois atributos obrigatórios *id* e *visual*. O atributo *id* mantém a ordem de inserção dos artefatos, sendo a ordem iniciada pelo índice número 01. O atributo *visual* indica se o *feedback* a ser retornado ao paciente é visual (*yes*) ou não visual (*no*). O fragmento de código 8.3 indica *feedback*, por meio de ícones para o paciente.

Código 8.3 Fragmento XML do *Feedback*

```
1  <Feedback id="1" visual="yes">
2          <Message>feed01.png</Message>
3  </Feedback>
```

Um exemplo da atuação dos componentes de *gamificação feedback* e narrativa são destacados pela elípse visualizado na figura 8.4.

A *tag* <*Avatar*> contém dois atributos requeridos *src* e *time*. O atributo *src* está associado ao nome do arquivo que representa a imagem do *avatar*.

Figura 8.4: *Feedback e narrativa atuando na motivação do paciente.*

O atributo *time* representa o tempo de espera inicial da exibição das mensagens textuais e visuais no *display* do dispositivo. Visto no fragmento de código 8.4 e a atuação do componente avatar é mostrado na figura 8.5

Código 8.4 `Fragmento XML do Avatar`

```
1 <Avatar src="face.png"/>
```

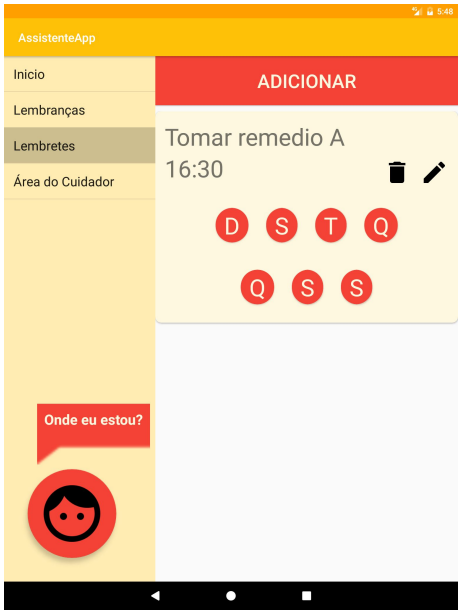

Figura 8.5: *Avatar atuando na motivação do paciente.*

A inserção de regras permite ao cuidador controlar os itens pontuação recebida a cada erro ou acerto, mudança de nível de dificuldade, momentos de recebimento de recompensas.

A determinação do conjunto de regras é realizada pela escolha de um ou vários itens das regras e a atribuição dos valores para cada item mostrados no código XML 8.5.

O exemplo do código XML 8.5 exemplifica a pontuação atribuída aos acerto a uma ação correta do paciente, a pontuação para a mudança de nível e o tempo de espera às respostas do paciente durante as ações da atividade, quando oferecer recompensas pelas ações, pontuação para oferecimento de insígnias, permitir nova tentativa em caso de erro, solicitação de auxílio e taxa de variação de tempo.

As *tags* <*Rules*> e <*/Rules*>, respectivamente abrem e fecham o bloco XML de definição das regras. Após a escolha dos itens de regras o cuidador pressiona o botão *Salvar Regras*.

Código 8.5 Fragmento XML das Regras

```
<Rules>
    <Points>
       <Correct value="10"/>
       <First_Error value="5"/>
    </Points>
    <Level id="1" value="100" time="30"/>
    <Level id="2" value="200" time="20"/>
    <Offer_Reward>
        <Abort>2</Abort>
        <Abort_Alternate>3</Abort_Alternate>
        <Error_Action>2</Error_Action>
        <Correct_Action>3</Correct_Action>
   </Offer_Reward>
   <Offer_Ensign value="100"/>
   <Action_Replay value="yes"/>
   <Action_Help value="yes"/>
   <Question_Time time="60"/>
   <Time_Variation rate="0.1"/>
</Rules>
```

O componente pontos permite ao cuidador definir a pontuação atribuída as realizações do paciente. A pontuação é informada por meio de campos texto e inserida no código XML, depois de pressionado o botão *inserir pontuação*. Visto no fragmento de código 8.6.

A aplicação utiliza pontuações diferentes para cada resultado das ações do paciente (exemplo: Pontos atribuídos quando paciente acerta um exercício previsto na atividade terapêutica, erra um exercício pela primeira vez).

O par de *tags* <*Points*></*Points*> agrupa *tags* que definem a pontuação para situações de erro aplicações acertos <*Correct*></*Correct*> e <*First_Error*></*First_Error*>.

Código 8.6 Fragmento XML da Definição dos Pontos

```
1 <Points>
2     <Correct value="10"/>
3     <First_Error value="5"/>
4 </Points>
```

As insígnias oferecidas ao paciente durante a execução da atividade são selecionadas de uma lista de ícones já definida. A cada insígnia escolhida, a pontuação para oferecimento ao paciente é determinada. Após a configuração o botão *salvar insígnia* deve ser pressionado. O código XML referente à inserção das insígnias é apresentado no código XML contido em 8.7

As *tag* <*Insign*> contém os atributos *id* e *src*, para identificação e representação visual da insígnia respectivamente.

O nível de dificuldade para o recebimento das insígnias é definido no XML pela *tag* <*Offer_Ensign*>, através do atributo *value*.

Código 8.7 Fragmento XML da Definição das Insígnias

```
1 <Rules>
2      <Offer_Ensign value="100"/>
3 </Rules>
4 <Ensign id="1" src="flower01.png"/>
5 <Ensign id="2" src="flower02.png"/>
```

Na figura 8.6 são exemplificados a utilização dos componetes insígnias e pontos na aplicação, os quais foram atribuídos ao paciente pelo seu desempenho. Os níveis de dificuldade são definidos pelo cuidador, através da atribuição da pontuação para mudança de nível e o tempo de espera para realização das ações relativas a atividade. O código XML relativo a definição da dificuldade dos níveis encontram-se na sequência XML 8.8.

A *tag* <*Level*> informa o número do nível, a pontuação necessária para a mudança do grau de dificuldade das aplicações e o tempo para realização das ações, pelos atributos *id*, *value* e *time* respectivamente. Após definidas as configurações o botão *salvar níveis* deve ser pressionado.

Código 8.8 Fragmento XML da Definição dos Níveis

```
1 <Level id="1" value="100" time="30"/>
2 <Level id="2" value="200" time="20"/>
```

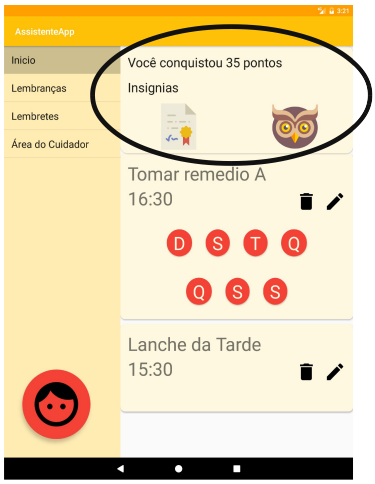

Figura 8.6: *Componentes pontos e insígnias utilizados na aplicação.*

O componente progressão calcula e mantêm dados relativos ao mini exame de estado mental do paciente, realizado durante a execução da aplicação. O valor inicial do MEEM é inserido pelo cuidador e registrado através do botão *salvar progressão*. A *tag* *<Progress>* é inserida no código XML e o valor do atributo meem é definido. O XML da progressão é mostrado a seguir:

Código 8.9 `Fragmento XML da Definição do Nível de Progressão da DA`

```
1    <Progress meem="23"/>
```

As recompensas são uma forma de reconhecimento do esforço e o possível êxito na realização das atividades desenvolvidas pelo paciente. As recompensas ajudam a motivar o paciente na realização das atividades.

O componente *Reward* mantém a lista de recompensas passíveis de serem oferecidas ao paciente como forma de reconhecimento e motivação. O par de *tags* *<Reward></Reward>* é gerada após o desenvolvedor escolher imagens e textos motivacionais. O salvamento das recompensas deve ser feita uma a uma pressionando o botão *salvar recompensas*. O XML relativo as recompensas pode ser visto no fragmento de código 8.10 a seguir:

No interior da *tag* *<Reward>* podem ser utilizados os pares de *tags* *<Text></Text>* e *<Image></Image>* definindo textos e imagens de motivação respectivamente.

Código 8.10 Fragmento XML da Definição das Regras para Recompensar o Paciente

```
<Rules>
        <Offer_Reward>
                <Abort>2</Abort>
                <Abort_Alternate>3</Abort_Alternate>
                <Error_Action>2</Error_Action>
                <Correct_Action>3</Correct_Action>
        </Offer_Reward>
</Rules>
<Reward id ="1">
    <Text>Muito bem</Text>
    <Image src="recompensa01.png"/>
</Reward>
<Reward id="2">
   <Text>Bom Trabalho Continue Tentando</Text>
   <Image src="recompensa02.png"/>
</Reward>

```

O desenvolvedor precisa definir as regras para a aplicação oferecer recompensas ao paciente. Após definir cada recompensa, também são definidos os casos para oferecimento de recompensas (nível de acerto, erros frequentes (para evitar a desistência), desistência (para estimular o retorno às atividades). Com todas as regras definidas para oferecimento de recompensas, o botão *salvar recompensas* deve ser pressionado.

As regras geradas são representadas pelos pares de *tags* *<Abort></Abort>*, *<Abort_Alternate></Abort_Alternate>*, *<Error_Action></Error_Action>* e *<Correct_ACtion></Correct_Action>*.

8.5.4 Etapa 3: Relatórios

As formas de relatórios previstos são direcionadas para o paciente e para o cuidador. O relatório do paciente apresenta dados sobre o rendimento relativos a execução das atividades.

Os dados são dizem respeito à da pontução atingida, insígnias, conquistas alcançadas, erros e acertos na execução das atividades.

O relatório do cuidador apresenta uma evolução das habilidades cognitivas, calculadas pelo índice MEEM total e a evolução individual de cada função cognitiva.

Durante o processo de criação da aplicação, o cuidador pode escolher entre três opções de relatório: apenas cuidador, paciente, e cuidador e paciente.

A *tag* <*Report*> contém o atributo *user*, que pode assumir 1) cuidador, 2) paciente, 3) cuidador e paciente, após a escolha de uma das opções o desenvolvedor pressiona o botão *salvar relatório*. O código XML visto em 8.11 exemplifica, que no momento da geração da aplicação, o relatório gerado será direcionado para o cuidador/familiar. Ao escolher a opção número 1 será gerado o relatório com dados de identificação do paciente (nome, idade, data do diagnóstico), dados do exame de estado mental(orientação temporal, orientação espacial, memória, cálculo e linguagem) da sessão anterior, dados do exame de estado mental da sessão atual e data atual.

Código 8.11 Fragmento XML para Emissão de Relatório

```
1  <Report user = "1"/>
```

8.6 Geração do Código da Aplicação de Terapia de Reminiscência: Fase 2

Nesta fase o desenvolvedor auxiliado pela ferramenta de autoria constroi a atividade XML da aplicação do usuário final (paciente e cuidador). O mapeamento das aplicações criadas com a arquitetura de *gamificação* é visto na figura 8.7. Após gerado o arquivo atividade xml o artefato de *software parserxml* identifica as classes referentes aos componentes de *gamificação* e os insere no projeto da aplicação do usuário final.

As classes representante das *interfaces* da aplicação do usuário final estão contidas no repositório *interface do usuário* e selecionadas pelo *núcleo integrador*, com base no xml construído pelo desenvolvedor e contido no arquivo *atividade xml*.

A atividade fonte gerada contempla a declaração dos objetos das classes dos componentes de *gamificação* , classes das *interfaces*, tabelas para o armazenamento de dados e evolução do paciente.

Os códigos fonte do projeto da aplicação final são empacotados para importação do ambiente de desenvolvimento *Android Studio*.

8.6.1 Hierarquia do Projeto da Aplicação

O processo de geração de código constrói a estrutura hirárquica de todo o projeto, que é compilado pela plataforma de desenvolvimento do *Android Studio*.

O projeto da aplicação desenvolvida é inserido no ambiente de desenvolvimento do *Android Studio* a partir da pasta *app*. Abaixo da pasta *app* são geradas as seguintes subpastas:

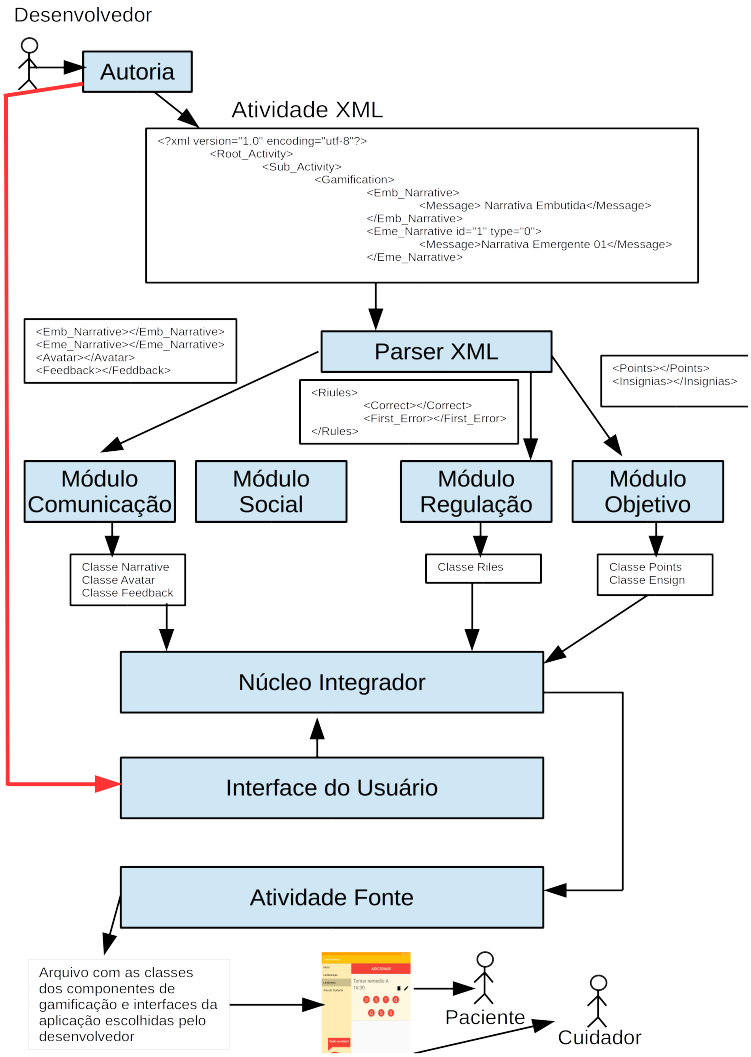

Figura 8.7: *Exemplo contrução da Aplicação do Usuário Final.*

- **Pasta *manifest*:** Esta pasta guarda as configurações do aplicativo no arquivo *AndroidManifest.xml*. O arquivo *AndroidManifest.xml* define as permissões de comunicação, *activity* principal, permissão de gravação etc;
- **Pasta *java*:** Contém o código fonte java da aplicação e o código para teste da aplicação. O projeto pode conter diversos arquivos com a extensão java representando as classes que compõe a aplicação, sendo pelo menos o arquivo fonte da *activity* principal (exemplo: Ativdadeprincipal.java);
- **Pasta *assets*:** Local onde ficam armazenados os arquivos de banco de dados, mídia usadas na aplicação, um exemplo os artefatos de entrada utilizados para realização da terapia de reminiscência; e
- **Pasta *res*:** Esta pasta é responsável por manter os recursos utilizados para o funcionamento da aplicação. A pasta *res* se subdivide em diversas pastas,

drawable (ícones, imagens da aplicação), *layouts* (arquivos XML que representam as *interfaces* de usário da aplicação), *menus* (XML dos *menus*, se existirem),*mipmap/ic_launcher.png* (ícone da aplicação em várias resoluções), *values* (*color.xml* cores da aplicação e seus componentes, *string.xml* rótulos utilizados na aplicação, *values/dimens.xml* dimensões e borda dos componentes de orgamização visual da *interface*, *styles.xml* estilo de fundo de janela, barras de título da aplicação).

8.7 Compilação e Instalação da Aplicação de Terapia de Reminiscência: Fase 3

A compilação da aplicação móvel pode ser direcionada para dispositivos virtuais configurados no emulador da plataforma, ou transmitidos para dispositivos reais previamente plugados, via porta de comunicação *Universal Serial Bus* (USB).

Na compilação para o emulador é preciso pressionar o botão *run* no ambiente da plataforma do *Android Studio* e escolher o dispositivo virtual. Esse processo de escolha só ocorre a primeira vez, quando o emulador do dispositivo deve ser carregado.

A compilação e instalação do código *apk* (pacote da aplicação móvel do *Android*) necessita de permissão de desenvolvedor no dispositivo móvel (a forma de ativar a permissão de desenvolvedor muda conforme o modelo do dispositivo e fabricante). Depois da concessão da permissão, pressiona-se o botão *run* no ambiente da plataforma de desenvolvimento e a compilação e instalação da aplicação é realizada.

8.8 Atividades do Cotidiano da Aplicação 2

A atividade do cotidiano está relacionada à ida ao supermercado, para realização das compras de gêneros alimentícios e de higiene pessoal e da residência do paciente. Com a colaboração do cuidador, o paciente seção 8.1 cria a lista de compras a ser trazida do supermercado e a registra na atividade.

Em outro instante o paciente se dirige ao supermercado com seu cuidador, para a realização da compra dos produtos definidos anteriormente.

No retorno à sua residência o paciente supervisionado pelo cuidador guarda os produtos comprados no seu devido lugar.

8.9 Atividade Terapêutica da Aplicação 2

A atividade terapêutica trabalha a função da memória e a orientação espacial do paciente. A atividade é constituída de duas tarefas. A primeira tarefa consiste na preparação da lista de produtos a serem trazidos do supermercado, a identificação do local onde serão realizadas as compras. A preparação da lista ocorre na residência do paciente, com a colaboração do seu cuidador.

A segunda tarefa integrante da atividade consiste na compra dos produtos definidos na primeira tarefa.

8.9.1 Detalhamento da Primeira Tarefa

A tarefa se inicia solicitando ao paciente a inserção do nome e a localização do supermercado, a data e o horário que serão feitas as compras.

Na sequência é iniciada a criação da lista de compras, através da escolha de uma série de produtos cadastrados na aplicação. Os produtos estão classificados por gênero (higiene pessoal, higiene da casa, café da manhã, almoço, lanche da tarde, jantar) e são escolhidos pelo paciente dentro de cada categoria, uma após a outra.

A lista de compras permite a inserção da quantidade de cada produto a ser adquirido.

8.9.2 Detalhamento da Segunda Tarefa

Na segunda fase da atividade, com a aplicação aberta no modo *ida as compras*, os produtos classificados por gênero são listados e o paciente deve colocá-los no carrinho e marcá-los como adquiridos ou não encontrados. No final de cada categoria de produtos comprados deve ser salva.

Os componentes de *gamificação* utilizados na atividade proposta estão descritos na tabela 8.4.

8.9.3 Artefatos de Entrada Aplicação 2

Os artefatos de entrada são representados por texto, que compõem a lista de compras a ser adquirida, a localização do supermercado, o nome do estabelecimento e a data e hora de ida às compras.

Os dados da lista de compras são preenchidos por formulário próprio e armazenados no banco de dados na primeira fase, para serem utilizados na segunda fase da atividade denominada *ida às compras*.

Na segunda fase, o paciente seleciona os produtos da lista, que encontrou e colocou no carrinho de compras. Ao final de cada gênero de produtos, o botão registrar mantém na base de dados os itens comprados.

8.10 Descrição da Aplicação 2

A aplicação proposta está dividida e duas fases: a primeira fase consiste no cadastramento da lista de produtos a serem adquiridos, sendo o cadastramento dos produtos organizados por gênero.

A organização por gênero objetiva facilitar o processo de recuperação da memória do paciente relacionado aos itens que precisa adquirir no supermercado.

A segunda fase acontece durante a realização das compras no supermercado. O paciente ao ativar a aplicação, surge no *display* a lista de produtos classificados por gênero. A Cada produto colocado no carrinho de compras, o item correspondente na lista de compras deve ser marcado, como adquirido ou não encontrado.

No final de cada listagem por gênero, o botão registrar deve ser pressionado para manter na base de dados os itens adquiridos e avançar para o próximo gênero.

8.10.1 Formas de interação da Atividade com o Paciente

A aplicação se comunica com o paciente pelas mensagens de *feedback* geradas e apresentadas pelos componentes narrativa, *feedback* e *avatar*.

Na elaboração da lista de compras (primeira fase) as orientaçãoes de execução são proferidas ao paciente pelos componentes de *gamificação* (narrativa, *avatar* e *feedback*).

Durante a elaboração da lista de compras, o paciente escolhe os produtos classificados por gênero e salva a lista intermediária, sendo o paciente parabenizado e motivado, após o salvamento da lista intermediária.

Os gêneros de produtos não inclusos na lista de compras são desconsiderados com o pressionamento do botão avançar.

A realização das compras (segunda fase) ao iniciar a aplicação, explicações sobre o funcionamento desta etapa da atividade é realizada utilizando os componentes de *gamificação* (narrativa, *avatar* e *feedback*).

A listagem de produtos no momento das compras é apresentada e classificada por gênero e apresenta dois campos de checagem, para registrar a aquisição ou a falta dos produtos.

No final cada lista por gênero é salva no banco de dados, pressionando o botão registrar. Dois tipos de mensagem podem ser retornadas ao paciente, uma mensagem de sucesso e uma mensagem avisando o paciente que algum item da lista foi esquecido.

Com todos os itens da lista de um gênero adquiridos, a próxima lista é apresentada após o registro da lista anterior.

No encerramento das compras, o paciente é parabenizado pela realização da atividade através dos componentes de *gamificação*.

8.10.2 Exercícios Propostos na Atividade

Os exercícios realizados pelo paciente consistem na lembrança dos itens, para a montagem da lista de compras do supermercado. Na segunda fase a localização espaço temporal relativa ao deslocamento até o supermercado e a busca pelos produtos anotados na lista de compras.

Fluxo de Execução da Atividade

A atividade é constituída por duas fases executadas não sequenciais no tempo. A lista de compras (primeira fase) é elaborada e a compra dos itens(segunda fase) propriamente ditos é executada em outro momento.

Fase 1:

- Passo 1: Paciente escolhe criar lista itens;
- Passo 2: Paciente escolhe criar lista intermediária de itens para a categoria ou avança a categoria;
- Passo 3: Paciente define os produtos a serem adquiridos classificados na categoria;
- Passo 4: Paciente salva a lista intermediária;
- Passo 5: Paciente escolhe avançar categoria;
- Passo 6: Se não há categorias finalizar lista; e
- Passo 7: Voltar passo 2.0.

Fase 2:

- Passo 1: Paciente seleciona a categoria;
- Passo 2: Paciente marca os produtos como adquiridos ou não encontrados;
- Passo 3: Paciente salva a lista intermediária;
- Passo 4: Se houver categorias o paciente seleciona a próxima categoria;
- Passo 5: Senão finaliza as compras.

8.11 Modelando a Aplicação Móvel de Apoio a Atividades da Vida Diária: Fase 01

Nesta primeira fase, o cuidador pode definir os artefatos de entrada, as *interfaces* da aplicação e os elementos de *gamificação*.

8.11.1 Etapa 1: Escolha dos Artefatos de Entrada

Nesta etapa, o cuidador auxiliado pelo paciente escolhe os artefatos (imagens e textos) utilizados durante a realização de atividades diárias.

Os artefatos escolhidos representam imagens de produtos passíveis de serem adquiridos em supermercados.

A cada lista de artefatos criada é necessário confirmar a criação, pressionando o botão criar lista. A ação de criação da lista gera uma sequência de *tags* XML.

A área de inserção de artefatos de entrada é demarcada pelas *tags* <*Input*></*Input*>. No interior da *tag input* estão listadas as *tags* representantes dos artefatos utilizados na aplicação terapêutica.

A *tag* <*Photo*> é composta por dois atributos obrigatórios, *id* e *src*. O atributo *id* mantém a ordem de inserção dos artefatos, sendo a ordem iniciada pelo índice 1. O atributo *src* está associado ao nome do arquivo de imagem integrante da lista.

8.11.2 Etapa 2: Escolha das *Interfaces* da Aplicação

As *interfaces* são construídas com antecedência, seguindo as definições e *layouts* xml da plataforma *Android*. O número de *interfaces* utilizadas depende de quantas *activitys* a aplicação irá conter.

As *interfaces* são escolhidas pela seleção múltiplos modelos predefinidos de *layouts*, para as janelas de interação com o usuário. Após a seleção, o botão salvar *layouts* é pressionado.

8.11.3 Etapa 3: Escolha dos Elementos de *Gamificação* da Aplicação

Após a definição das *interfaces* da aplicação, os componentes de *gamificação* e suas configurações são definidos e inseridos na aplicação.

A aplicação teste é composta por três componentes de *gamificação* escolhidos pelo cuidador.

Na sequência é exemplificado o processo de inserção e configuração dos componentes de *gamificação* da aplicação de apoio a atividades da vida diária.

O cuidador tem a possibilidade de selecionar os componentes de *gamificação* de uma lista predefinida, para o projeto da aplicação *gamificada*. Após a escolha, a configuração dos atributos e salvamento dos componentes, uma sequência de *tags* XML é gerada.

A área de inserção dos componentes de *gamificação* é demarcada pelo par de *tags* <*Gamification*></*Gamification*>. No interior da *tag gamification* estão listadas as *tags* representantes dos componentes de *gamificação* utilizados na aplicação.

O componente narrativa é representado por duas *tags*. O primeiro par de *tags* <*Emb_Narrative*></*Emb_Narrative*> representa as narrativas embarcadas (seção 5.6 narrativa), entre o par de *tags* é inserido o texto relativo as informações de funcionamento da atividade contida na aplicação.

O segundo par de *tags* <*Eme_Narrative*></*Eme_Narrative*> representa as narrativas emergentes (seção 5.6), os atributos *id* e *type* são obrigatórios. O atributo *id* mantém a ordem de inserção dos artefatos, sendo a ordem iniciada pelo índice número 1. O atributo *type* identifica os tipos de narrativas emergentes retornadas como *feedback* para o paciente (tabela 6.6). Entre o par de *tags* é inserido o texto relativo as mensagens utilizadas, como resposta as interações do paciente com a atividade.

As mensagens de *feedback* são inseridas na aplicação, por meio da *tag* <*Feedback*> composta por dois atributos obrigatórios *id* e *visual*. O atributo *id* mantém a ordem de inserção dos artefatos, sendo a ordem iniciada pelo índice número 01. O atributo *visual* indica se o *feedback* a ser retornado ao paciente é visual (*yes*) ou não visual (*no*).

A inserção do avatar nas aplicações criadas pela arquitetura ocorrem, por meio da *tag* <*Avatar*> e a configuração dos atributos requeridos *src* e *time*. O atributo *src* está associado ao nome do arquivo que representa a imagem do *avatar*, e o atributo *time* representa o tempo de espera inicial da exibição das mensagens textuais e visuais no *display* do dispositivo.

8.12 Geração do Código da Aplicação de Apoio a Atividades da Vida Diária: Fase 02

Nesta fase após a realização das três etapas da primeira fase, o cuidador pressiona o botão *gerar código fonte da aplicação*.

O arquivo XML criado é utilizado para a geração do código fonte escrito na linguagem Java. O artefato *ParserXML* é responsável por integrar as *interfaces* (XML e *activity*), os componentes de *gamificação* na aplicação, as entradas e relatórios.

A atividade principal contém a instância de todos os componentes de *gamificação* utilizados na aplicação. O objetivo da *activity* principal é promover a integração de

todas as outras *activitys* da aplicação.

O código xml das atividades selecionados na etapa 2 da primeira fase são copiados para a pasta *layouts* do projeto.

A saída produzida pelo artefato *ParserXML* resulta em um projeto de aplicação móvel, que será compilado pelo *Android Studio*.

A estrutura hierárquica e a compilação da aplicação dois é feita conforme as seções 8.6.1 e 8.7.

Tabela 8.1: *Componentes de gamificação utilizados na Aplicação 1*

Componente	Forma de Utilização
Narrativa	Consistem em mensagens classificadas em duas categorias (embutida e emergente), armazenadas em um arquivo no formato XML. As narrativas embutidas descrevem cada atividade e seu funcionamento. Narrativas embutida: • Nesta primeira etapa da atividade serão apresentadas uma coleção de fotografias e vídeos relacionados a pessoas e acontecimentos familiares. • Agora na próxima etapa uma série de perguntas ligadas ao que assistimos será realizada. As narrativas emergentes se subdividem em 4 categorias: êxito total na execução da ação, solicitação de auxílio, erro na execução da ação e desistência da ação. **Êxito total na ação:** • Congratulações, hoje seu desempenho está excelente. • Hoje tu fostes surpreendente! **Solicitação de auxílio:** • Pois não, eu repito as instruções. • Fique tranquila estou aqui para ajudá-la. **Erro na execução da ação:** • Sem problemas, equívocos acontecem. Vamos tentar novamente? • Vamos tentar mais uma vez? Sem pressa e bastante calma. **Desistência da ação:** • Tudo bem, tentaremos em outra oportunidade. • Esta sem vontade de fazer algo é natural tentaremos mais tarde.
Avatar	A representação do avatar consiste em arquivos no formato *png* com dimensões 256x256 *pixels* escolhidas pela paciente ou cuidador, através de uma lista predefinida de imagens.
Feedback	O *feedback* ocorre mediante a interação da paciente com a aplicação. A cada resposta fornecida pela paciente é verificada sua corretude, então uma mensagem de retorno é atribuída e disparada pelo *avatar*. No caso de pedido de auxilío o *feedback* com explicação da atividade é retornado pelo avatar e uma mensagem de incentivo na sequência também é proferida a paciente. As mensagens são selecionadas mediante às ações realizadas pela paciente.
insígnias	à medida que a paciente realiza as ações de forma correta os pontos são acumulados e convertidos em imagens de orquídeas. A cada realização da paciente dentro da atividade é atribuída uma orquídea diferente, quanto maior a conquista mais rara é a orquídea oferecida. Os arquivos com as imagens estão no formato *png* e dimensões 256x256 *pixels*.

Tabela 8.2: *Componentes de gamificação utilizados na Aplicação 1 (continuação)*

Componente	Forma de Utilização
Regras	Controla as ações que a paciente pode realizar durante a execução da atividades, em que condições são oferecidas recompensas, insígnias, como atribuir a pontuação, tempo de espera para realização das ações, variação do tempo de espera. A entidade regras persistida na base de dados mantém as restrições de uso da aplicação, quando atribuir recompensas, insígnias a paciente, pontuação obtida a cada ação realizada, tempo para execução das ações e variação do tempo. **Regra: início da atividade** • caso único - A cada inicio da atividade o tempo de resposta para as ações é redefinido para o padrão (60 segundos). **Regra: Oferecimento de recompensas** • caso 01 - Durante a execução da atividade, se a paciente desistir de duas ações em sequência ou três desistências alternadas; • caso 02 - Durante a execução da atividade, se a paciente errar duas vezes a mesma ação; e • caso 03 - Durante a execução da atividade, se a paciente realizar de forma correta três ações, em um tempo de resposta decrescente. **Regra: Atribuição de insígnias** • caso único - A cada 100 pontos conquistados pela paciente durante a realização das ações uma insígnia é atribuída a coleção da paciente. **Regra: Execução das atividades** • caso 01 - Repetição da tentativa de realização da ação em caso de erro; • caso 02 - Não repetição da tentativa de realização da ação em caso de erro; • caso 03 - Utilização de auxílio na realização das ações; e • caso 04 - Não utilização de auxílio na realização das ações. **Regra: Atribuição de pontuação** • caso 01 - Atribuição de 10 pontos a cada acerto na primeira tentativa; e • caso 02 - Atribuição de 5 pontos a cada acerto depois da primeira tentativa.

Tabela 8.3: *Componentes de gamificação utilizados na aplicação 1 continuação*

Componente	Forma de Utilização
Regras	**Regra: Tempo de execução ação** • caso 01 - Na realização de pelo menos três ações abaixo do tempo e de forma correta, a espera é reduzida, conforme a variação de tempo definida; • caso 02 - O erro na realização de uma ação por mais de duas vezes o tempo de espera é acrescido; conforme a variação de tempo definida; e • caso 03 - Desconsideração do tempo de execução da ação. **Regra: Variação do tempo** • caso 01 - Variação de tempo em 10% do tempo inicial padrão; e • caso 02 - O tempo de espera pela realização de uma ação atingir 100% do tempo de espera inicial padrão, a atividade não utilizará a contagem de tempo (0 segundos).
Recompensas	Utilizada como forma de estímulo, as recompensas são compostas por mensagens de incentivo do grupo *Erro na execução da ação* das narrativas e imagens de pessoas ligadas a paciente. Na situação de desistência as recompensas são compostas por mensagens de incentivo do grupo *Desistência da ação* das narrativas. Ao acertar a realização das ações a paciente recebe mensagens do grupo *êxito total na ação* das narrativas parabenizando pelo desempenho. As mensagens de incentivo que agrupam texto e imagens são expressas na forma de cartões oferecidos a paciente, conforme a interação com a aplicação.
Pontos	A pontuação é atribuída a cada ação realizada pela paciente de forma correta na primeira tentativa, ou em mais de uma tentativa. O tempo necessário para a realização das ações influenciam na pontuação conquistada. Os pontos são representados pelos números naturais e conferidos a paciente, conforme as regras definidas no componente regras.
Progressão	Comparativo entre os resultados obtidos na realização da mesma atividade em outros momentos terapêuticos. A cada realização da atividade é calculado o MEEM persistidos na entidade progressão. Para comparação do progresso da paciente na realização da atividade será utilizado o MEEM, que pode ser acompanhado pelo cuidador por um gráfico de barras.
Níveis	Controla o tempo espera em que a paciente necessita para realizar as ações da atividade. O nível de dificuldade a cada nova execução da atividade retorna ao patamar inicial, definido no componente regras. à medida que a paciente realiza as ações mais rapidamente e as respostas estão corretas, o tempo de espera para a resposta da próxima ação é reduzido. Caso sejam constatados alongados erros e pedidos de auxílio, o tempo para realizar uma ação é acrescido. Este tempo de resposta pode variar conforme os níveis de habilidade da paciente se alteram. A variação do tempo de espera para realização é definido pelo componente regras.

Tabela 8.4: *Componentes de gamificação utilizados Aplicação 2*

Componente	Forma de Utilização
Narrativa	Consistem em mensagens classificadas em duas categorias (embutida e emergente) armazenadas em um arquivo no formato XML. **Narrativas embutidas na primeira fase:** • Qual o nome do supermercado que iremos? E em que endereço fica o supermercado?; • Vamos construir a lista de compras do supermercado. A lista será montada por gênero dos produtos (alimentação, higiene pessoal etc); • Ao preencher cada gênero com os itens e quantidades, pressione o botão registrar, para armazenar cada parte da lista.; e • Se algum gênero de produto não for necessário pressione o botão avançar. **Narrativas embutidas na segunda fase:** • iniciando as compras pelos gênero dos produtos.; As narrativas emergentes se subdividem em 3 categorias: êxito total na execução da ação, solicitação de auxílio, erro na execução da ação. **Narrativas emergentes primeira fase:** **Êxito total na ação:** • Muito bem, meus parabéns!!; • Tarefa concluída com sucesso.; e • Lista de compras concluída com sucesso. **Solicitação de auxílio:** • Sempre que os produtos pertencentes a um gênero forem todos relacionados é preciso pressionar o botão registrar.; e • Se não houver produtos a serem comprados de um determinado gênero pressione o botão avançar. **Narrativas emergentes segunda fase:** **Êxito total na ação:** • Muito bem, meus parabéns!!; e • Tarefa concluída com sucesso. **Erro na execução da ação:** • Acho que esquecemos alguma coisa, vamos verificar nossa lista de compras?. **Solicitação de auxílio:** • Sempre que os produtos pertencentes a um gênero forem todos comprados é preciso pressionar o botão registrar.

Tabela 8.5: *Componentes de gamificação utilizados na Aplicação 2(continuação)*

Componente	Forma de Utilização
Feedback	O *feedback* ocorre mediante a interação da paciente com a aplicação. A cada ação realizada pelo paciente uma resposta é retornada pelo avatar. No caso de pedido de auxílio o *feedback* com explicação da atividade é retornado pelo avatar e uma mensagem de incentivo na sequência também é proferida a paciente. As mensagens são selecionadas mediante às ações realizadas pela paciente.
Avatar	A representação do *avatar* consiste em arquivos no formato *png* com dimensões 256x256 *pixels* escolhidas pela paciente ou cuidador, através de uma lista predefinida de imagens.

CAPÍTULO 9

Conclusões e Trabalhos Futuros

O envelhecimento da população levou ao surgimento de doenças ligadas à idade. A doença de Alzheimer é uma doença neurodegenerativa sem cura, que afeta funções motoras e cognitivas. No decorrer do tempo pacientes acometidos pela doença de Alzheimer apresentam um quadro de depressão e instabilidade comportamental, que dificulta à realização de atividades diárias e o convívio social.

Na tentativa de amenizar os sintomas, medicamentos e ações terapêuticas são indicadas por médicos e terapeutas, para retardar o avanço da doença e proporcionar uma qualidade de vida melhor para os pacientes.

A terapia de reminiscência tem sido bastante aceita e utilizada para amenizar os sintomas da depressão, desvio do comportamento e reviver as memórias passadas e do presente do paciente.

A evolução das tecnologias de informação e comunicação permitiu o desenvolvimento de iniciativas, que empreguem os recursos computacionais, no apoio à realização da reminiscência, com o objetivo de melhorar os resultados do efeito das sessões nos paciente.

Os resultados relatados na literatura indicam que o uso das TICs têm efeitos positivos sobre pacientes com DA. Porém os elementos dos jogos, *interfaces* mais amigáveis, sensoriamento, comunicação e mobilidade dos dispositivos móveis precisam ser mais explorados.

Neste trabalho foi proposta uma arquitetura de *gamificação*, composta de dezessete componentes de *gamificação*, para apoiar a construção de aplicações terapêuticas *gamificadas* utilizáveis por pacientes diagnosticados com Alzheimer, nos níveis leve e moderado da doença.

A arquitetura foi definida para apoiar o desenvolvimento de aplicações móveis, por desenvolvedores de *software* com a colaboração de pacientes, cuidadores/familiares, para fornecer dados sobre as habilidades, preferências e artefatos que colaborem com o bem-estar dos pacientes.

Os protótipos criados para atestar a viabilidade dos conceitos desenvolvidos foram baseadas em personas, características e dados clínicos de paciente com doença

de Alzheimer e dados clínicos de pacientes presentes na literatura médica.

Os componentes de *gamificação* e aplicações exemplo propostos nesta dissertação consideraram itens de usabilidade específicos para o desenvolvimento aplicações móveis para usuários idosos.

Testes com aplicações desenvolvidas, por meio da arquitetura precisam ser realizados, para verificar os efeitos positivos na motivação e o engajamento de pacientes diagnosticados com Alzheimer.

Diversas limitações foram verificadas e estão listadas na sequência:

- A falta de conectividade com as redes sociais;
- A não integração com sistemas de monitoramento de saúde;
- Dados do paciente e das sessões terapêuticas persistidos apenas localmente; e
- Ausência de testes com pacientes.

Diante das limitações listadas abre-se a possibilidade de continuidade deste trabalho para ampliação e consolidação, sugeridas a seguir:

- Criar uma ferramenta de autoria;
- Utilizar as aplicações na terapia reminiscência de pacientes com DA, para a avaliação da efetividade da tecnologia proposta;
- Desenvolver a ferramenta de autoria para os cuidadores desenvolverem de forma automatizada aplicações com atividades para os pacientes com doença de Alzheimer;
- Melhorar a detecção do comportamento e humor dos pacientes;
- Desenvolver aplicações para outros tipos de demência; e
- Avaliar com idosos a acessibilidade e usabilidade dos componentes de *gamificação* e aplicações teste desenvolvidas.

Referências Bibliográficas

[1] ABU HASHIM, A. H.; ISMAIL, A. N.; MOHD RIAS, R.; MOHAMED, A. **The development of an individualized digital memory book for Alzheimer's disease patient: A case study**. In: *Technology Management and Emerging Technologies (ISTMET), 2015 International Symposium on*, p. 227–232. IEEE, 2015.

[2] ALBAN, A.; DE MARCHI, A. C. B.; SCORTEGAGNA, S. A.; LEGUISAMO, C. P. **Ampliando a usabilidade de interfaces web para idosos em dispositivos móveis: uma proposta utilizando design responsivo**. *RENOTE*, 10(3), 2012.

[3] ALM, N.; ASTELL, A.; ELLIS, M.; DYE, R.; GOWANS, G.; CAMPBELL, J. **A cognitive prosthesis and communication support for people with dementia**. *Neuropsychological rehabilitation*, 14(1-2):117–134, 2004.

[4] ALMEIDA, O. P. **Mini exame dos estado mental e o diagnóstico de demência no Brasil**. *Arquivos de Neuro-Psiquiatria*, 56:605 – 612, 09 1998.

[5] ANNUZZI, J.; DARCEY, L.; CONDER, S. **Introduction to Android application development: Android essentials**. Pearson Education, 2014.

[6] AŞIRET, G. D.; KAPUCU, S. **The effect of reminiscence therapy on cognition, depression, and activities of daily living for patients with Alzheimer disease**. *Journal of Geriatric Psychiatry and Neurology*, p. 0891988715598233, 2015.

[7] ASTELL, A.; ALM, N.; GOWANS, G.; ELLIS, M.; DYE, R.; VAUGHAN, P. **Involving older people with dementia and their carers in designing computer based support systems: some methodological considerations**. *Universal Access in the Information Society*, 8(1):49–58, 2008.

[8] ASTELL, A. J.; ELLIS, M. P.; BERNARDI, L.; ALM, N.; DYE, R.; GOWANS, G.; CAMPBELL, J. **Using a touch screen computer to support relationships between people with dementia and caregivers**. *Interacting with Computers*, 22(4):267 – 275, 2010. Supportive Interaction: Computer Interventions for Mental Health.

[9] ÁVILA, R.; OTHERS. **Resultados da reabilitação neuropsicológica em paciente com doença de Alzheimer leve**. *Revista de psiquiatria clínica*, 30(4):139–146, 2003.

[10] BAÑOS, R. M.; ETCHEMENDY, E.; CASTILLA, D.; GARCIA-PALACIOS, A.; QUERO, S.; BOTELLA, C. **Positive mood induction procedures for virtual environments designed for elderly people**. *Interacting with Computers*, 24(3):131–138, 2012.

[11] BARBAN, F.; ANNICCHIARICO, R.; PANTELOPOULOS, S.; FEDERICI, A.; PERRI, R.; FADDA, L.; CARLESIMO, G. A.; RICCI, C.; GIULI, S.; SCALICI, F.; OTHERS. **Protecting cognition from aging and Alzheimer's disease: a computerized cognitive training combined with reminiscence therapy**. *International journal of geriatric psychiatry*, 2015.

[12] BARNETT, J.; HARRICHARAN, M.; FLETCHER, D.; GILCHRIST, B.; COUGHLAN, J. **mypace: An integrative health platform for supporting weight loss and maintenance behaviors**. *IEEE journal of biomedical and health informatics*, 19(1):109–116, 2015.

[13] BAYLOR, A. L. **Promoting motivation with virtual agents and avatars: role of visual presence and appearance**. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 364(1535):3559–3565, 2009.

[14] BERNDT, A.; CARVALHO, S. T.; ALBUQUERQUE, E. **Uma classificação para aplicações terapêuticas gamificadas cognitivas para pacientes com doença de alzheimer**. In: *CBIS 2016 ()*, http://www.sbis.org.br/cbis2016, nov 2016.

[15] BERTOLUCCI, P. H.; BRUCKI, S.; CAMPACCI, S. R.; JULIANO, Y. **The mini-mental state examination in an outpatient population: influence of literacy**. *Arquivos de Neuro-psiquiatria*, 52(1):01–07, 1994.

[16] BOHLMEIJER, E.; VALENKAMP, M.; WESTERHOF, G.; SMIT, F.; CUIJPERS, P. **Creative reminiscence as an early intervention for depression: Results of a pilot project**. *Aging & Mental Health*, 9(4):302–304, 2005.

[17] BRAUNER, P.; VALDEZ, A. C.; SCHROEDER, U.; ZIEFLE, M. **Increase physical fitness and create health awareness through exergames and gamification**. In: *Human Factors in Computing and Informatics*, p. 349–362. Springer, 2013.

[18] CAIXETA, L. **Doença de Alzheimer**. Artmed Editora, 2012.

[19] CORRÊA, S. E. S.; SILVA, D. B. D. **Abordagem cognitiva na intervenção terapêutica ocupacional com indivíduos com doença de Alzheimer**. *Rev. Bras. Geriat. Geront*, 12(3):463–474, 2009.

[20] COSTA, A. C. S.; MARCHIORI, P. Z. **Gamificação, elementos de jogos e estratégia: uma matriz de referência**. *InCID: Revista de Ciência da Informação e Documentação*, 6(2):44–65, 2015.

[21] COTA, T. T.; ISHITANI, L.; JR., N. V. **Mobile game design for the elderly: A study with focus on the motivation to play**. *Computers in Human Behavior*, 51, Part A:96 – 105, 2015.

[22] COTELLI, M.; MANENTI, R.; ZANETTI, O. **Reminiscence therapy in dementia: A review**. *Maturitas*, 72(3):203–205, 2012.

[23] COWAN, L. T.; VAN WAGENEN, S. A.; BROWN, B. A.; HEDIN, R. J.; SEINO-STEPHAN, Y.; HALL, P. C.; WEST, J. H. **Apps of steel: are exercise apps providing consumers with realistic expectations? A content analysis of exercise apps for presence of behavior change theory**. *Health Education & Behavior*, p. 1090198112452126, 2012.

[24] CUGELMAN, B. **Gamification: what it is and why it matters to digital health behavior change developers. jmir serious games. 2013; 1 (1): e3. 3139**.

[25] DAVIS, B. H.; SHENK, D. **Beyond reminiscence using generic video to elicit conversational language**. *American journal of Alzheimer's disease and other dementias*, p. 1533317514534759, 2014.

[26] DAVISON, T. E.; NAYER, K.; COXON, S.; DE BONO, A.; EPPINGSTALL, B.; JEON, Y.-H.; VAN DER PLOEG, E. S.; O'CONNOR, D. W. **A personalized multimedia device to treat agitated behavior and improve mood in people with dementia: A pilot study**. *Geriatric Nursing*, 37(1):25 – 29, 2016.

[27] DE ALMEIDA, R. X. E.; FERREIRA, S. B. L.; SOARES, H. P. **Recomendações para desenvolvimento de interfaces web em tablet ipad com ênfase em usuários da terceira idade**. In: *Proceedings of the 13th Brazilian Symposium on Human Factors in Computing Systems*, IHC '14, p. 21–30, Porto Alegre, Brazil, Brazil, 2014. Sociedade Brasileira de Computação.

[28] DE MARCHI, A. C. B.; COLUSSI, E. L.; ZIMMER, M.; TROMBETTA, M.; BIDUSKI, D. **Identificando problemas de usabilidade em um aplicativo móvel para treino de memória em idosos**. In: *Proceedings of the 13th Brazilian Symposium on Human Factors in Computing Systems*, IHC '14, p. 373–376, Porto Alegre, Brazil, Brazil, 2014. Sociedade Brasileira de Computação.

[29] DE OLIVEIRA SANTOS, L. G. N.; ISHITANI, L.; NOBRE, C. N. **Uso de jogos casuais em celulares por idosos: um estudo de usabilidade**. *Revista de Informática Aplicada*, 9(1), 2014.

[30] DETERDING, S.; DIXON, D.; KHALED, R.; NACKE, L. **From game design elements to gamefulness: defining gamification**. In: *Proceedings of the 15th international*

academic MindTrek conference: Envisioning future media environments, p. 9–15. ACM, 2011.

[31] DEVELOPERS, A. **Android, the world's most popular mobile platform**. developer.android.com/about/index.html. Acessado: o5-08-2016.

[32] DEVELOPERS, A. **Dashboards**. developer.android.com/about/dashboards/index.html. Acessado: 05-08-2016.

[33] ENGELHARDT, E.; GOMES, M. D. M. **Alzheimer's 100th anniversary of death and his contribution to a better understanding of senile dementia**. *Arquivos de Neuro-Psiquiatria*, 73:159 – 162, 02 2015.

[34] ETCHEVERRIA, A. K.; PREDIGER, I. O.; BAUER, M. A.; CORRÊA, A. O.; BIGGOWEIT, M. S. B.; ZANOTTO, L. R. S.; TEIXEIRA, A. R.; OTHERS. **Estudo sobre a audição em idosos e associação com sintomatologia depressiva**. *Revista Brasileira de Ciências do Envelhecimento Humano*, 11(2), 2014.

[35] FARLEY, P. C. **Using the computer game foldit to entice students to explore external representations of protein structure in a biochemistry course for non-majors**. *Biochemistry and Molecular Biology Education*, 41(1):56–57, 2013.

[36] FERREIRA, S. B. L.; NUNES, R. R.; DA SILVEIRA, D. S.; SOARES, H. P. **Tornando os requisitos de usabilidade mais aderentes às diretrizes de acessibilidade**. *Usabilidade, Acessibilidade e Inteligibilidade Aplicadas em Interfaces para Analfabetos, Idosos e Pessoas com Deficiência*, p. 43, 2008.

[37] FOGG, B. J. **A behavior model for persuasive design**. In: *Proceedings of the 4th international Conference on Persuasive Technology*, p. 40. ACM, 2009.

[38] FOWLER, M. **Patterns of enterprise application architecture**. Addison-Wesley Longman Publishing Co., Inc., 2002.

[39] FREEMAN, E.; ROBSON, E.; BATES, B.; SIERRA, K. **Head first design patterns**. "O'Reilly Media, Inc.", 2004.

[40] GAMMA, E. **Design patterns: elements of reusable object-oriented software**. Pearson Education India, 1995.

[41] GONÇALVES, D. C.; ALBUQUERQUE, P. B.; MARTÍN, I. **Reminiscência enquanto ferramenta de trabalho com idosos: Vantagens e limitações**. *Análise Psicológica*, 26(1):101–110, 2012.

[42] GONÇALVES, V. P.; ALMEIDA NERIS, V. P.; SERAPHINI, S.; DIAS, T. C. M.; PESSIN, G.; JOHNSON, T.; UEYAMA, J. **Providing adaptive smartphone interfaces targeted at elderly people: an approach that takes into account diversity among the elderly**. *Universal Access in the Information Society*, p. 1–21, 2015.

[43] GOOD, B. M.; SU, A. I. **Games with a scientific purpose**. *Genome Biology*, 12(12):1–3, 2011.

[44] HABER, D. **Life review: Implementation, theory, research, and therapy**. *The International Journal of Aging and Human Development*, 63(2):153–171, 2006.

[45] HARGOOD, C.; MILLARD, D. E.; WEAL, M. J. **A thematic approach to emerging narrative structure**. In: *Proceedings of the Hypertext 2008 Workshop on Collaboration and Collective Intelligence*, WebScience '08, p. 41–45, New York, NY, USA, 2008. ACM.

[46] HARRICHARAN, M.; GEMEN, R.; CELEMÍN, L. F.; FLETCHER, D.; DE LOOY, A. E.; WILLS, J.; BARNETT, J. **Integrating mobile technology with routine dietetic practice: the case of mypace for weight management**. *Proceedings of the Nutrition Society*, 74:125–129, 5 2015.

[47] HEDMAN, A.; HALLBERG, J. **Cognitive endurance for brain health: Challenges of creating an intelligent warning system**. *KI - Künstliche Intelligenz*, 29(2):123–129, 2015.

[48] ISAKOVIĆ, M.; SEDLAR, U.; VOLK, M.; BEŠTER, J. **Usability pitfalls of diabetes mhealth apps for the elderly**. *Journal of Diabetes Research*, 2016, 2016.

[49] KAPP, K. M. **The gamification of learning and instruction fieldbook: Ideas into practice**. John Wiley & Sons, 2013.

[50] KING, D.; GREAVES, F.; EXETER, C.; DARZI, A. **Gamification: Influencing health behaviours with games**. *Journal of the Royal Society of Medicine*, 106(3):76–78, 2013.

[51] KURNIAWAN, S. **Older people and mobile phones: A multi-method investigation**. *International Journal of Human-Computer Studies*, 66(12):889 – 901, 2008. Mobile human-computer interaction.

[52] LANCIONI, G. E.; SINGH, N. N.; O'REILLY, M. F.; SIGAFOOS, J.; FERLISI, G.; ZULLO, V.; SCHIRONE, S.; PRISCO, R.; DENITTO, F. **A computer-aided program for helping patients with moderate Alzheimer's disease engage in verbal reminiscence**. *Research in developmental disabilities*, 35(11):3026–3033, 2014.

[53] LAZAR, A.; THOMPSON, H.; DEMIRIS, G. **A systematic review of the use of technology for reminiscence therapy**. *Health education & behavior*, 41(1 suppl):51S–61S, 2014.

[54] LENIHAN, D. **Health games: a key component for the evolution of wellness programs**. *Games for Health: Research, Development, and Clinical Applications*, 1(3):233–235, 2012.

[55] LEUNG, R.; MCGRENERE, J.; GRAF, P. **Age-related differences in the initial usability of mobile device icons**. *Behaviour & Information Technology*, 30(5):629–642, 2011.

[56] LISTER, C.; WEST, J. H.; CANNON, B.; SAX, T.; BRODEGARD, D. **Just a fad? Gamification in health and fitness apps**. *JMIR serious games*, 2(2), 2014.

[57] LOPES, T.; AFONSO, R.; RIBEIRO, O. **Impacto de intervenções de reminiscência em idosos com demência: revisão da literatura**. *Psicologia, Saúede & Doenças*, 15:597 – 611, 12 2014.

[58] LOPES, T. S.; AFONSO, R. M. L. B. M.; RIBEIRO, Ó. M. S. **Programa de reminiscência simples para pessoas idosas com demência**. *International Journal of Developmental and Educational Psychology*, 2013.

[59] MACEDO, B. G. D.; PEREIRA, L. S. M.; GOMES, P. F.; SILVA, J. P. D.; CASTRO, A. N. V. D. C. **Impacto das alterações visuais nas quedas, desempenho funcional, controle postural e no equilíbrio dos idosos: uma revisão de literatura**. *Revista Brasileira de Geriatria e Gerontologia*, 11:419 – 432, 12 2008.

[60] MATURO, A.; SETIFFI, F. **The gamification of risk: how health apps foster self-confidence and why this is not enough**. *Health, Risk & Society*, p. 1–18, 2016.

[61] MICHIE, S.; VAN STRALEN, M. M.; WEST, R. **The behaviour change wheel: a new method for characterising and designing behaviour change interventions**. *Implementation Science*, 6(1):42, 2011.

[62] OLIVEIRA, K. C. V.; BARROS, A. L. S.; SOUZA, G. **Mini-exame do estado mental (MEEM) e clinical dementia rating (CDR) em idosos com doença de Alzheimer**. *Rev Neurocienc*, 16(2):101–106, 2008.

[63] OWENS, M.; ALLEN, G. **SQLite**. Springer, 2010.

[64] PEREIRA, P.; DUARTE, E.; REBELO, F.; NORIEGA, P. **A review of gamification for health-related contexts**. In: *Design, User Experience, and Usability. User*

Experience Design for Diverse Interaction Platforms and Environments, p. 742–753. Springer, 2014.

[65] PILGER, C. A.-O.; MENON, M. U.; MATHIAS, T. A. D. F. **Capacidade funcional de idosos atendidos em unidades básicas de saúde do SUS** . *Revista Brasileira de Enfermagem*, 66:907 – 913, 12 2013.

[66] PRAKASH, E. C.; RAO, M. **Transforming Learning and IT Management through Gamification**, chapter Introduction to Gamification, p. 35–46. Springer International Publishing, Cham, 2015.

[67] PRINCE, M. J. **World Alzheimer Report 2015: The Global Impact of Dementia an Analysis of Prevalence, Incidence, Cost and Trends**. Alzheimer's Disease International (ADI), 2015.

[68] RAGLIO, A.; BELLELLI, G.; MAZZOLA, P.; BELLANDI, D.; GIOVAGNOLI, A.; FARINA, E.; STRAMBA-BADIALE, M.; GENTILE, S.; GIANELLI, M.; UBEZIO, M.; OTHERS. **Music, music therapy and dementia: a review of literature and the recommendations of the italian psychogeriatric association**. *Maturitas*, 72(4):305–310, 2012.

[69] ROCHA, R.; REIS, L. P.; REGO, P. A.; MOREIRA, P. M. **Serious games for cognitive rehabilitation: Forms of interaction and social dimension**. In: *Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on*, p. 1 – 6, June 2015.

[70] SEABORN, K.; FELS, D. I. **Gamification in theory and action: A survey**. *International Journal of Human-Computer Studies*, 74:14 – 31, 2015.

[71] SERRANI AZCURRA, D. J. L. **A reminiscence program intervention to improve the quality of life of long-term care residents with Alzheimer's disease: a randomized controlled trial**. *Revista Brasileira de Psiquiatria*, 34(4):422–433, 2012.

[72] SQLITE.ORG. **About sqlite**. www.sqlite.org/about.html, ago 2016. Acessado: 02-08-2016.

[73] SQLITE.ORG. **Datatypes in sqlite version 3**. www.sqlite.org/datatype3.html/#section_1, ago 2016. Acessado: 02-08-2016.

[74] STINSON, J. N.; JIBB, L. A.; NGUYEN, C.; NATHAN, P. C.; MALONEY, A. M.; DUPUIS, L. L.; GERSTLE, J. T.; ALMAN, B.; HOPYAN, S.; STRAHLENDORF, C.; OTHERS. **Development and testing of a multidimensional iphone pain assessment application for adolescents with cancer**. *Journal of medical Internet research*, 15(3):e51, 2013.

[75] STUART, A. G. **Exercise as therapy in congenital heart disease? A gamification approach**. *Progress in Pediatric Cardiology*, 38(1):37–44, 2014.

[76] SULAIMAN, S.; SOHAIMI, I. S. **An investigation to obtain a simple mobile phone interface for older adults**. In: *Intelligent and Advanced Systems (ICIAS), 2010 International Conference on*, p. 1–4, June 2010.

[77] VECHIATO, F. L.; VIDOTTI, S. A. B. G. **Recomendações de usabilidade e de acessibilidade em projetos de ambientes informacionais digitais para idosos**. *Tendências da pesquisa brasileira em ciência da informação*, p. 1–23, 2012.

[78] VERAS, R. P.; MATTOS, L. C. **Audiologia do envelhecimento: revisão da literatura e perspectivas atuais**. *Rev Bras Otorrinolaringol*, 73(1):128–34, 2007.

[79] VIANNA, Y.; VIANNA, M.; MEDINA, B.; TANAKA, S. **Gamification, Inc: Como reinventar empresas a partir de jogos**. MJV Press, 2013.

[80] WEBSTER, J. D.; BOHLMEIJER, E. T.; WESTERHOF, G. J. **Mapping the future of reminiscence: A conceptual guide for research and practice**. *Research on Aging*, 32(4):527–564, 2010.

[81] WEI, H. **Embedded narrative in game design**. In: *Proceedings of the International Academic Conference on the Future of Game Design and Technology*, Futureplay '10, p. 247–250, New York, NY, USA, 2010. ACM.

[82] WEISER, P.; BUCHER, D.; CELLINA, F.; DE LUCA, V. **A taxonomy of motivational affordances for meaningful gamified and persuasive technologies**. *ICT for Sustainability (ICT4S)*, 2015.

[83] WERBACH, K.; HUNTER, D. **For the win: How game thinking can revolutionize your business**. Wharton Digital Press, 2012.

[84] ZMILY, A.; MOWAFI, Y.; MASHAL, E. **Study of the usability of spaced retrieval exercise using mobile devices for alzheimer's disease rehabilitation**. *JMIR mHealth and uHealth*, 2(3), 2014.

APÊNDICE A

Documento de Definição de Tipo (Arquivo DTD)

Este apêndice apresenta a definição das entidades e tipos válidos na construção dos arquivos XMLs das atividades.

A.1 Arquivo DTD

Código A.1 Documento de Definição de Tipo

```
<!DOCTYPE Root_Activity [
<!ELEMENT Root_Activity (Sub_Activity+)>
<!ATTLIST Root_Activity xmlns CDATA #FIXED
"http://www.berndt.com/gamification/autoria/atividade">
<!ELEMENT Sub_Activity (Gamification, Input, Action*,Report*)>
<!ATTLIST Sub_Activity id CDATA #REQUIRED type CDATA #REQUIRED>
<!--gamification inicio-->
<!ELEMENT Gamification (Emb_Narrative, Eme_Narrative+, Feedback,
Avatar, Rules?,  Ensign+, Reward*, Progress?, Mission+,
Achievment+, Challenge*, Competition*)>
<!ELEMENT Emb_Narrative (Message)>
<!ELEMENT Message (#PCDATA)>
<!ELEMENT Eme_Narrative (Message)>
<!ATTLIST Eme_Narrative id CDATA #REQUIRED type CDATA #REQUIRED>
<!ELEMENT Feedback EMPTY>
<!ATTLIST Feedback audible (yes|no) "yes" #REQUIRED>
<!ELEMENT Avatar EMPTY>
<!ATTLIST Avatar src CDATA #REQUIRED time CDATA #REQUIRED>
<!ELEMENT Rules (Points, Level+, Offer_Reward, Offer_Ensign,
Action_Replay?, Action_Help?, Question_Time?, Time_Variation?)>
```

Código A.2 Documento de Definição de Tipo continuação

```
1          <!ELEMENT Points (Correct,First_Error)>
2          <!ELEMENT Correct EMPTY>
3          <!ATTLIST Correct value CDATA #REQUIRED>
4          <!ELEMENT First_Error EMPTY>
5          <!ATTLIST First_Error value CDATA #REQUIRED>
6          <!ELEMENT Level EMPTY>
7          <!ATTLIST Level id CDATA #REQUIRED value CDATA #REQUIRED>
8          <!ELEMENT Offer_Reward (Abort?, Abort_Alternate?, Error_Action?,
9          Correct_Action?)>
10         <!ELEMENT Abort (#PCDATA)>
11         <!ELEMENT Abort_Alternate (#PCDATA)><!ELEMENT Error_Action (#PCDATA)
12         <!ELEMENT Correct_Action (#PCDATA)>
13         <!ELEMENT Offer_Ensign EMPTY>
14         <!ATTLIST Offer_Ensign value CDATA #REQUIRED>
15         <!ELEMENT Action_Replay EMPTY>
16         <!ATTLIST Action_Replay value (yes|no) "yes">
17         <!ELEMENT Action_Help EMPTY>
18         <!ATTLIST Action_Help value (yes|no) "yes">
19         <!ELEMENT Question_Time EMPTY>
20         <!ATTLIST Question_Time time CDATA #REQUIRED>
21         <!ELEMENT Time_Variation EMPTY>
22         <!ATTLIST Time_Variation rate CDATA #REQUIRED>
23         <!ELEMENT Ensign EMPTY>
24         <!ATTLIST Ensign id CDATA #REQUIRED src CDATA #REQUIRED>
25         <!ELEMENT Reward (Text,Image?)>
26         <!ATTLIST Reward id CDATA #REQUIRED>
27         <!ELEMENT Text (#PCDATA)>
28         <!ELEMENT Image EMPTY>
29         <!ATTLIST Image src CDATA #REQUIRED>
30         <!ELEMENT Progress EMPTY>
31         <!ATTLIST Progress meem CDATA #REQUIRED>
32         <!ELEMENT Mission EMPTY>
33         <!ATTLIST Mission type CDATA #REQUIRED, time CDATA>
```

Código A.3 Documento de Definição de Tipo continuação

```
<!ELEMENT Challenge EMPTY>
<!ATTLIST  Challenge correct CDATA #REQUIRED error CDATA
 #REQUIRED endeavor CDATA #REQUIRED>
<!ELEMENT Acheivement EMPTY>
<!ATTLIST Acheivement score (yes|no) "no" ensign (yes|no) "no"
reward (yes|no) "no">
<!ELEMENT Competition EMPTY>
<!ATTLIST Competition difficulty CDATA #REQUIRED)>
<!--gamification final-->
<!--input inicio-->
<!ELEMENT Input (Photo+, Video*, Sound*, Information*)>
<!ELEMENT Photo EMPTY>
<!ATTLIST Photo id CDATA #REQUIRED src CDATA #REQUIRED>
<!ELEMENT Video EMPTY)>
<!ATTLIST Video id CDATA #REQUIRED src CDATA #REQUIRED>
<!ELEMENT Sound EMPTY)>
<!ATTLIST Sound id CDATA #REQUIRED src CDATA #REQUIRED>
<!ELEMENT Information EMPTY>
<!ATTLIST Information id CDATA #REQUIRED src CDATA #REQUIRED>
<!--input final-->
<!--Action inicio-->
<!ELEMENT Action (Question*, Notice*, Listing*)>
<!ATTLIST Action id CDATA #REQUIRED>
<!--Question inicio-->
<!ELEMENT Question (Text, Alternative*, Alternative_Image*,
Multi_Alternative*, (Answer|Textual_Answer)?)>
<!ATTLIST Question id CDATA #REQUIRED type CDATA #REQUIRED>
<!ELEMENT Alternative EMPTY>
<!ATTLIST Alternative id CDATA #REQUIRED src CDATA #REQUIRED>
<!ELEMENT Alternative_Image EMPTY>
<!ATTLIST Alternative_Image id CDATA #REQUIRED src CDATA #REQUIRED>
<!ELEMENT Multi_Alternative (Item*, Multi_Answer*)>
```

Código A.4 Documento de Definição de Tipo continuação

```
<!ELEMENT Item EMPTY>
<!ATTLIST Item id CDATA #REQUIRED src CDATA #REQUIRED>
<!ATTLIST Item id CDATA #REQUIRED src CDATA #REQUIRED>
<!ELEMENT Multi_Answer (Ans+)><!ELEMENT Ans (#PCDATA)>
<!ELEMENT Answer (#PCDATA)>
<!ELEMENT Textual_Answer (#PCDATA)>
<!--Question final-->
<!--Notice inicio-->
<!ELEMENT Notice (Event_Name, Day, Month, Year, Place, People*)>
<!ATTLIST Notice id CDATA #REQUIRED>
<!ELEMENT Event_Name (#PCDATA)>
<!ELEMENT Day (#PCDATA)>
<!ELEMENT Month (#PCDATA)>
<!ELEMENT Year (#PCDATA)>
<!ELEMENT Place (#PCDATA)>
<!ELEMENT People (#PCDATA)>
<!--Notice final-->
<!--Listing inicio-->
<!ELEMENT Listing EMPTY>
<!ATTLIST id CDATA #REQUIRED src CDATA #REQUIRED>
<!--Listing final-->
<!--Action final-->
<!--Report inicio-->
<!ELEMENT Report (Patient?,Caregiver+)>
<!ELEMENT Patient EMPTY>
<!ATTLIST Patient view (yes|no) "no">
<!ELEMENT Caregiver (#PCDATA)> <!--Email destinatario-->
<!--Report final-->
]>
```

APÊNDICE B

Exemplo Arquivo XML

Este apêndice apresenta uma atividade XML exemplo.

B.1 Arquivo XML

Código B.1 Definição XML da atividade

```
<?xml version="1.0" encoding="utf-8"?>
<Root_Activity>
        <Sub_Activity>
                <Gamification>
                        <Emb_Narrative>
                                <Message> Narrativa Embutida</Message>
                        </Emb_Narrative>
                        <Eme_Narrative id="1" type="0">
                                <Message>Narrativa Emergente 01</Message>
                        </Eme_Narrative>
                        <Eme_Narrative id="2" type="1">
                                <Message> Narrativa Emergente 02</Message>
                        </Eme_Narrative>
                        <Feedback visual="yes">
                                <Message>feed01.png</Message>
                        </Feedback>
                        <Avatar src="Face.jpg" time="10"/>
                        <Rules>
                                <Points>
                                        <Correct>10</Correct>
                                        <First_Error>5</First_Error>
                                </Points>
```

Código B.2 Definição XML da atividade continuação

```
                                    <Level id="1">100</Level>
                                    <Level id="2">200</Level>
                                    <Reward id="1">
                                            <Text>Parabens</Text>
                                            <Symbol>img.png</Symbol>
                                    </Reward>
                                    <Reward id="2">
                                            <Text>Ok</Text>
                                            <Symbol>img2.png</Symbol>
                                    </Reward>
                            </Rules>
                            <Ensign id="1" src="Flower.jpg"/>
                            <Ensign id="2" src="Flower2.jpg"/>
                    </Gamification>
                    <Input>
                            <Photo id="1">filho.jpg</Photo>
                            <Video id="1">casamento.mp4</Video>
                            <Sound id="1">piano.mp3</Sound>
                    </Input>
                    <Question id="1">
                            <Text>Qual nome do seu filho?</Text>
                            <Alternative id="1">Jose</Alternative>
                            <Alternative id="2">Pedro</Alternative>
                            <Alternative id="3">Carlos</Alternative>
                            <Answer>2</Answer>
                    </Question>
                    <Question id="2">
                    <Text>Qual fotografia mostra seu filho?</Text>
                    <Alternative_img id="1">foto01.jpg</Alternative_img>
                    <Alternative_img id="2">foto02.jpg</Alternative_img>
                    <Alternative_img id="3">foto03.jpg</Alternative_img>
                    <Answer>3</Answer>
```

Código B.3 Definição XML da atividade continuação

```
                </Question>
                <Question id="3">
                        <Text>Quais cidades tu conheces?</Text>
                        <Multi_anternative>
                                <Item id="1">Paris</Item>
                                <Item id="2">SÃ£o Paulo</Item>
                                <Item id="3">Oslo</Item>
                                <Item id="4">CuiabÃ¡</Item>
                                <Multi_Answer>
                                        <Ans>1</Ans>
                                        <Ans>2</Ans>
                                        <Ans>3</Ans>
                                </Multi_Answer>
                        </Multi_anternative>
                </Question>
                <Question id="4">
                        <Text>Qual teu prato preferido?</Text>
                        <Textual_Answer>Feijoada</Textual_Answer>
                </Question>
        </Sub_Activity>
</Root_Activity>
```

APÊNDICE C

Métodos dos Componentes *Gamificação*

Este apêndice descreve os métodos dos componentes de *gamificação* integrantes da arquitetura proposta e os métodos das classes auxiliares.

C.1 Métodos do Componente *Narrative*

Tabela C.1: *Componente narrative método construtor*

	Descrição
Nome Método:	Narrative
Propósito:	Criar uma instância do componente Narrative.
Descrição:	Esse método cria uma instância do objeto representante do componente Narrative.
Assinatura:	Narrative(void)
Escopo:	Public
Retorno:	void
Parâmetros:	void

Tabela C.2: *Componente narrative método getNarrative*

	Descrição
Nome Método:	getNarrative
Propósito:	Buscar narrativas de ambas as categorias embarcada ou emergente.
Descrição:	Esse método retorna um objeto do tipo ArrayList<String> contendo narrativas embarcadas e emergentes.
Assinatura:	ArrayList<String>getNarrative(<ParserNarrative obj>)
Escopo:	Public
Retorno:	ArrayList<String>
Parâmetros:	ParserNarrative obj
obj	Contém a referência as narrativas lidas pelo módulo ParserXML

C.2 Métodos do Componente *Avatar*

Tabela C.3: *componente avatar método construtor*

	Descrição
Nome Método:	Avatar
Propósito:	Criar uma instância do componente Avatar.
Descrição:	Esse método cria uma instância do objeto representante do componente avatar.
Assinatura:	Avatar(void)
Escopo:	Publico
Retorno:	void
Parâmetros:	void

Tabela C.4: *componente avatar método setSpeak*

	Descrição
Nome Método	setSpeak
Propósito	Habilitar e desabilitar o som da voz na apresentação das narrativas.
Descrição	Esse método acessa o objeto ParserFeedback e altera o status do atributo audible, que ativa ou desativa o som da voz na apresentação das narrativas.
Assinatura	<void>setSpeak(<boolean status>)
Escopo	Público
Retorno	void
Parâmetros	boolean status
Status	Contém o valor que habilita ou desabilita a fala.

Tabela C.5: *componente avatar método FontSize*

	Descrição
Nome Método:	FontSize
Propósito:	Manipular o tamanho da fonte.
Descrição:	Esse método possibilita a redução e/ou ampliação do tamanho da fonte utilizada para apresentar as narrativas ao usuário.
Assinatura:	<void>FontSize(int size)
Escopo:	Público
Retorno:	void
Parâmetros:	int size
size	Parâmetro utilizado no redimensionamento do tamanho da fonte do texto das narrativas.

Tabela C.6: *componente avatar método FontColor*

	Descrição
Nome Método:	FontColor
Propósito:	Manipula a cor da fonte
Descrição:	Esse método possibilita a troca da cor da fonte utilizada no texto das narrativas.
Assinatura:	<void>FontColor(<int color>)
Escopo:	Público
Retorno:	void
Parâmetros:	int color
color	Parâmetro utilizado na alteração de cor da fonte da texto da narrativa.

Tabela C.7: *componente avatar método Timemsg*

	Descrição
Nome Método	Timemsg
Propósito	Dimensionar o tempo de exibição da narrativa.
Descrição	Esse método possibilita dimensionar a duração do tempo, que o texto da narrativa ficará visível no display.
Assinatura	<void>Timemsg(<int time>)
Escopo	Público
Retorno	void
Parâmetros	int time
time	Parâmetro que define a duração em segundos que as mensagens ficarão visíveis.

Tabela C.8: *componente avatar método Message*

	Descrição
Nome Método	Message
Propósito	Exibe a mensagem no display com a figura do avatar.
Descrição	Esse método exibe a mensagem de texto referente a narrativa no display e a figura do avatar.
Assinatura	<void>Message(<Context ctx>,<String msg>,<int img>)
Escopo	Público
Retorno	void
Parâmetros	Context ctx, String ms, int msg
msg	Parâmetro que recebe a mensagem de texto referente a narrativa.
ctx	Parâmetro que recebe o contexto da aplicação (activity).
img	Parâmetro que recebe a imagem representativa da figura do avatar.

Tabela C.9: *componente avatar método SpeakText*

	Descrição
Nome Método	SpeakText
Propósito	Realiza a leitura e a fala do texto de uma narrativa.
Descrição	Esse método é encarregado de realizar a leitura e síntese de voz das mensagens de texto mantidas na lista de narrativas.
Assinatura	<void>SpeakText(<Context ctx, String msg>)
Escopo	Público
Retorno	void
Parâmetros	Context ctx, String msg
ctx	Parâmetro que recebe o contexto da aplicação (activity).
msg	Parâmetro que recebe a mensagem a ser falada.

Tabela C.10: *componente avatar método setLanguage*

	Descrição
Nome Método:	setLanguage
Propósito:	Definir o idioma da fala.
Descrição:	Esse método possibilita a definição do idioma utilizado pelo sintetizador de voz na leitura das narrativas.
Assinatura:	<void>setLanguage(<Locale idioma>)
Escopo:	Público
Retorno:	void
Parâmetros:	Locale idioma
idioma	Parâmetro que contém um dos idiomas definidos na plataforma Andriod.

Tabela C.11: *componente avatar método getLanguage*

	Descrição
Nome Método:	getLanguage
Propósito:	Retornar o idioma da fala em uso.
Descrição:	Esse método retorna o idioma utilizado no sintetizador de voz para pronunciar as narrativas.
Assinatura:	<Locale>getLanguage(<void>)
Escopo:	Público
Retorno:	Locale
Parâmetros:	void

Tabela C.12: *componente avatar método isSpeak*

	Descrição
Nome Método	isSpeak
Propósito	Retorna se o síntese de voz está ativa.
Descrição	Esse método retorna o status do atributo speak, que indica o estado ativo ou inativo do sintetizador de voz.
Assinatura	<boolean>isSpeak(<void>)
Escopo	Público
Retorno	boolean
Parâmetros	void

Tabela C.13: *componente avatar método setImage*

	Descrição
Nome Método:	setImage
Propósito:	Alterar a imagem do avatar
Descrição:	Esse método permite redefinir a imagem utilizada pelo avatar definida pelo objeto que representa a atividade.
Assinatura:	<void>setImage(<ParserAvatar img>)
Escopo:	Público
Retorno:	void
Parâmetros:	ParserAvatar img
img	Parâmetro que altera a imagem do avatar

C.3 Métodos do Componente *Feedback*

Tabela C.14: *Componente feedback método construtor*

	Descrição
Nome Método:	Feedback
Propósito:	Criar uma instância do componente feedback.
Descrição:	Esse método cria uma instância do objeto representante do componente feedback.
Assinatura:	Feedback(void)
Escopo:	Public
Retorno:	void
Parâmetros:	void

Tabela C.15: *Componente feedback método AvatarExist*

	Descrição
Nome Método:	AvatarExist
Propósito:	Verificar a existência de uma instância do avatar.
Descrição:	Esse método verifica se uma instância do componente avatar foi criada pela atividade
Assinatura:	<boolean>AvatarExist(<void>)
Escopo:	Público
Retorno:	boolean
Parâmetros:	void

Tabela C.16: *Componente feedback método FontSize*

	Descrição
Nome Método:	FontSize
Propósito:	Manipular o tamanho da fonte.
Descrição:	Esse método possibilita a redução e/ou ampliação do tamanho da fonte utilizada para apresentar as narrativas ao usuário.
Assinatura:	<void>FontSize(int size)
Escopo:	Público
Retorno:	void
Parâmetros:	int size
size	Parâmetro utilizado no redimensionamento do tamanho da fonte do texto das narrativas.

Tabela C.17: *Componente feedback método FontColor*

	Descrição
Nome Método:	FontColor
Propósito:	Manipula a cor da fonte.
Descrição:	Esse método possibilita a troca da cor da fonte utilizada no texto das narrativas.
Assinatura:	<void>FontColor(<int color>)
Escopo:	Público
Retorno:	void
Parâmetros:	int color
color	Parâmetro utilizado na alteração de cor da fonte da texto da narrativa.

Tabela C.18: *Componente feedback método Timemsg*

	Descrição
Nome Método	Timemsg
Propósito	Dimensionar o tempo de exibição da narrativa.
Descrição	Esse método possibilita dimensionar a duração do tempo, que o texto da narrativa ficará visível no display.
Assinatura	<void>Timemsg(<int time>)
Escopo	Público
Retorno	void
Parâmetros	int time
time	Parâmetro define a duração em segundos que o texto da narrativa ficará visível no display.

Tabela C.19: *Componente feedback método BackgroundColor*

	Descrição
Nome Método:	BackgroundColor
Propósito:	Alterar a cor de fundo da narrativa.
Descrição:	Esse método permite definir a cor de fundo do texto da narrativa presentado no display.
Assinatura:	<void>BackgroundColor(<int color>)
Escopo:	Público
Retorno:	void
Parâmetros:	int color
color	Parâmetro que possibilita a alteração da cor do fundo do texto da narrativa.

Tabela C.20: *Componente feedback método Message*

	Descrição
Nome Método	Message
Propósito	Exibe a mensagem no display.
Descrição	Esse método exibe a mensagem de texto referente a narrativa no display.
Assinatura	<void>Message(<Context ctx>,<String msg>)
Escopo	Público
Retorno	void
Parâmetros	Context ctx, String msg
msg	Parâmetro que recebe a mensagem de texto referente a narrativa.
ctx	Parâmetro que recebe o contexto da aplicação (activity)

Tabela C.21: *Componente feedback método SendMessageAvatar*

	Descrição
Nome Método:	SendMessageAvatar
Propósito:	Enviar a mensagem de texto referente a narrativa ao avatar.
Descrição:	Esse método envia para o avatar a mensagem de texto selecionada para feedback gerada pela narrativa.
Assinatura:	<String>SendMessageAvatar(<void>)
Escopo:	Público
Retorno:	String
Parâmetros:	void

Tabela C.22: *Componente feedback método SeekNarrative*

Nome Método	SeekNarrative
Propósito	Buscar narrativas de ambas as categorias embarcada ou emergente.
Descrição	Esse método retorna um objeto do tipo String contendo uma narrativa embarcada ou emergente.
Assinatura	<String>SeekNarrative(<ArrayList<String> obj>,<int tipo>,<int ordem>)
Escopo	Publico
Retorno	String
Parâmetros	ArrayList<String> obj, int tipo, int ordem
obj	Contêm a referência as narrativas lidas pelo módulo ParserXML.
tipo	Tipo da narrativa (0 - Embarcada, 1,2,3,4 - Emergente).
Ordem	Coresponde a 1ª, 2ª,3ª,..., narrativa do tipo especificado.

C.4 Métodos do Componente *Competition*

Tabela C.23: *Componente competition método construtor*

	Descrição
Nome Método	Competitions
Propósito	Cria a instância de um componente da competição.
Descrição	Esse método cria a instância de um objeto da classe Competition, que controla os níveis de dificuldade aos competidores do usuário da atividade.
Assinatura	Competition(<ParserCompetition parsercompetition>)
Escopo	Público
Retorno	void
Parâmetros	ParserCompetition parsercompetition
parsercompetition	Parâmetro que definine o nível de dificuldade a ser utilizado no mecanismo de handcap.

Tabela C.24: *Componente competition método setLevels*

	Descrição
Nome Método	setLevels
Propósito	Define e redefine os níveis de dificuldade para um competidor.
Descrição	Esse método define e redefine o nível de pontos necessários para que um concorrente mude de nível.
Assinatura	<void> setLevel (<ParserOfferEnsign offerensign, ParserReward reward, ParserLevel point, ParserQuestionTime qtime>)
Escopo	Público
Retorno	void
Parâmetros	ParserOfferEnsign offerensign, ParserReward reward, ParserLevel point, ParserQuestionTime qtime
offerensign	Parâmetro contêm a pontuação necessária para oferecimento de insígnias aos competidores.
reward	Parâmetro contém a pontuação necessária para o oferecimento de recompensas ao competidor.
point	Parâmetro contém a pontuação necessária ao competidor para mudança de nível.
qtime	Parâmetro que contém o tempo espera para a resposta do competidor.

Tabela C.25: *Componente competition método setRules*

	Descrição
Nome Método:	setRules
Propósito:	Define e redefine as regras para o competidor.
Descrição:	Esse método define e redefine as regras para realização de uma atividade por parte do competidor.
Assinatura:	<void>setRules(<Rules rule>)
Escopo:	Público
Retorno:	void
Parâmetros:	Rules rule
rule	Parâmetro contém as regras de de realização de uma atividade por parte do competidor.

Tabela C.26: *Componente competition método setDifficulty*

	Descrição
Nome Método:	setDifficulty
Propósito:	Define e redefine o nível de dificuldade para o competidor.
Descrição:	Esse método define e redefine o nível de dificuldade para realização de uma atividade por parte do competidor.
Assinatura:	<void>setDifficulty(<int difficult>)
Escopo:	Público
Retorno:	void
Parâmetros:	int difficult
difficult	Parâmetro contém o nível de dificuldade para o competidor.

Tabela C.27: *Componente competition método getDifficulty*

	Descrição
Nome Método:	getDifficult
Propósito:	Retorna o nível e dificuldade de execução de uma atividade.
Descrição:	Esse método retorna o nível de dificuldade para a realização de uma atividade por parte do competidor.
Assinatura:	<int>getDifficult(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.28: *Componente competition método getCompetition*

	Descrição
Nome Método:	getCompetition
Propósito:	Retorna a configuração da atividade do competidor.
Descrição:	Esse método retorna a configuração da realização de uma atividade por parte do competidor.
Assinatura:	<Competition>getCompetition(<void>)
Escopo:	Público
Retorno:	Competitions
Parâmetros:	void

Tabela C.29: *Componente competition método getRules*

	Descrição
Nome Método:	getRules
Propósito:	Retorna as regras da atividade definidas para o competidor.
Descrição:	Esse método retorna as regras definidas para a realização a atividade por parte do competidor.
Assinatura:	<Rules>getRules{<void>)
Escopo:	Público
Retorno:	Rules
Parâmetros:	void

Tabela C.30: *Componente competition método getLevels*

	Descrição
Nome Método:	getLevels
Propósito:	Retorna os percentuais associados aos níveis de dificuldade da atividade definidas para o competidor.
Descrição:	Esse método retorna os percentuais associadas ao níveis de dificuldades para a realização a atividade por parte do competidor.
Assinatura:	<Levels>getLevels{<void>)
Escopo:	Público
Retorno:	Levels
Parâmetros:	void

C.5 Métodos do Componente *Cooperation*

Tabela C.31: *Componente cooperation método construtor*

	Descrição
Nome Método:	Cooperation
Propósito:	Criar uma instância do componente cooperation.
Descrição:	Esse método cria uma instância do componente cooperation.
Assinatura:	Cooperation(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.32: *Componente cooperation método setName*

	Descrição
Nome Método	setName
Propósito	Definir e redefinir o nome da pessoa que recebeu cooperação.
Descrição	Esse método define e redefine o nome do último contato ligado a rede social do usuário, que recebeu cooperação do usuário.
Assinatura	<void>setName(<String nome>)
Escopo	Público
Retorno	void
Parâmetros	String nome
Nome	Parâmetro que contém o nome da última pessoa que recebeu cooperação do usuário.

Tabela C.33: *Componente cooperation método setEmail*

	Descrição
Nome Método	setEmail
Propósito	Definir e redefinir o e-mail da pessoa que recebeu cooperação.
Descrição	Esse método define e redefine o e-mail do último contato ligado a rede social do usuário, que recebeu cooperação do usuário.
Assinatura	<void>setEmail(<String email >)
Escopo	Público
Retorno	void
Parâmetros	String email
email	Parâmetro que contém o email da última pessoa que recebeu cooperação do usuário.

Tabela C.34: *Componente cooperation método getName*

	Descrição
Nome Método	getName
Propósito	Retorna o nome da pessoa que recebeu cooperação.
Descrição	Esse método retorna o nome do último contato ligado a rede social do usuário, que recebeu cooperação do usuário."
Assinatura	<String>getName(<void>)
Escopo	Público
Retorno	String
Parâmetros	void

Tabela C.35: *Componente cooperation método getEmail*

	Descrição
Nome Método	getEmail
Propósito	Retorna o email da pessoa que recebeu cooperação.
Descrição	Esse método retorna o email do último contato ligado a rede social do usuário, que recebeu cooperação do usuário.
Assinatura	<String>getEmail(<void>)
Escopo	Público
Retorno	String
Parâmetros	void

Tabela C.36: *Componente cooperation método getCooperation*

	Descrição
Nome Método	getCooperation
Propósito	Retorna os dados do última pessoa que recebeu cooperação.
Descrição	Esse método etornar os dados do último contato ligado a rede social do usuário, que recebeu cooperação do usuário.
Assinatura	<Cooperation>getCooperation(<void>)
Escopo	Público
Retorno	Cooperation
Parâmetros	void

Tabela C.37: *Componente cooperation método SendEmail*

	Descrição
Nome Método	sendEmail
Propósito	Envia um e-mail para a pessoa que recebeu cooperação.
Descrição	Esse método envia um e-mail para o nome do último contato ligado a rede social do usuário, que recebeu cooperação do usuário.
Assinatura	<void>sendEmail(<void>)
Escopo	Público
Retorno	void
Parâmetros	void

Tabela C.38: *Componente cooperation método CreateMessage*

	Descrição
Nome Método	CreateMessage
Propósito	Cria a mensagem de cooperação.
Descrição	Esse método cria a mensagem a ser enviada ao último contato ligado a rede social do usuário, que recebeu cooperação do usuário.
Assinatura	<void>createMessage(<String message>)
Escopo	Público
Retorno	void
Parâmetros	String message
message	Parâmetro que contém a mensagem a ser transferida via e-mail ao último contato do usuário.

C.6 Métodos do Componente *Relationship*

Tabela C.39: *Componente relationship método construtor*

	Descrição
Nome Método:	Relationship
Propósito:	Criar uma instância do componente relationship.
Descrição:	Esse método cria uma instância do componente relationship.
Assinatura:	Relationship(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.40: *Componente relationship método UpDateRelationship*

	Descrição
Nome Método:	upDateRelationship
Propósito:	Atualiza uma ocorrência de relacionamento.
Descrição:	Esse método atualiza uma ocorrência na base de dados relativo aos relacionamentos do usuário.
Assinatura:	<void>upDateRelationship(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.41: *Componente relationship método getRelationship*

	Descrição
Nome Método	getRelationship
Propósito	Retorna uma ocorrência de relacionamento.
Descrição	Esse método retorna uma ocorrência na base de dados relativo aos relacionamentos do usuário.
Assinatura	<Relationship>getRelationship(<String name, String kinship>)
Escopo	Público
Retorno	Relationship
Parâmetros	String name, String kinship
name	Parâmetro utilizado para localizar uma pessoa da rede social do usuário
kinship	Parâmetro utilizado para localizar o tipo de parentesco da pessoa da rede social do usuário.

Tabela C.42: *Componente relationship método InsertRelationship*

	Descrição
Nome Método:	InsertRelationship
Propósito:	Insere uma ocorrência de relacionamento.
Descrição:	Esse método insere uma ocorrência na base de dados relativo aos relacionamentos do usuário.
Assinatura:	<void>InsertRelationship(<Relationship relationship>)
Escopo:	Público
Retorno:	void
Parâmetros:	Relationship relationship
relationship	Parâmetro utilizado para inserir um novo relacionamento na rede social do usuário.

Tabela C.43: *Componente relationship método setName*

	Descrição
Nome Método:	setName
Propósito:	Definir e redefinir o nome da pessoa participante das relações do usuário.
Descrição:	Esse método define e redefine o nome da pessoa integrante da rede social do usuário.
Assinatura:	<void>setName(<String nome>)
Escopo:	Público
Retorno:	void
Parâmetros:	String nome
nome	Parâmetro que contém o nome da pessoa a ser inserida na rede de relacionamentos do usuário.

Tabela C.44: *Componente relationship método setEmail*

	Descrição
Nome Método:	setEmail
Propósito:	Definir e redefinir o email da pessoa participante das relações do usuário.
Descrição:	Esse método define e redefine o email da pessoa integrante da rede social do usuário.
Assinatura:	<void>setEmail(<String email>)
Escopo:	Público
Retorno:	void
Parâmetros:	String email
email	Parâmetro que contém o email da pessoa a ser inserida na rede de relacionamentos do usuário.

Tabela C.45: *Componente relationship método setPhone*

	Descrição
Nome Método:	setPhone
Propósito:	Definir e redefinir o telefone da pessoa participante das relações do usuário.
Descrição:	Esse método define e redefine o telefone da pessoa integrante da rede social do usuário.
Assinatura:	<void>setPhone(<String phone>)
Escopo:	Público
Retorno:	void
Parâmetros:	String phone
phone	Parâmetro que contém o phone da pessoa a ser inserida na rede de relacionamentos do usuário.

Tabela C.46: *Componente relationship método setKinship*

	Descrição
Nome Método:	setKinship
Propósito:	Definir e redefinir o parentesco da pessoa participante das relações do usuário.
Descrição:	Esse método define e redefine o parentesco da pessoa integrante da rede social do usuário.
Assinatura:	<void>setKinship(<String kinship>)
Escopo:	Público
Retorno:	void
Parâmetros:	String kinship
kinship	Parâmetro que contém o parentesco da pessoa a ser inserida na rede de relacionamentos do usuário.

Tabela C.47: *Componente relationship método setId*

	Descrição
Nome Método:	setId
Propósito:	Definir e redefinir o identificador da pessoa participante das relações do usuário.
Descrição:	Esse método define e redefine o identificador da pessoa integrante da rede social do usuário.
Assinatura:	<void>setId(<int id>)
Escopo:	Público
Retorno:	void
Parâmetros:	int id
id	Parâmetro que contém o identificador da pessoa a ser inserida na rede de relacionamentos do usuário.

C.7 Métodos do Componente *Socialgraph*

Tabela C.48: *Componente socialgraph método construtor*

	Descrição
Nome Método:	SocialGraph
Propósito:	Criar uma instância do componente socialgraph.
Descrição:	Esse método cria uma instância do componente socialgraph.
Assinatura:	socialgraph(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.49: *Componente socialgraph método upDateSocialgraph*

	Descrição
Nome Método:	upDateSocialgraph
Propósito:	Atualiza uma ocorrência da rede social.
Descrição:	Esse método atualiza uma ocorrência na base de dados relativo a rede social do usuário.
Assinatura:	<void>upDateSocialgraph(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.50: *Componente socialgraph método getSocialgraph*

	Descrição
Nome Método:	getSocialgraph
Propósito:	Retorna a rede social do usuário.
Descrição:	Esse método retorna a rede social do usuário registrada na base de dados.
Assinatura:	<ArrayList<socialgraph»getSocialgraph(<void>)
Escopo:	Público
Retorno:	ArrayList<SocialGraph>
Parâmetros:	void

Tabela C.51: *Componente socialgraph método InsertSocialgraph*

	Descrição
Nome Método:	InsertSociagraph
Propósito:	Insere uma ocorrência do gráfico social.
Descrição:	Esse método insere uma ocorrência na base de dados relativo ao gráfico social do usuário.
Assinatura:	<void>Insertsocialgraph(<Socialgraph socialgraph>)
Escopo:	Público
Retorno:	void
Parâmetros:	SocialGraph socialgraph
socialgraph	Parâmetro utilizado para inserir uma nova ocorrência na rede social do usuário.

Tabela C.52: *Componente socialgraph método DisplaySocialgraph*

	Descrição
Nome Método:	DisplaySocialgraph
Propósito:	Mostra a rede social do usuário.
Descrição:	Esse método mostra a rede social do usuário.
Assinatura:	<void>DisplaySocialgraph(<ArrayList<Socialgraph> socialgraph>)
Escopo:	Público
Retorno:	void
Parâmetros:	ArrayList<Socialgraph> socialgraph
socialgraph	Parâmetro contém a rede social do usuário.

Tabela C.53: *Componente socialgraph método setId*

	Descrição
Nome Método:	setId
Propósito:	Definir e redefinir o identificador da rede de relações do usuário.
Descrição:	Esse método define e redefine o identificador da rede social do usuário.
Assinatura:	<void>setId(<int id>)
Escopo:	Público
Retorno:	void
Parâmetros:	int id
id	Parâmetro que contém o identificador da rede de relações do usuário a ser inserida.

Tabela C.54: *Componente socialgraph método setFkid*

	Descrição
Nome Método:	setFkid
Propósito:	Definir e redefinir o identificador da pessoa participante das relações do usuário.
Descrição:	Esse método define e redefine o identificador da pessoa integrante da rede social do usuário.
Assinatura:	<void>setFkid(<int fkid>)
Escopo:	Público
Retorno:	void
Parâmetros:	int fkid
fkid	Parâmetro que contém o identificador da pessoa a ser inserida na rede de relacionamentos do usuário.

Tabela C.55: *Componente socialgraph método setCooperation*

	Descrição
Nome Método:	setCooperation
Propósito:	Definir e redefinir a quantidade de cooperações da pessoa participante das relações do usuário.
Descrição:	Esse método define e redefine a quantidade de cooperações da pessoa integrante da rede social do usuário.
Assinatura:	<void>setCooperation(<int cooperation>)
Escopo:	Público
Retorno:	void
Parâmetros:	int cooperation
cooperation	Parâmetro que contém a quantidade de cooperações da pessoa a ser inserida na rede de relacionamentos do usuário.

Tabela C.56: *Componente socialgraph método setDtCooperation*

	Descrição
Nome Método:	setDtCooperation
Propósito:	Definir e redefinir a data da última cooperação da pessoa participante das relações do usuário.
Descrição:	Esse método define e redefine a data da última cooperação da pessoa integrante da rede social do usuário.
Assinatura:	<void>setDtcooperation(<Date dtcooperation>)
Escopo:	Público
Retorno:	void
Parâmetros:	Date dtcooperation
dtcooperation	Parâmetro que contém a data da última cooperação da pessoa a ser inserida na rede de relacionamentos do usuário.

Tabela C.57: *Componente socialgraph método setDtCompetition*

	Descrição
Nome Método:	setDtCompetition
Propósito:	Definir e redefinir a data da última disputa da pessoa participante das relações do usuário.
Descrição:	Esse método define e redefine a data da última disputa da pessoa integrante da rede social do usuário.
Assinatura:	<void>setDtcompetition(<Date dtcompetition>)
Escopo:	Público
Retorno:	void
Parâmetros:	Date dtcooperation
dtcompetition	Parâmetro que contém a data da última disputa da pessoa a ser inserida na rede de relacionamentos do usuário.

C.8 Métodos do Componente *Points*

Tabela C.58: *Componente points método construtor*

	Descrição
Nome Método:	Points
Propósito:	Instanciar o objeto do tipo points.
Descrição:	Esse método permite criar uma instância de objeto do componente Points.
Assinatura:	Points(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.59: *Componente points método Score*

	Descrição
Nome Método:	Score
Propósito:	Retornar a pontuação do usuário.
Descrição:	Esse método retorna a pontuação conseguida pelo usuário na realização de ações executadas dentro de uma atividade.
Assinatura:	<int>Score(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.60: *Componente points método IncrementCorrect*

	Descrição
Nome Método	IncrementCorrect
Propósito	Somar pontos ao score do usuário
Descrição	Esse método permite o incremento de pontos ao score do usuário relativo ao acerto na execução das ações propostas na atividade,
Assinatura	<void>IncrementCorrect(<ParserPoints points>)
Escopo	Público
Retorno	void
Parâmetros	ParsePoints points
points	Parâmetro que contêm a pontuação a ser aplicada na realização de ações de uma atividade.

Tabela C.61: *Componente points método IncrementError*

	Descrição
Nome Método	IncrementError
Propósito	Somar pontos ao score do usuário
Descrição	Esse método permite o incremento de pontos ao score do usuário relativo ao erro na execução das ações propostas na atividade,
Assinatura	<void>IncrementError<ParserPoints points>)
Escopo	Público
Retorno	void
Parâmetros	ParsePoints points
points	Parâmetro que contêm a pontuação a ser aplicada na realização de ações de uma atividade.

Tabela C.62: *Componente points método IsChangeScore*

C.9 Métodos do Componente *Progress*

	Descrição
Nome Método:	IsChangeScore
Propósito:	Detectar a mudança do score do usuário
Descrição:	Esse método detecta a mudança do score do usuário mediante a realização de uma ação pertencente a atividade.
Assinatura:	<boolean>IsChangeScore(<void>)
Escopo:	Público
Retorno:	boolean
Parâmetros:	void

Tabela C.63: *Componente progress método construtor*

	Descrição
Nome Método	Progress
Propósito	Criar a instância de um objeto progress.
Descrição	Esse método cria a instância do objeto progress, que registra a progressão do usuário no decorrer da atividade e a evolução da doença.
Assinatura	Progress(<void>)
Escopo	Público
Retorno	void
Parâmetros	void

Tabela C.64: *Componente progress método getMemory*

	Descrição
Nome Método:	getMemory
Propósito:	Retornar a pontuação associada a memória.
Descrição:	Esse método retorna a pontuação associada a função cognitiva memória.
Assinatura:	<int>getMemory(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.65: *Componente progress método getAtention*

	Descrição
Nome Método:	getAtention
Propósito:	Retornar a pontuação associada a atenção
Descrição:	Esse método retorna a pontuação relacionada a função cognitiva atenção.
Assinatura:	<int>getAtention(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.66: *Componente progress método getLanguage*

	Descrição
Nome Método:	getLanguage
Propósito:	Retornar a pontuação associada a linguagem
Descrição:	Esse método retorna a pontuação relacionada a função cognitiva linguagem.
Assinatura:	<int>getLanguage(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.67: *Componente progress método getExecutiveFunction*

	Descrição
Nome Método:	getExecuteFunction
Propósito:	Retornar a pontuação associada as funções executivas.
Descrição:	Esse método retorna a pontuação relacionada a função cognitiva funções executivas.
Assinatura:	<int>getExecutiveFunction(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.68: *Componente progress método getOrientation*

	Descrição
Nome Método:	getOrientation
Propósito:	Retornar a pontuação associada a orientação
Descrição:	Esse método retorna a pontuação relacionada a função cognitiva orientação.
Assinatura:	<int>getOrientation(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.69: *Componente progress método getPercpt*

	Descrição
Nome Método:	getPercept
Propósito:	Retornar a pontuação associada a percepção.
Descrição:	Esse método retorna a pontuação relacionada a função cognitiva percepção.
Assinatura:	<int>getPerception(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.70: *Componente progress método getProgress*

	Descrição
Nome Método:	getProgress
Propósito:	Retornar a pontuação associada a progressão do usuário.
Descrição:	Esse método retorna a pontuação relacionada a progressão da saúde do usuário.
Assinatura:	<Progress>getProgress(<void>)
Escopo:	Público
Retorno:	Progress
Parâmetros:	void

Tabela C.71: *Componente progress método getProgressPatient*

	Descrição
Nome Método:	getProgressPatient
Propósito:	Retornar os feitos do usuário.
Descrição:	Esse método retorna os feitos do usuário relativo as conquistas das ações na atividade.
Assinatura:	<Gamification>getProgressPatient(<void>)
Escopo:	Público
Retorno:	Gamification
Parâmetros:	void

Tabela C.72: *Componente progress método setMemory*

	Descrição
Nome Método:	setMemory
Propósito:	Definir a pontuação associada a memória.
Descrição:	Esse método redefine a pontuação relacionada a função cognitiva memória.
Assinatura:	<void>setMemory(<int memory>)
Escopo:	Público
Retorno:	void
Parâmetros:	int memory
memory	Parâmetro define o valor da pontuação relacionada a função cognitiva memória.

Tabela C.73: *Componente progress método setAtention*

	Descrição
Nome Método:	setAtention
Propósito:	Definir a pontuação associada a atenção.
Descrição:	Esse método redefine a pontuação relacionada a função cognitiva atenção.
Assinatura:	<void>setAtention(<int atention>)
Escopo:	Público
Retorno:	void
Parâmetros:	int atention
atention	Parâmetro define o valor da pontuação relacionada a função cognitiva atenção.

Tabela C.74: *Componente progress método setLanguage*

	Descrição
Nome Método:	setLanguage
Propósito:	Definir a pontuação associada a linguagem.
Descrição:	Esse método redefine a pontuação relacionada a função cognitiva linguagem.
Assinatura:	<void>setlanguage(<int language>)
Escopo:	Público
Retorno:	void
Parâmetros:	int language
language	Parâmetro define o valor da pontuação relacionada a função cognitiva linguagem.

Tabela C.75: *Componente progress método setExecutiveFunction*

	Descrição
Nome Método:	setExecutiveFunction
Propósito:	Definir a pontuação associada as funções executivas.
Descrição:	Esse método redefine a pontuação relacionada as funções executivas.
Assinatura:	<void>setExecutiveFunction(<int execfunction>)
Escopo:	Público
Retorno:	void
Parâmetros:	int execfunction
execfunction	Parâmetro define o valor da pontuação relacionada as funções executivas.

Tabela C.76: *Componente progress método setOrietation*

	Descrição
Nome Método:	setOrientation
Propósito:	Definir a pontuação associada a orientação.
Descrição:	Esse método redefine a pontuação relacionada a função cognitiva orientação.
Assinatura:	<void>setOrientation(<int orientation>)
Escopo:	Público
Retorno:	void
Parâmetros:	int orientation
orientation	Parâmetro define o valor da pontuação relacionada a função cognitiva orientação.

Tabela C.77: *Componente progress método setPercept*

	Descrição
Nome Método:	setPercept
Propósito:	Definir a pontuação associada a percepção
Descrição:	Esse método redefine a pontuação relacionada a função cognitiva percepção.
Assinatura:	<void>setPercept(<int percept>)
Escopo:	Público
Retorno:	void
Parâmetros:	int percept
percept	Parâmetro define o valor da pontuação relacionada a função cognitiva memória.

Tabela C.78: *Componente progress método MemCalculator*

	Descrição
Nome Método:	MeemCalculator
Propósito:	Calcular o índice MEEM do usuário.
Descrição:	Esse método permite calcular o índice MEEM relacionado a evolução da saúde do usuário.
Assinatura:	<Progress>MeemCalculator(<void>)
Escopo:	Público
Retorno:	Progress
Parâmetros:	void
percept	Parâmetro define o valor da pontuação relacionada a função cognitiva memória.

C.10 Métodos do Componente *Levels*

Tabela C.79: *Componente levels método construtor*

	Descrição
Nome Método:	Levels
Propósito:	Instanciar objetos da classe Levels.
Descrição:	Esse método permite criar instâncias de objetos da classe Levels.
Assinatura:	Levels(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.80: *Componente levels método UpLevelPointsEnsign*

	Descrição
Nome Método:	UpLevelPointsEnsign
Propósito:	Incrementar o nível de dificuldade para obtenção de insígnias.
Descrição:	Esse método amplia as exigências em pontos necessárias para conquista de insígnias pelo usuário.
Assinatura:	<int>UpLevelPointsEnsign(<ParserLevels obj>)
Escopo:	Público
Retorno:	int
Parâmetros:	ParserLevels obj:
obj:	Parâmetro que contêm a a pontuação necessária para a mudança de nível e a obtençao das insígnias.

Tabela C.81: *Componente levels método DownLevelPointsEnsign*

	Descrição
Nome Método:	DownLevelPointsEnsign
Propósito:	Decrementa o nível para obtenção de insígnias.
Descrição:	Esse método reduz as exigências em pontos necessárias para a conquista de insígnias pelo usuário.
Assinatura:	<int>DownLevelPointsEnsign(<ParserLevels obj>)
Escopo:	Público
Retorno:	int
Parâmetros:	ParserLevels obj:
obj:	Parâmetro que contêm a a pontuação necessária para a mudança de nível e a obtenção das insígnias.

Tabela C.82: *Componente levels método UpLevelTime*

	Descrição
Nome Método	UpLevelTime
Propósito	Incrementar o tempo de espera, para a realização de ações do usuário.
Descrição	Esse método amplia o tempo de espera, para a realização de uma ação por parte do usuário. Caso o tempo se esgote a ação é considerada não realizada de forma correta.
Assinatura	<void>UpLevelTime(<ParserQuestionTime obj>)
Escopo	Público
Retorno	void
Parâmetros	ParserQuestionTime obj
obj	Parâmetro que contém o tempo de espera para realização de uma ação pertencente a uma atividade.

Tabela C.83: *Componente levels método DownlevelTime*

	Descrição
Nome Método	DownLevelTime
Propósito	Reduz o tempo de espera, para a realização de ações do usuário.
Descrição	Esse método reduz o tempo de espera, para a realização de uma ação por parte do usuário. Caso o tempo se esgote a ação é considerada não realizada de forma correta.
Assinatura	<void>DownLevelTime(<ParserQuestionTime obj>)
Escopo	Público
Retorno	void
Parâmetros	ParserQuestionTime obj
obj	Parâmetro que contém o tempo de espera para realização de uma ação pertencente a uma atividade.

Tabela C.84: *Componente levels método UpLevelOfferEnsign*

	Descrição
Nome Método:	UpLevelOfferEnsign
Propósito:	Amplia os níveis de exigência para o recebimento de recompensas
Descrição:	Esse método amplia as exigências de pontuação para o oferecimento de recompensas.
Assinatura:	<void>UpLevelOfferEnsign(<ParserParserOfferensign obj>)
Escopo:	Público
Retorno:	void
Parâmetros:	ParserParserOfferensign obj
obj:	Parâmetro que determina o nível de pontuação necessário para o oferecimento de recompensas

Tabela C.85: *Componente levels método DownLevelOfferEnsign*

	Descrição
Nome Método:	DownLevelOfferEnsign
Propósito:	Reduz os níveis de exigência para o recebimento de recompensas.
Descrição:	Esse método reduz as exigências de pontuação para o oferecimento de recompensas.
Assinatura:	<void>DownLevelOfferEnsign(<ParserParserOfferensign obj>)
Escopo:	Público
Retorno:	void
Parâmetros:	ParserParserOfferensign obj
obj:	Parâmetro que determina o nível de pontuação necessário para o oferecimento de recompensas

Tabela C.86: *Componente levels método UpLevel*

	Descrição
Nome Método:	UpLevel
Propósito:	Amplia o nível.
Descrição:	Esse método amplia o nível do usuário dentro da atividade.
Assinatura:	<void>UpLevel(<int level>)
Escopo:	Público
Retorno:	void
Parâmetros:	int
obj:	Parâmetro que determina o novo nível do usuário dentro da atividade.

C.10.1 Métodos da Classe *LevelData*

Tabela C.87: *Classe levelsdata método construtor*

	Descrição
Nome Método	LevelData
Propósito	Cria a instância de um objeto de classe.
Descrição	Método que cria uma instância do objeto LevelData, que identifica os níveis de dificuldade de um nível.
Assinatura	LevelData(<void>)
Escopo	Público
Retorno	void
Parâmetros	void

Tabela C.88: *Classe levelsdata método setPointrewardsrate*

	Descrição
Nome Método:	setPointrewardsrate
Propósito:	Define percentual de ajuste da pontuação para as recompensas.
Descrição:	Esse método define e redefine a taxa de pontuação aplicada na ampliação ou redução da dificuldade para obtenção de recompensas.
Assinatura:	<void>setPointrewardsrate(<double rate>)
Escopo:	Privado
Retorno:	void
Parâmetros:	double rate
rate:	Parâmetro que define a nova taxa de alteração dos níveis de pontuação para obtenção de recompensas.

Tabela C.89: *Classe levelsdata método setEnsignsrate*

	Descrição
Nome Método:	setEnsignsrate
Propósito:	Define percentual de ajuste da pontuação para as insígnias.
Descrição:	Esse método define e redefine a taxa de pontuação aplicada na ampliação ou redução da dificuldade para obtenção de insígnias.
Assinatura:	<void>setEnsignrate(<double rate>)
Escopo:	Privado
Retorno:	void
Parâmetros:	double rate
rate:	Parâmetro que define a nova taxa de alteração dos níveis de pontuação para obtenção de insígnias.

Tabela C.90: *Classe levelsdata método setTimerate*

	Descrição
Nome Método:	setTimerate
Propósito:	Define percentual de ajuste do tempo para realização das ações de uma atividade.
Descrição:	Esse método define e redefine a taxa de ajuste aplicada na ampliação ou redução do tempo de resposta do usuário nas ações propostas na atividade.
Assinatura:	<void>setTimerate(<double rate>)
Escopo:	PrivadoFigura 4.4. Sequência de funcionamento componente Points.
Retorno:	void
Parâmetros:	double rate
rate:	Parâmetro que define a nova taxa de alteração dos níveis de de tempo para a realização das ações de uma atividade.

Tabela C.91: *Classe levelsdata método setLevelid*

	Descrição
Nome Método:	setLevelid
Propósito:	Define o número que identifica o nível.
Descrição:	Esse método define e redefine o id do nível de uma atividade.
Assinatura:	<void>setLevelid(<int id>)
Escopo:	Privado
Retorno:	void
Parâmetros:	double rate
id:	Parâmetro que define o identificador que representa o número do nível da atividade.

Tabela C.92: *Classe levelsdata método getLevelid*

	Descrição
Nome Método:	getLevelid
Propósito:	Retorna o número que identifica o nível.
Descrição:	Esse método retorna o id do nível de uma atividade.
Assinatura:	<double>getLevelid(<void>)
Escopo:	Privado
Retorno:	double
Parâmetros:	void

Tabela C.93: *Classe levelsdata método getPointrewardsrate*

	Descrição
Nome Método:	getPointrewardsrate
Propósito:	Retorna a taxa percentual de redefinição da pontuação de obtenção das recompensas.
Descrição:	Esse método retorna a taxa percentual de pontuação para obtenção de recompensas definidas para a atividade.
Assinatura:	<double>gePointrewardsrate(<void>)
Escopo:	Privado
Retorno:	double
Parâmetros:	void

Tabela C.94: *Classe levelsdata método getEnsignrate*

	Descrição
Nome Método:	getEnsignsrate
Propósito:	Retorna a taxa percentual de redefinição da pontuação de obtenção das insígnias.
Descrição:	Esse método retorna a taxa percentual de pontuação para obtenção de insígnias definidas para a atividade.
Assinatura:	<double>geEnsignrate(<void>)
Escopo:	Privado
Retorno:	double
Parâmetros:	void

Tabela C.95: *Classe levelsdata método getTimerate*

	Descrição
Nome Método:	getTimerate
Propósito:	Retorna a taxa percentual de redefinição do tempo de resposta das ações de uma atividade.
Descrição:	Esse método retorna a taxa percentual do tempo de resposta necessários para a realização de uma ação ligada a uma atividade.
Assinatura:	<double>geTimerate(<void>)
Escopo:	Privado
Retorno:	double
Parâmetros:	void

C.11 Métodos do Componente *Missions*

Tabela C.96: *Componente missions método construtor*

	Descrição
Nome Método:	Missions
Propósito:	Criar uma instância do componente missões
Descrição:	Esse método cria uma instância de componente da classe missions.
Assinatura:	<void>Missions(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void
percept	Parâmetro define o valor da pontuação relacionada a função cognitiva memória.

Tabela C.97: *Componente missions método setMissionObjective*

	Descrição
Nome Método:	setMissionObjective
Propósito:	Definir o objetivo da atividade.
Descrição:	Esse método define o objetivo da missão a ser realizada dentro de uma atividade pelo usuário.
Assinatura:	<void>setMissionObjective(<String objective>)
Escopo:	Público
Retorno:	void
Parâmetros:	String objective
objective	Parâmetro define o texto a ser expresso como narrativa embutida ao usuário.

Tabela C.98: *Componente missions método getMissionObjective*

	Descrição
Nome Método:	getMissionObjective
Propósito:	Recupera o objetivo da missão relacionada a atividade.
Descrição:	Esse método recupera o objetivo da missão a ser realizada dentro de uma atividade pelo usuário.
Assinatura:	<ArrayList<String»getMissionObjective(<void>)
Escopo:	Público
Retorno:	ArrayList<String>
Parâmetros:	void
objective	Parâmetro define o texto a ser expresso como narrativa embutida ao usuário.

Tabela C.99: *Componente missions método setMissionStatement*

	Descrição
Nome Método:	setMissionStatement
Propósito:	Definir as instruções de funcionamento da atividade.
Descrição:	Esse método define as instruções de funcionamento da atividade a ser realizada pelo usuário.
Assinatura:	<void>setMissionStatement(<String statement>)
Escopo:	Público
Retorno:	void
Parâmetros:	String statement
statement	Parâmetro define o texto a ser expresso como narrativa embutida ao usuário.

Tabela C.100: *Componente missions método getMissionStatement*

	Descrição
Nome Método:	getMissionStatement
Propósito:	Recupera as instruções da missão relacionada a atividade.
Descrição:	Esse método recupera as instruções da missão a ser realizada dentro de uma atividade pelo usuário.
Assinatura:	<String>getMissionStatement(<void>)
Escopo:	Público
Retorno:	String
Parâmetros:	void
statement	Parâmetro define o texto a ser expresso como narrativa embutida ao usuário.

Tabela C.101: *Componente missions método setMissionFinish*

	Descrição
Nome Método:	setMissionFinish
Propósito:	Definir como a atividade será encerrada.
Descrição:	Esse método define a forma de encerramento da atividade a ser realizada pelo usuário.
Assinatura:	<void>setMissionFinish<String finish>)
Escopo:	Público
Retorno:	void
Parâmetros:	String finish
finish	Parâmetro define o texto a ser expresso como narrativa embutida ao usuário como final da atividade.

Tabela C.102: *Componente missions método setMissionTime*

	Descrição
Nome Método:	setMissionTime
Propósito:	Definir a duração de exibição do texto explicativo da missão da atividade.
Descrição:	Esse método define a duração de exibição do texto explicativo da missão a ser realizada dentro de uma atividade.
Assinatura:	<void>setMissionTime(<int time>)
Escopo:	Público
Retorno:	void
Parâmetros:	int time
time	Parâmetro que define o tempo de duração de exibição do texto explicativo da missão.

Tabela C.103: *Componente missions método getMissionTime*

	Descrição
Nome Método:	getMissionTime
Propósito:	Recupera a duração da exibição do texto da missão relacionada a atividade.
Descrição:	Esse método recupera o tempo de exibição das mensagens relativas a missão a ser realizada dentro de uma atividade.
Assinatura:	<int>getMissionTime(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

C.12 Métodos do Componente *Challenge*

Tabela C.104: *Componente challenge método costrutor*

	Descrição
Nome Método:	Challenges
Propósito:	Criar a instância de objetos da classe Challenge.
Descrição:	Esse método cria a instância do objeto challenge a ser superada dentro de uma atividade pelo usuário.
Assinatura:	Challenges(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.105: *Componente challenge método setErrorlimit*

	Descrição
Nome Método:	setErrorlimit
Propósito:	Definir e redefinir o limite de erros permitidos na atividade.
Descrição:	Esse método define e redefine o limite máximo aceitável de erros cometidos pelo usuário dentro da atividade.
Assinatura:	<void>setErrorlimit(<int erro>)
Escopo:	Público
Retorno:	void
Parâmetros:	int
erro	Parâmetro contém o limite aceitável de erros.

Tabela C.106: *Componente challenge método setCorrectminlimit*

	Descrição
Nome Método:	setCorrectminlimit
Propósito:	Definir e redefinir o limite mínimo de acertos na atividade.
Descrição:	Esse método define e redefine o limite mínimo de acertos aceito pelo usuário dentro da atividade.
Assinatura:	<void>setCorrectminlimit(<int correct>)
Escopo:	Público
Retorno:	void
Parâmetros:	int
correct	Parâmetro contém o limite mínimo de acerto aceitáveis.

Tabela C.107: *Componente challenge método setEndeveorlimits*

	Descrição
Nome Método:	SetEndeveorlimits
Propósito:	Definir e redefinir o limite máximos de tentativas.
Descrição:	Esse método define e redefine o limite máximo de tentativas que o usuário pode realizar a atividade.
Assinatura:	<void>SetEndeveorlimits(<int endeveor>)
Escopo:	Público
Retorno:	void
Parâmetros:	int
endeveor	Parâmetro contém o limite máximo de tentativas aceitáveis.

Tabela C.108: *Componente challenge método getErrorlimit*

	Descrição
Nome Método:	getErrorlimit
Propósito:	Retornar o limite de erros permitidos na atividade.
Descrição:	Esse método retorna o limite máximo aceitável de erros cometidos pelo usuário dentro da atividade.
Assinatura:	<int>getErrorlimit(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.109: *Componente challenge método getCorrectlimit*

	Descrição
Nome Método:	getCorrectminlimit
Propósito:	Retornar o limite mínimo de acertos na atividade.
Descrição:	Esse método retorna o limite mínimo de acertos aceito pelo usuário dentro da atividade.
Assinatura:	<int>getCorrectminlimit(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.110: *Componente challenge método getEndeveourlimit*

	Descrição
Nome Método:	getEndeveorlimits
Propósito:	Retornar o limite máximos de tentativas.
Descrição:	Esse método retorna o limite máximo de tentativas que o usuário pode realizar a atividade.
Assinatura:	<int>getEndeveorlimits(<void>)
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.111: *Componente challenge método getChallenge*

	Descrição
Nome Método:	getChallenge
Propósito:	Retornar uma instância de desafios.
Descrição:	Esse método retorna dados do objeto desafios
Assinatura:	<Challenges>getChallenge(<void>)
Escopo:	Público
Retorno:	Challenges
Parâmetros:	void

C.13 Métodos do Componente *Achievement*

Tabela C.112: *Componente achievement método construtor*

	Descrição
Nome Método:	Achievement
Propósito:	Criar instância do componente conquistas
Descrição:	Esse método cria a instância do componente conquistas.
Assinatura:	Achievement(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.113: *Componente achievement método setScore*

	Descrição
Nome Método:	setScore
Propósito:	Define e redefine o compartilhamento do atributo score
Descrição:	Esse método define e redefine a permissão de compartilhamento do score atingido pelo usuário na realização de uma atividade.
Assinatura:	<void>setScore(<boolean score>)
Escopo:	Público
Retorno:	void
Parâmetros:	boolean score
score	Parâmetro contém o valor que redefine a permissão de compartilhamento do score

Tabela C.114: *Componente achievement método setEnsign*

	Descrição
Nome Método:	setEnsign
Propósito:	Define e redefine o compartilhamento do atributo ensign.
Descrição:	Esse método define e redefine a permissão de compartilhamento das insígnias conquistadas pelo usuário na realização de uma atividade.
Assinatura:	<void>setEnsign(<boolean ensign>)
Escopo:	Público
Retorno:	void
Parâmetros:	boolean ensign
ensign	Parâmetro contém o valor que redefine a permissão de compartilhamento das insígnias.

Tabela C.115: *Componente achievement método setREward*

	Descrição
Nome Método:	setReward
Propósito:	Define e redefine o compartilhamento do atributo rewardn.
Descrição:	Esse método define e redefine a permissão de compartilhamento das recompensas conquistadas pelo usuário na realização de uma atividade.
Assinatura:	<void>setReward(<boolean reward>)
Escopo:	Público
Retorno:	void
Parâmetros:	Boolean reward
reward	Parâmetro contém o valor que redefine a permissão de compartilhamento das recompensas.

Tabela C.116: *Componente achievement método getScore*

	Descrição
Nome Método:	getScore
Propósito:	Retorna valor do compartilhamento do atributo score.
Descrição:	Esse método retorna a permissão de compartilhamento do score conquistado pelo usuário na realização de uma atividade.
Assinatura:	<boolean>getScore(<void>)
Escopo:	Público
Retorno:	boolean
Parâmetros:	void

Tabela C.117: *Componente achievement método getEnsign*

	Descrição
Nome Método:	getEnsign
Propósito:	Retorna valor do compartilhamento do atributo ensign.
Descrição:	Esse método retorna a permissão de compartilhamento das insígnias conquistadas pelo usuário na realização de uma atividade.
Assinatura:	<boolean>getEnsign(<void>)
Escopo:	Público
Retorno:	boolean
Parâmetros:	void

Tabela C.118: *Componente achievement método getReward*

	Descrição
Nome Método:	getReward
Propósito:	Retorna valor do compartilhamento do atributo reward.
Descrição:	Esse método retorna a permissão de compartilhamento das recompensas conquistadas pelo usuário na realização de uma atividade.
Assinatura:	<boolean>getReward<void>)
Escopo:	Público
Retorno:	boolean
Parâmetros:	void

Tabela C.119: *Componente achievement método getAchievement*

	Descrição
Nome Método:	getAchievements
Propósito:	Retorna valor dos compartilhamentos dos atributos definidos pelo objeto conquistas
Descrição:	Esse método retorna a permissão dos compartilhamentos do contidos no objeto conquistas.
Assinatura:	<Acheivements>getAcheivements(<void>)
Escopo:	Público
Retorno:	Acheivements
Parâmetros:	void

Tabela C.120: *Componente achievement método SendScore*

	Descrição
Nome Método	SendScore
Propósito	Compartilha a pontuação do usuário.
Descrição	Esse método compartilha a pontuação conquistada pelo usuário durante a realização de uma atividade.
Assinatura	<int>sendScore(<Context ctx, Points point>)
Escopo	Público
Retorno	int
Parâmetros	Context ctx, Points point
ctx	Parâmetro que contêm o contexto do activity de uma atividade.
point	Parâmetro que contêm os pontos conquistados pelo usuário.

Tabela C.121: *Componente achievement método SendReward*

	Descrição
Nome Método	SendReward
Propósito	Compartilha as recompensas do usuário.
Descrição	Esse método compartilha as recompensas conquistadas pelo usuário durante a realização de uma atividade.
Assinatura	<ArrayList<Stream»sendReward(<Context ctx, ParserReward reward, Reward indexs>)
Escopo	Público
Retorno	ArrayList<Stream>
Parâmetros	Context ctx, ParserReward reward, Reward indexs
ctx	Parâmetro que contêm o contexto do activity de uma atividade.
reward	Parâmetro que contêm as recompensas a serem oferecidas ao usuário.
indexs	Parâmetro que contêm a última recompensa conquistada pelo usuário.

Tabela C.122: *Componente achievement método SendEnsign*

	Descrição
Nome Método	SendEnsign
Propósito	Compartilha as insígnias do usuário.
Descrição	Esse método compartilha as insignias conquistadas pelo usuário durante a realização de uma atividade.
Assinatura	<ArrayList<Stream»sendEnsign(<Context ctx, ParserEnsign ensign, Ensign indexs>)
Escopo	Público
Retorno	ArrayList<Stream>
Parâmetros	Context ctx, ParserEnsign ensign, Ensign indexs
ctx	Parâmetro que contêm o contexto do activity de uma atividade.
ensign	Parâmetro que contêm as insígnias a serem oferecidas ao usuário.
indexs	Parâmetro que contêm a última insígnia conquistada pelo usuário.

C.14 Métodos do Componente *Ensign*

Tabela C.123: *Componente ensign método construtor*

	Descrição
Nome Método	Ensign
Propósito	Instanciar um objeto do tipo insígnias.
Descrição	Esse método cria uma instância do objeto insígnias, com as recompensas a serem oferecidas ao usuário.
Assinatura	Ensign<ParserEnsign ensign>)
Escopo	Público
Retorno	void
Parâmetros	ParserEnsign ensign
ensign	Parâmetro contém o objeto com as insígnias disponíveis a serem oferecidas na realização da atividade.

Tabela C.124: *Componente ensign método EnsignIssue*

	Descrição
Nome Método:	EnsignIssue
Propósito:	Exibir as Insignias conquistadas no display.
Descrição:	Esse método exibe no display as insignias conquistadas com a realização das ações propostas na atividade.
Assinatura:	<void>EnsignIssue(<Context ctx>)
Escopo:	Público
Retorno:	void
Parâmetros:	Context ctx
ctx	Parâmetro que contém a activity relacionada a atividade.

Tabela C.125: *Componente ensign métodoSeekEnsign*

	Descrição
Nome Método:	SeekEnsign
Propósito:	Busca a insígnia a ser oferecida ao usuário.
Descrição:	Esse método busca as insignias a serem oferecidas com a realização das ações propostas na atividade.
Assinatura:	<void>SeekEnsign(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.126: *Componente ensign método setNumberEnsign*

	Descrição
Nome Método:	setNumberEnsign
Propósito:	Redefine o número de insígnias oferecidas.
Descrição:	Esse método define e redefine o número de insígnias a oferecer ao usuário a medida que realiza as ações de uma atividade.
Assinatura:	<void>setNumberEnsign(<int numberensign>)
Escopo:	Público
Retorno:	void
Parâmetros:	int numberensign
numberensign	Parâmetro que redefine o número de insígnias a oferecer

Tabela C.127: *Componente ensign método getNumberEnsign*

	Descrição
Nome Método:	getNumberEnsign
Propósito:	Retorna o número de ensignias oferecidas.
Descrição:	Esse método retorna a quantidade de insígnias a serem oferecidas ao usuário a medida que realiza uma atividade.
Assinatura:	<int>getNumberEnsign(<void>
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.128: *Componente ensign método setCurrentEnsign*

	Descrição
Nome Método:	setCurrentEnsign
Propósito:	Redefine a insígnias atualmente a ser oferecida.
Descrição:	Esse método define e redefine a insígnia a ser oferecida ao usuário a medida que realiza as ações de uma atividade.
Assinatura:	<void>setCurrentEnsign(<int currentensign>)
Escopo:	Público
Retorno:	void
Parâmetros:	int currenterensign
numberensign	Parâmetro que redefine ainsígnia a ser oferecida.

Tabela C.129: *Componente ensign método getCurrentEnsign*

	Descrição
Nome Método:	getCurrentEnsign
Propósito:	Retorna o o indice da ensignia a ser oferecida.
Descrição:	Esse método retorna o índice atual da insígnia a ser oferecida ao usuário a medida que realiza uma atividade.
Assinatura:	<int>getCurrentEnsign(<void>
Escopo:	Público
Retorno:	int
Parâmetros:	void

C.15 Métodos do Componente *Rewards*

Tabela C.130: *Componente reward método construtor*

	Descrição
Nome Método	Rewards
Propósito	Instanciar um objeto do tipo recompensas.
Descrição	Esse método cria uma instância do objeto recompensa, com as recompensas a serem oferecidas ao usuário.
Assinatura	Rewards<ParserReward reward>)
Escopo	Público
Retorno	void
Parâmetros	ParserReward reward
reward	Parâmetro contém o objeto com as recompensas disponíveis a serem oferecidas na realização da atividade.

Tabela C.131: *Componente reward método RewardIssue*

	Descrição
Nome Método	RewardIssue
Propósito	Emitir no display a recompensa conquistada pelo usuário.
Descrição	Esse método permite no display a recompensa conquistada pelo usuário referente as suas ações dentro da atividade.
Assinatura	<void>RewardIssue<Context ctx, ArrayList<String> array)
Escopo	Público
Retorno	void
Parâmetros	Context ctx, ArrayList<String>
ctx	Parâmetro contém referência a activity que compões a atividade que está em execução.
array	Parâmetro que contém a lista de recompensas a serem oferecidas.

Tabela C.132: *Componente reward método SeekReward*

	Descrição
Nome Método:	SeekReward
Propósito:	Busca as recompensas a serem oferecidas.
Descrição:	Esse método retorna a lista de recompensas a serem oferecidas mediante as ações realizadas pelo usuário durante uma atividade.
Assinatura:	ArrayList<String>SeekReward(<ParserReward obj>)
Escopo:	Público
Retorno:	ArrayList<String>
Parâmetros:	ParserReward obj
obj	Parâmetro que contém as recompensas a serem oferecidas ao usuário.

Tabela C.133: *Componente reward método setNumberReward*

	Descrição
Nome Método:	setNumberReward
Propósito:	Redefine o número de recompensas oferecidas.
Descrição:	Esse método define e redefine o número de recompensas a oferecer ao usuário a medida que realiza as ações de uma atividade.
Assinatura:	<void>setNumberReward(<int numberreward>)
Escopo:	Público
Retorno:	void
Parâmetros:	int numberreward
numberensign	Parâmetro que redefine o número de recompensas a oferecer

Tabela C.134: *Componente reward método getNumberReward*

	Descrição
Nome Método:	getNumberReward
Propósito:	Retorna o número de recompensas oferecidas.
Descrição:	Esse método retorna a quantidade de recompensas a serem oferecidas ao usuário a medida que realiza uma atividade.
Assinatura:	<int>getNumberReward(<void>
Escopo:	Público
Retorno:	int
Parâmetros:	void

Tabela C.135: *Componente reward método setCurrentReward*

	Descrição
Nome Método:	setCurrentReward
Propósito:	Redefine a irecompensa atualmente a ser oferecida.
Descrição:	Esse método define e redefine a recompensa a ser oferecida ao usuário a medida que realiza as ações de uma atividade.
Assinatura:	<void>setCurrentReward(<int currentreward>)
Escopo:	Público
Retorno:	void
Parâmetros:	int currenterensign
numberensign	Parâmetro que redefine a recompensa a ser oferecida.

Tabela C.136: *Componente reward método getCurrentReward*

	Descrição
Nome Método:	getCurrentReward
Propósito:	Retorna o o indice da recompensa a ser oferecida.
Descrição:	Esse método retorna o índice atual da recompensa a ser oferecida ao usuário a medida que realiza uma atividade.
Assinatura:	<int>getCurrentReward(<void>
Escopo:	Público
Retorno:	int
Parâmetros:	void

C.16 Métodos do Componente *Rules*

Tabela C.137: *Componente rule método construtor*

	Descrição
Nome Método:	Rules
Propósito:	Instanciar objetos da classe Rules.
Descrição:	Esse método cria uma instância do objeto Rules.
Assinatura:	Rules(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.138: *Componente rule método CheckEnsign*

	Descrição
Nome Método	CheckEnsign
Propósito	Verificar se a pontuação conquistada pelo usuário é suficiente para atribuição de uma insígnia.
Descrição	Esse método verifica a pontuação conquistada pelo usuário sempre que uma atualização de pontuação acontece nas ações realizadas pelo usuário.
Assinatura	<boolean>CheckEnsign(<ParserOfferEnsign obj, Points obj2>)"
Escopo	Público
Retorno	boolean
Parâmetros	ParserOfferEnsign obj, Points obj2
obj	Parâmetro que acessa os níveis de pontuação necessários para atribuição das insígnias.
obj2	Parâmetro que acessa a pontuação conquistada pelo usuário até o presente momento.

Tabela C.139: *Componente rule método CheckRewards*

	Descrição
Nome Método:	CheckRewards
Propósito:	Verificar se a pontuação conquistada pelo usuário é suficiente para atribuição de recompensas.
Descrição:	Esse método verifica a pontuação conquistada pelo usuário sempre que uma atualização de pontuação acontece nas ações realizadas pelo usuário.
Assinatura:	<boolean>CheckRewards(<ParserOfferReward obj>)
Escopo:	Público
Retorno:	boolean
Parâmetros:	ParserOfferReward obj
obj	Parâmetro que acessa os níveis de pontuação necessários para atribuição das recompensas.

Tabela C.140: *Componente rule método CheckActionAbort*

	Descrição
Nome Método	CheckActionAbort
Propósito	Verificar a ação realizada pelo usuário
Descrição	Esse método verifica a ação realizada pelo usuário, se esta foi uma ação de desistência.
Assinatura	<boolean>CheckActionAbort(<void>)
Escopo	Público
Retorno	boolean
Parâmetros	void

Tabela C.141: *Componente rule método CheckActionAbortAlternate*

	Descrição
Nome Método	CheckActionAbortAlternate
Propósito	Verificar a ação realizada pelo usuário
Descrição	Esse método verifica a ação realizada pelo usuário, se esta foi uma ação de desistência de forma alternada.
Assinatura	<boolean>CheckActionAbortAlternate(<void>)
Escopo	Público
Retorno	boolean
Parâmetros	void

Tabela C.142: *Componente rule método CheckActionCorrect*

	Descrição
Nome Método	CheckActionCorrect
Propósito	Verificar a ação realizada pelo usuário
Descrição	Esse método verifica a ação realizada pelo usuário, se esta foi uma ação executada de modo correto.
Assinatura	<boolean>CheckActionCorrect(<void>)
Escopo	Público
Retorno	boolean
Parâmetros	void

Tabela C.143: *Componente rule método CheckActionError*

	Descrição
Nome Método	CkeckActionError
Propósito	Verificar a ação realizada pelo usuário.
Descrição	Esse método verifica a ação realizada pelo usuário, se esta foi uma ação executada de modo incorreto.
Assinatura	<boolean>CkeckActionError(<void>)
Escopo	Público
Retorno	boolean
Parâmetros	void

Tabela C.144: *Componente rule método ErrorIncrement*

	Descrição
Nome Método:	ErrorIncrement
Propósito:	Incrementa uma ação realizada incorretamente.
Descrição:	Esse método incrementa o número de ações realizadas pelo usuário de modo incorreto dentro de uma atividade.
Assinatura:	<void>ErrorIncrement<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.145: *Componente rules método CorrectIncrement*

	Descrição
Nome Método:	CorrectIncrement
Propósito:	Incrementa uma ação realizada corretamente.
Descrição:	Esse método incrementa o número de ações realizadas pelo usuário de modo correto dentro de uma atividade.
Assinatura:	<void>CorrectIncrement<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.146: *Componente rules método AbortaltIncrement*

	Descrição
Nome Método:	AbortaltIncrement
Propósito:	Incrementa uma ação realizada corretamente.
Descrição:	Esse método incrementa o número de ações de desistência da atividade realizadas pelo usuário de modo alternado.
Assinatura:	<void>AbortaltIncrementt<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.147: *Componente rules método AbortIncrement*

	Descrição
Nome Método:	AbortIncrement
Propósito:	Incrementa uma ação realizada corretamente.
Descrição:	Esse método incrementa o número de ações de desistência da atividade realizadas pelo usuário.
Assinatura:	<void>AbortIncrement<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.148: *Componente rules método CheckActionHelp*

	Descrição
Nome Método	CheckActionHelp
Propósito	Verificar a ação realizada pelo usuário
Descrição	Esse método verifica a ação realizada pelo usuário, se esta foi uma ação de solicitação de auxílio.
Assinatura	<boolean>CheckActionHelp(<void>)
Escopo	Público
Retorno	boolean
Parâmetros	void

Tabela C.149: *Componente rules método CheckActionReplay*

	Descrição
Nome Método	CheckActionReplay
Propósito	Verificar a ação realizada pelo usuário.
Descrição	Esse método verifica a ação realizada pelo usuário, se esta foi uma ação que permite ser repetida em caso de ser realizada incorretamente.
Assinatura	<boolean>CheckActionReplay(<void>)
Escopo	Público
Retorno	boolean
Parâmetros	void

Tabela C.150: *Componente rules método ReturnRuleCode*

	Descrição
Nome Método	ReturnRuleCode
Propósito	Retorna o código da ação realizada pelo usuário
Descrição	Esse método retorna o código da ação realizada pelo usuário.
Assinatura	<int>ReturnRuleCode(<void>)
Escopo	Público
Retorno	int
Parâmetros	void

Tabela C.151: *Componente rules método CheckLevelChange*

	Descrição
Nome Método	CheckLevelChange
Propósito	Verificar a mudança de nível de dificuldade na atividade.
Descrição	Esse método verifica se houve uma mudação de nível de dificuldade da atividade mediante ao resultado das ações do usuário.
Assinatura	<boolean>CheckLevelChange(<ParserLevel obj, Points obj2>)
Escopo	Público
Retorno	boolean
Parâmetros	ParserLevel obj, Points obj2
obj	Parâmetro que contém os valores de pontuação necessários para a mudança de nível e consequente premiação.
obj2	Parâmetro que contém os pontos conquistados pelo usuário até o momento.

Tabela C.152: *Componente rules método AbortInitial*

	Descrição
Nome Método:	Abortinitial
Propósito:	Reinicializar a contagem de ações de desistência.
Descrição:	Esse método inicializa a contagem de ações de desistência realizadas pelo usuário durante uma atividade.
Assinatura:	<void>Abortinitial(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.153: *Componente rules método AbortaltInitial*

	Descrição
Nome Método:	Abortaltinitial
Propósito:	Reinicializar a contagem de ações de desistência alternada.
Descrição:	Esse método inicializa a contagem de ações de desistência alternada realizadas pelo usuário durante uma atividade.
Assinatura:	<void>Abortaltinitial(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.154: *Componente rules método CorrectInitial*

	Descrição
Nome Método:	Correctinitial
Propósito:	Reinicializar a contagem de ações corretas.
Descrição:	Esse método inicializa a contagem de ações de corretas realizadas pelo usuário durante uma atividade.
Assinatura:	<void>Correctinitial(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	void

Tabela C.155: *Componente rules método ErrorInitial*

	Descrição
Nome Método:	Errorinitial
Propósito:	Reinicializar a contagem de ações erradas.
Descrição:	Esse método inicializa a contagem de ações erradas realizadas pelo usuário durante uma atividade.
Assinatura:	<void>Errorinitial(<void>)
Escopo:	Público
Retorno:	void
Parâmetros:	Void

Tabela C.156: *Componente rules método getRulesAction*

	Descrição
Nome Método	getRulesAction
Propósito	Retornar dados relativos as ações do usuário
Descrição	Esse Método retorna os dados dos acertos, erros, desistências e auxlios relativos a execução de uma atividade.
Assinatura	<Rules>getRulesAction()
Escopo	Público
Retorno	Rules
Parâmetros	void

C.17 Métodos do Componente *Hability*

Tabela C.157: *Componente hability método InsertAction*

	Descrição
Nome Método:	InsertAction
Propósito:	Persiste o resultados das ações do usuário no banco de dados.
Descrição:	Esse método registra os dados relativos a execução de uma atividade na base de dados.
Assinatura:	<void>InsertAction(<Rules rule>)
Escopo:	Público
Retorno:	void
Parâmetros:	Rules rule
rules	Parâmetro que contêm o resultado da execução das ações do usuário em uma atividade.

Tabela C.158: *Componente hability método SeekAction*

	Descrição
Nome Método:	SeekAction
Propósito:	Retorna os dados relativo as ações do usuário.
Descrição:	Esse método retorna os dados relativos as ações do usuário realizadas durante uma atividade.
Assinatura:	<Rules>SeekAction(<void>)
Escopo:	Público
Retorno:	Rules
Parâmetros:	Void

Tabela C.159: *Componente hability método DisplayAction*

	Descrição
Nome Método:	DisplayAction
Propósito:	Gera o relatório com as ações do usuário.
Descrição:	Esse método gera o relatório relativo as ações do usuário durante a execução de uma atividade.
Assinatura:	<void>DisplayAction(<Rules rule>)
Escopo:	Público
Retorno:	void
Parâmetros:	Rules rule
rule	Parâmetro que contêm oas ações do usuário executadas em uma atividade.

APÊNDICE D

Descrição das Personas

Este apêndice é possível visualizar a persona utilizada para representação dos pacientes acometidos pela doença de Alzheimer.

D.1 Personas

Descrição do perfil do paciente 01.
Nome: Antônio Medeiros de Lima
Idade Atual: 74 anos
Sexo: Masculino
Estado civil: Viúvo
Tempo estado civil atual: 2 anos
Raça: Calcasiana
Data diagnóstico: 6 meses
Pontuação MEEM: 27
Estágio da doença: Inicial
Problemas relatados: Lápsos de memória recente, confusão linguagem (dificuldade de encontrar palavras, troca de contexto), desorientação no tempo e no espaço, dificuldade para tomar decisões, perda de iniciativa e de motivação, sinais de depressão, agressividade, diminuição do interesse por atividades e passatempos.
Atividade que exercia: Engenheiro civil
Atividade que exerce: Aposentado, frequenta grupos da terceira idade (dança, jogos de carta, tabuleiro, sessões de cinema), faz as compras da casa geralmente sozinho.
Hobby: jogos com cartas, jornais, revistas, palavras cruzadas, cinema.
Filhos: 3
Moram próximos: 2 na mesma cidade do paciente e 1 em outro estado.
Netos: Sim 1 neta idade 8 anos.
Visita filhos e netos: Os filhos próximos visitam o pai todos os dias no final da tarde.
Datas comemorativas: Natal e Aniversário do paciente.
Mora sozinho: Sim

Atividades da casa: Realizada por uma empregada doméstica.

Descrição do perfil do paciente 02.

Nome: Elizabth Andresien
Idade Atual: 73 anos
Sexo: Feminino
Estado civil: Viúva
Tempo estado civil atual: 3 anos
Raça: Calcasiana
Data diagnóstico: 1 ano
Pontuação MEEM: 26
Estágio da doença: Inicial
Problemas relatados: Lápsos de memória recente, desorientação espaço temporal em alguns momentos e desatenção na realização de atividades da vida diária como: pagamento de contas, esquecimento de compromissos, objetos deixados em locais não apropriados.
Atividade que exercia: Secretária BilÃngue e professora de piano
Atividade que exerce: Aposentada, frequenta grupos da terceira idade (dança), faz as compras da casa acompanhada pela cuidadora.
Hobby: Tocar piano, ouvir mÃºsica clássica, jornais, revistas, palavras cruzadas.
Filhos: 2
Moram próximos: 2 na mesma cidade do paciente.
Netos: Não
Visita filhos e netos: Os filhos próximos visitam a mãe todos os dias no final da tarde.
Datas comemorativas: Natal e Aniversário do paciente.
Mora sozinho: Não, acompanhada por uma cuidadora.
Atividades da casa: Realizada pela paciente e cuidadora.

APÊNDICE E

Cognição e Sentidos

Este apêndice são listadas as restrições/habilidades comuns em indivíduos idosos.

E.1 Restrições/Habilidades

As habilidades e/ou restrições estão relacionadas as capacidades ou limitações existentes no paciente. As habilidades/restrições foram categorizadas em visual, auditiva, motora, cognitiva e emocional.

- **Visual:** Está relacionada a capacidade do usuário distinguir e identificar os elementos gráficos, textuais, contrastes das *interfaces* da aplicação. Os níveis de habilidades relativos à visão adotados foram visão normal, visão restrita, cegueira.
  - **Visão Normal:** É considerada visão normal a capacidade de identificar posicionamento de ícones gráficos, imagens, tamanhos e estilos de fonte, e o contraste de cores das *interfaces*. A visão é considerada normal mesmo que o usuário utilize dispositivos assistivos como as lentes de contato com grau e óculos.
  - **Visão Restrita:** É considerada visão restrita os usuários com dificuldade de identificar o posicionamento de ícones gráficos, imagens, tamanho e estilo de fonte, o contraste de cores das *interfaces*. A visão é considerada restrita mesmo que o usuário utilize dispositivos assistivos como as lentes de contato com grau e óculos, ainda assim a visão não seja adequada. Nesta condição a utilização de recursos de áudio para apoiar a identificação dos elementos da *interface* é recomendável.
  - **Cegueira:** É considerado cego a pessoa que não dispõe a capacidade de ver ícones gráficos, fontes e contrastes de cores das *interfaces*, mesmo com o auxílio de dispositivos assistivos, como lentes de contato de grau ou óculos. Nesta condição a utilização de recursos de áudio para apoiar a identificação dos elementos da *interface* é necessária.

- **Auditiva:** Está relacionado a habilidade do usuário distinguir e identificar elementos gráficos e textuais através da audição. Os níveis de habilidades adotados para a audição foram audição normal, audição restrita e surdez.
  - **Audição Normal:** É considerada audição normal a capacidade de identificar o som da voz no pronunciamento das palavras de uma mensagem, na audição de sons produzidos por objetos e animais. A audição normal é considerada mesmo se o usuário utilize dispositivos assistivos como aparelho auditivo.
  - **Audição Restrita:** É considerada audição restrita a dificuldade de identificar o som da voz no pronunciamento palavras de uma mensagem, na audição de sons produzidos por objetos e animais. A audição restrita é considerada mesmo se o usuário utilize dispositivos assistivos como aparelho auditivo, ainda assim, a audição não seja adequada. Nesta condição é recomendável identificar o posicionamento de ícones gráficos, imagens, fontes de tamanho e estilo adequado, contraste de cores das *interfaces* que auxiliem a visualização.
  - **Surdez:** É considerado surdo a pessoa que não dispõe a capacidade de ouvir o som da voz no pronunciamento das palavras de uma mensagem, na audição de sons produzidos por objetos e animais, mesmo com auxílio de dispositivos assistivos como aparelhos auditivos. Nesta condição a utilização de recursos visuais para apoiar a identificação dos elementos da *interface* é necessária.
- **Motora:** É relacionada a habilidade do usuário manusear e utilizar dispositivos de interação (tela toutchscreen, teclado, mouse etc) em ambientes digitais. Os níveis de habilidade relativos a motricidade adotados foram motricidade normal, motricidade restrita e sem motricidade, aplicadas para os membros superiores e inferiores.
  - **Motricidade Normal:** É considerada motricidade normal a capacidade de realizar movimentos finos (agilidade e precisão com dedo das mãos) ou não com os membros inferiores e superiores. Um movimento fino consideramos clicar o botão do mouse, selecionar e arrastar objetos na tela, digitação. A movimentação de braços e pernas é considerada normal, quando o usuário consegue interagir com aplicações sensoreadas captam seus movimentos.
  - **Motricidade Reduzida:** É considerada motricidade reduzida as pessoas que possuem dificuldades em realizar movimentos finos ou não com os membros superiores e inferiores. Nesta situação é recomendado um maior afastamento entre os ícones gráficos, área de seleção ampliada, tempo de espera para a resposta do usuário maior, redução do número de cliques e seleções simultâneas na interação, comandos de voz.
  - **Sem Motricidade:** É considerada sem motricidade a pessoa sem a capacidade de realizar movimentos finos ou não com membros superiores e inferiores. Os

usuários nessa condição é necessário a interação com aplicações utilizando comandos de voz.

- **Cognitiva:** Está relacionada a habilidade da pessoa em reter informações recentes tendo condições de expressá-la de forma lógica e precisa. Ao interagir com o ambiente ter a capacidade de se localizar, planejar, sequenciar o relacionamento e combinação de objetos durante a realização de atividades, sem perder o foco e dentro de um tempo adequado. As funções cognitivas estão descritas na capítulo 3.
- **Estado Emocional:** É relacionado as condições do paciente ligadas ao humor, se o paciente é uma pessoa feliz ou triste, temerosa ou confiante, entusiasmado, motivada ou desmotivada. Estes fatores influenciam na escolha dos desenvolvedores, em que narrativa utilizar, quais as recompensas oferecer, o nível de dificuldade utilizado nas atividades, a proposta dos desafios.