Simulação com Sistemas Multiagentes Inteligentes de Cenário com Paciente Alzheimer

Nicolas Liberalesso Inácio
Curso de Sistemas de Informação
Universidade Franciscana
Santa Maria, Brasil
n.inacio@ufn.edu.br

Alexandre de Oliveira Zamberlan
Curso de Sistemas de Informação
Universidade Franciscana
Santa Maria, Brasil
alexz@ufn.edu.br

***Resumo*—O trabalho apresenta o projeto de um sistema multiagente cognitivo que simula alguns comportamentos de paciente Alzheimer, situado em um cômodo específico. Esse paciente é monitorado por dois tipos de sensores (agentes), de forma que agentes atuadores do ambiente simulado possam planejar e acionar um esquema de iluminação e/ou de Smart TV. Dessa forma, com esse planejamento automatizado, pretendeu-se identificar situações de intervenção do cuidador às chamadas do paciente. Assim, via a linguagem AgentSpeak(L) e seu interpretador Jason, objetivou-se criar um Sistema Multiagente que simulasse um cenário com agentes representando doente Alzheimer, sensores, atuadores, equipamentos e cuidador.**

***Index Terms*—SMA; Jason; AgentSpeak(L).**

I. Introdução

O Alzheimer é uma doença neurodegenerativa que tem por principal sintoma a perda gradativa de habilidade de lembrar de informações recebidas recentemente [1]. Essa mesma obra apresenta que a doença tem três estágios e está na categoria de demência. É uma doença crônica que prejudica a cognição da pessoa afetada permanentemente e gradualmente. Além da perda de memória, outros sintomas comuns da Doença Alzheimer (DA) são: dificuldade em completar tarefas básicas, confusão com data, hora e localização e dificuldades nas horas de falar e escrever [1]. Os estágios do Alzheimer são classificados como leve, moderado e grave, respectivamente. Uma pessoa no estágio leve costuma apresentar perda de memória recente, desorientação no horário e localização e até mesmo agressividade em determinadas ocasiões. No segundo estágio, esses sintomas aumentam consequentemente, dificultando as atividades diárias. Também aumenta a necessidade de receber cuidados de outra pessoa. O terceiro estágio é o mais severo da doença, causando dificuldades para a pessoa afetada, ou seja, dificulta nas atividades como comer, andar, reconhecer parentes e pessoas próximas entre outros [1], [2].

A. Justificativa

Conforme o autor Pelzer citado em [3]: a DA pode ser considerada uma doença familiar, por mudar profundamente o cotidiano das famílias. Portanto, cuidar um paciente Alzheimer, acaba sobrecarregando o cuidador (principalmente familiares), devido a necessidade de proporcionar cuidados e atenção ininterruptos, muitas vezes abrindo mão da sua própria vida pessoal.

Portanto, faz-se necessário criar mecanismos ou recursos que diminuam a intervenção ou às chamadas de cuidadores (principalmente familiares) no trato com pacientes Alzheimer, de forma que minimize o desgaste emocional durante o processo de cuidar.

Uma alternativa para ajudar esses cuidadores é criar recursos em que o cuidador não seja acionado a todo momento pelo DA. Ou seja, projetar sistemas de sensoreamento e acionamento, que percebam movimentos ou sons do paciente e que atuem no ambiente, ligando e configurando alguns equipamentos, como esquema de iluminação ou de *Smart* TV. Dessa forma, há uma intervenção não humana, tentando acomodar da melhor forma o DA.

Na área da Inteligência Artificial (IA), há o paradigma Sistemas Multiagentes (SMA), que é uma abordagem com vários agentes que interagem entre si de acordo com a programação. Assim, proporcionando formas de simular a automatização de serviços que antes necessitariam de uma ou mais pessoas para executar ou auxiliar o ser humano na execução dos mesmos. Com SMA é possível projetar e implementar sistemas com comportamento inteligente e distribuído, de forma simulada, principalmente em componentes de automação residencial, como sensores e/ou atuadores.

Neste contexto, criar uma cenário de simulação comportamental de paciente Alzheimer de 2º estágio (moderado), pode ser realizado (atingido) via o paradigma de SMA, uma vez que há linguagem e interpretador consolidados e testados, como AgentSpeak(L) e o Jason, respectivamente. Ademais, há o trabalho de Aloisio Kneipp dos Santos [4], que realizou uma simulação e a integração, com tais tecnologias, para automação residencial com a presença de sensores (agentes simulados) e atuadores (acionadores reais). Esse trabalho, ou o cenário construído com a integração de serviços é a base desta simulação proposta.

B. Objetivo

Projetar e desenvolver um sistema multiagente que simule um cenário com agentes representando doente Alzheimer, sensores, atuadores, equipamentos e cuidador.

Para que o objetivo geral seja alcançado, identificaram-se alguns objetivos específicos:

- entender as fases do Alzheimer e identificar os principais comportamentos do paciente;
- mapear e compilar trabalhos relacionados;
- identificar e testar componentes de hardware (atuadores de áudio e vídeo, e sensores de movimento, por exemplo) que interajam com o interpretador Jason;
- modelar e implementar o sistema multiagente tratando cada tipo de sensor e atuador como um agente do sistema, ou seja, que percepções e ações cada agente terá;
- integrar o ambiente Jason aos atuadores do ambiente;
- simular um cenário de teste para verificar se houve comportamento inteligente (autônomo, proativo, flexível e comunicativo) dos agentes na camada de software (Jason) com os agentes na camada de hardware (sensores e atuadores).

II. Revisão bibliográfica

Esta seção trata dos fundamentos e processos trabalhados no texto, incluindo definições, tecnologias, exemplos e cenários dos tópicos que são usados no trabalho.

A. Doença Alzheimer e a Demência

O Alzheimer é uma doença cerebral degenerativa e sem cura. Ela se manifesta de forma lenta, porém é irreversível. Como consequência, a pessoa afetada passa a ter problemas na integridade física e mental, além de dificuldades para lembrar de acontecimentos, raciocinar, comunicar-se e realizar tarefas básicas, por exemplo [5]. O que torna indispensável o auxílio de uma pessoa para cuidar do afetado conforme o avanço da DA. Como citado por Maia e colaboradores [5], em 2018 já afetava mundialmente cerca de 46,8 milhões de pessoas, tendo previsões de atingir 74,7 milhões, em 2030, e 131,5 milhões, em 2050. O trabalho também cita que no ano de 2015 foram gastos aproximadamente 818 bilhões de dólares com pessoas afetadas pela doença, isso equivale a mais de 1% do PIB mundial. Desses custos, a maior parte é voltada aos cuidados e apoio dedicados a pessoas afetadas pela DA.

No Brasil, o segmento idoso na população vem aumentando constantemente conforme o crescimento da qualidade de vida. Em 2009, os brasileiros com mais de 60 anos já representavam cerca de 10,2% da população do país, com tendência de chegar a 14% até 2025 [3]. Carmo e colaboradores [2] afirmam que o Censo Demográfico do Instituto Brasileiro de Geografia e Estatística (IBGE) de 2010 aponta que a população de idosos constitui seis milhões de brasileiros na faixa etária de 60 a 64 anos e três milhões com 80 anos ou mais. Ele também escreve que 40% da população idosa com mais de 90 anos tende a desenvolver o Alzheimer.

Tendo esse constante aumento da população idosa no Brasil, é normal que o número de pessoas afetadas pela DA também aumente, visto que dentre as demências, a DA é considerada a principal e mais comum delas.

Finalmente, ao raciocinar linearmente, com 3 milhões de idosos acima de 80 anos, sendo que 40% deles com risco de Alzheimer, tem-se o número 1,2 milhões de prováveis doentes Alzheimer. Para cada doente Alzheimer há a necessidade de cuidador. Se dois DA forem cuidados pelo mesmo cuidador, são necessários 600 mil cuidadores, que de alguma forma vão precisar de algum apoio ou recurso emocional.

B. Internet das Coisas

A Internet das Coisas (do inglês *Internet of Things* (IoT)) emergiu dos avanços de várias áreas como sistemas embarcados, microeletrônica, comunicação e sensoriamento. IOT é uma extensão da Internet aplicada em objetos do dia-a-dia, que tenham capacidade computacional e acesso a Internet, assim se comunicando e realizando tarefas de acordo com a necessidade do ser humano [6].

Dentro da IOT, há subsistemas: ubíquos e pervasivos. Sistemas ubíquos atuam de forma imperceptível em dispositivos que podem ser incorporados em roupas, móveis ou outros objetos do dia a dia, tornando-se parte integrante do ambiente. São projetados para serem adaptáveis e interativos, respondendo às necessidades dos usuários em tempo real e oferecendo uma experiência personalizada e transparente. Sistemas pervasivos são caracterizados por dispositivos inteligentes onipresentes em um ambiente específico. Esses dispositivos são geralmente interconectados, podendo coletar e compartilhar informações com intuito de melhorar o funcionamento do sistema. Esses sistemas são projetados para trabalhar sem intervenção humana, e geralmente são orientados a tarefas específicas. A principal diferença deles é que Sistemas Ubíquos tem como foco fornecer uma experiência mais natural e imperceptível ao usuário e Sistemas Pervasivos priorizam uma tarefa específica e geralmente são mais visíveis no ambiente que atuam [7].

C. Tecnologias para automação

Há inúmeras tecnologias que auxiliam na automação em geral, seja residencial, automotiva, industrial, na saúde, por exemplo. Mas é possível destacar relés, sensores e acionadores.

1) Relés: Como escrito por Mattede [8], os relés são dispositivos elétricos que produzem modificações súbitas e pré-determinadas em um ou múltiplos circuitos elétricos de saída. Ainda em seu artigo, o autor define que o relé tem um circuito de comando, que, no momento que é alimentado por uma corrente, aciona um eletroímã que faz a mudança de posição de outro par de contatores, que estão ligados a um circuito ou comando secundário. Em resumo, o relé é um contato que abre e fecha de acordo com determinados fatores.

2) Sensores: existem diversos tipos de sensores, podendo ser de movimento, distância, temperatura, presença, entre outros. O trabalho de Cardoso [9] mostra que o funcionamento dessa ferramenta consiste em receber um tipo de energia e responder convertendo-a em outro tipo de energia. Como descrito pelo mesmo autor [9], sensores de movimento funcionam com um foto sensor que usando uma luz infravermelho joga elétrons em um espaço e os monitora. Para que eles detectem uma pessoa, é necessário que sejam sensíveis a temperatura do corpo humano, sendo ela aproximadamente de 34°C. Cardoso [9] também descreve que existem alguns tipos de sensores de movimento e um dos mais comuns são os que funcionam

enviando um feixe de luz cruzando o espaço perto da porta e um foto sensor do outro lado desse espaço. Quando alguma coisa cruza o feixe, o foto sensor detecta a mudança e envia uma resposta de acordo com sua configuração. Outro modelo funciona com um sistema de abertura de portas automática, que fica enviando um sinal e capta o seu retorno. No momento que algo atravessa este feixe e altera o sinal da volta, o sistema entende esta alteração e envia uma resposta. Esses sistemas são considerados ativos, pois eles injetam energia (luz, micro-ondas ou som) no ambiente para detectar qualquer espécie de alteração.

3) Acionadores: são um recurso de tecnologia assistiva que funcionam disparando uma função acionadora, eles podem ser usados tanto em conjunto como separadamente para se adequar melhor a necessidade de quem vai usar. Uma função pode variar de acordo com a configuração do dispositivo. Essa ferramenta pode ser usada em diversos equipamentos, como por exemplo, luminárias, mouses, ventiladores, tomadas, entre outros. Na área da tecnologia, eles são muito usados por pessoas que apresentam limitações físico-motoras, pois esse recurso dá a essas pessoas a autonomia necessária para realizar as tarefas nas quais têm dificuldade, como ligar e desligar luzes, acionar uma campainha, jogar videogame, entre outras [10]. Neste trabalho, o foco dos acionadores está para disparar eventos aos agentes do SMA, de acordo com o comportamento do paciente.

A relação desses três equipamentos acontece em sistemas de controle e automação. No sistema proposto, os sensores devem monitorar alguns comportamentos do DA, como por exemplo, movimentação e sons gerados pelo paciente. Uma vez que esse equipamento sensora ou percebe algo, as informações são processadas e podem ser usadas em um relé. O relé, por sua vez, atua como um interruptor ou um 'mini-controlador' elétrico que deve tratar o sinal vindo do sensor e/ou encaminhar a outro circuito elétrico, como um acionador. Por fim, um sensor detecta informações sobre o ambiente, o relé controla o fluxo e um acionador executa uma ação desejada em um determinado equipamento. Neste projeto, os equipamentos são *Smart* TV, ar condicionado e o celular do cuidador. O relé de acionamento é o Sonoff, enquanto que os sensores são simulados.

De acordo com a revisão realizada por Santos [4], Sonoff possui internamente um microcontrolador com *socket Wi-Fi* embutido, que tem a função de se conectar à rede. Algumas versões possuem fonte bivolt para ser ligada na rede elétrica. Assim, Sonoff torna-se um interruptor inteligente sem fio, versátil e de baixo custo. Registra-se, também, que Sonoff usa o protocolo *Message Queue Telemetry Transport* de comunicação máquina-máquina, que interconecta sistemas. Novamente de acordo com Santos [4], clientes do protocolo MQTT podem ser *publisher* e/ou *subscribers*, sendo que quando atua de maneira *publisher* o cliente envia informações para o *broker*, e quando atua como *subscriber* se associa para receber as informações. O *broker* recebe e armazena as informações do *publisher* e envia para os *subscribers*. Por fim, o protocolo MQTT possui três níveis de qualidade de serviço *QoS*: nível 0 (mensagem será enviada apenas uma vez e que a mensagem pode não ser recebida); nível 1 (protocolo garante que a mensagem será entregue pelo menos uma vez, podendo ser retransmitida mais de uma vez); nível 2 (enviada a mensagem e há verificação se a mensagem está sendo recebida apenas uma vez).

D. Sistemas Multiagentes, agentes e automação

Para Hübner [11], os SMA são desenvolvidos com foco na coletividade de diferentes indivíduos, que devem trabalhar em conjunto e de forma organizada para solucionar determinados problemas, os quais apenas um agente não seria capaz de resolver sozinho. Essa área da IA apresenta duas propriedades que alguns consideram contraditórias. A primeira delas é a autonomia dos agentes, ela destaca a existência de um agente independente dos demais. A outra propriedade é a organização, que por sua vez visa estabelecer restrições no comportamento desses agentes. Portanto, é importante compreender como elas interagem no contexto dos SMA, pois muitas das propriedades desejadas nessa área da IA advém do equilíbrio das propriedades citadas.

Um agente de SMA é um sistema computacional encapsulado que está situado em algum ambiente e é capaz de realizar ao menos um ação reflexiva e autônoma conforme sua programação. Ele é capaz de interagir com outros agentes e normalmente trabalha de forma assíncrona. Suas principais características são autonomia, proatividade e capacidade de interagir com outros agentes do mesmo ambiente [12].

De acordo com Santos e Zamberlan [4], é possível assumir que a teoria SMA esteja em diferentes aplicações, como simulação computacional, jogos eletrônicos e sistemas autônomos robóticos, por exemplo. Essas aplicações têm em comum componentes de software ou de hardware que precisam atuar e perceber o ambiente, sem que sejam controlados (conceito de autonomia). Além disso, devem executar ações ou tarefas de forma proativa, adaptáveis e flexíveis às situações inesperadas do ambiente. Também, precisam estar em constante comunicação entre si, coordenando tarefas ou ações junto ao ambiente.

E. Jason e AgentSpeak(L)

A linguagem AgentSpeak(L) é um recurso que possibilita a programação lógico-declarativa de agentes cognitivos do tipo BDI (*Belief, Desire, Intention*), via predicados ou proposições, no formato de crenças e planos. A linguagem dá condições para que o comportamento dos agentes seja modelado e implementado por meio da lógica de primeira ordem [13], em que todo o ciclo de raciocínio do agente (ou agentes) se dá por perceber (sensorar), planejar e executar (atuar).

De acordo com seus autores [11], Jason é um interpretador *Open Source* da linguagem AgentSpeak(L) implementado em Java, que inclui a comunicação entre agentes via diretivas nativas do interpretador, características de autonomia por meio de Threads embutidas e transparentes aos programadores, e planos em forma de proposições declarativas de lógica de primeira ordem[1]. Outras características do Jason são [11]:

[1]Muito similar aos fatos e regras da Linguagem Prolog.

- negação forte *strong negation*, possibilitando uso tanto de sistemas mundo-aberto *open-world*, quanto de sistemas mundo-fechado *closed-world*;
- tratamento de falhas em planos;
- anotações em identificadores nos planos, para melhorar a organização e dar mais opções de personalização na área dos planos;
- modelar e implementar o sistema multiagente tratando cada tipo de sensor e atuador como um agente do sistema, ou seja, que percepções e ações cada agente terá;
- suporte para desenvolvimento de ambientes, os quais normalmente são desenvolvidos em Java;
- bibliotecas básicas de ações internas e suas extensões.

Na instituição, há vários trabalhos que utilizaram o interpretador Jason, como por exemplo, o trabalho de Santos [4].

F. Metodologia Prometheus

Abrangendo desde a modelagem até a implementação em sistemas SMA, essa metodologia tem três diferentes fases, sendo elas, a especificação do sistema, o projeto arquitetural e o projeto detalhado. Os componentes dessas fases são utilizados tanto na geração do código, quanto na realização dos testes [14].

A especificação do sistema é a primeira fase, que é composta por duas atividades: definição de cenário e determinação de objetivos do sistema. O cenário é definido com informações do ambiente, ações e dados externos. Já o objetivo se responsabiliza por definir as funcionalidades necessárias para alcançar os resultados desejados nos cenários determinados [14].

Na seguinte fase o projeto arquitetural determina os agentes que existirão e suas interações de acordo com os resultados obtidos na primeira fase. Essa etapa divide-se em três etapas, sendo elas definição dos tipos de agentes, definição de estrutura do sistema e definição de interações entre agentes [14].

Na última etapa, o projeto detalhado responsabiliza-se por definir as capacidades de cada agente, junto com seus eventos internos, planos e uma estrutura detalhada de cada tipo de agente identificado na segunda etapa [14].

G. Trabalhos Relacionados

O trabalho de França e colaboradores [1] mostra que devido ao aumento da qualidade de vida, a quantidade de idosos tem crescido cada vez mais. Essa autora afirma que estudos apresentam que essa faixa etária, frequentemente, desenvolve doenças incapacitantes, dificultando o cotidiano. O Alz-HomeCare é um software desenvolvido pela Flávia França e colegas [1], com a proposta de auxiliar o cuidador de um paciente Alzheimer, a partir de notificações dos estados e das ações do paciente.

O Alz-HomeCare funciona utilizando diferentes tecnologias no monitoramento do paciente, sendo elas GPS, Wifi, celular, acelerômetro, *smartwatch*, etiqueta (*tag*), *Near Field Communication* (NFC) e *Bluetooth Low Energy* (BLE). Cada tecnologia aplicada tem sua determinada função nos cuidados do paciente.

Na Figura 1, é possível visualizar a arquitetura proposta por França e colaboradores [1]. Na arquitetura há 3 níveis: um para notificações, um para tratar dados e outro para os processos de monitoramento.

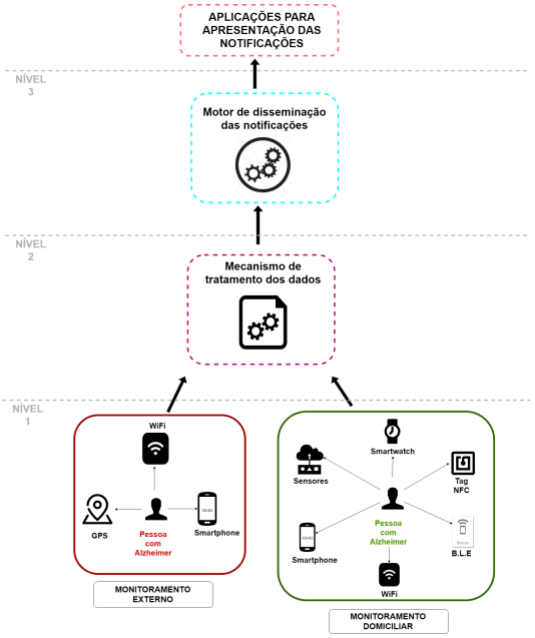

Figura 1. Arquitetura do Alz-HomeCare [1].

O Alz-HomeCare coleta dados referente ao estado do paciente, sendo eles localização, pressão, detecção de quedas, entre outros. Ao detectar algo fora do normal, o sistema envia uma notificação a um responsável. Essa pessoa pode ser um familiar cuidador ou o próprio médico, que deve fazer algo refente a situação.

O segundo trabalho [2] teve como objetivo identificar Tecnologias Assistivas (TA) para idosos com DA. O estudo foi focado em idosos que estavam nos estados iniciais da doença, mas já precisavam de cuidadores.

Dentre as tecnologias analisadas e citadas no estudo, o videogame se destacou como estimulante cognitivo. Outra tecnologia que se destacou foram as "Smart Homes", também conhecidas como casas inteligentes. Mais um caso com destaque foi o uso de robôs para estimulo cognitivo e social, porém tal tecnologia só deve ser utilizada com a supervisão de um cuidador.

Os estudos conduzidos por Carmo e colegas [2] também apontam que os idosos brasileiros tendem a esquecer o autocuidado, principalmente os homens, dos quais apenas 7,7% frequentam uma unidade de saúde e 24% costumam manter um intervalo superior a 180 dias em uma consulta e outra. Isso torna mais útil uma TA para auxiliar nos cuidados dos idosos com Alzheimer.

No trabalho de Vasconcelos e colegas [3], a partir de um estudo baseado em entrevistas semi-estruturadas, 85,71% dos cuidadores de familiares idosos são do sexo feminino. A

faixa etária desses cuidadores está em torno dos 55 anos, com extremos em 35 e 77 anos. Nesse estudo, houve uma preocupação com o desgaste emocional e foi notado que ele é quase nulo no primeiro estágio da DA, com o paciente apresentando poucas perdas de memórias e alterações de humor. Dessa forma, a tarefa do cuidador acaba sendo bem baixa. Conforme a demência vai se agravando e avançando para o estágio dois, a função do cuidador tende a aumentar proporcionalmente. Cabe ressaltar que nesse estágio ainda não se faz necessário a supervisão diuturnamente.

O maior pico do desgaste emocional é no terceiro estágio. Quando o afetado pelo Alzheimer perde totalmente a autonomia e passa a depender plenamente do cuidador. É nesses casos, que o familiar cuidador acaba abrindo mão de sua própria vida para cuidar do afetado, caso não tenha ajuda de terceiros. Como resultado, esse cuidador familiar acaba tendo consequências em suas interações familiares e sociais. Ele tem alterada a sua saúde física, apresenta perda de autoestima e da líbido [3].

Finalmente, na pesquisa realizada por Santos [4], foram analisados dispositivos Raspberry Pi e a tecnologia Jason para integrar Sistemas Multiagentes, com intuito de possibilitar que computadores e/ou dispositivos dotados de processamento e conexão à Internet possam oferecer serviços aos usuários, não importando o local e nem a hora.

Na implementação de seu projeto, Santos [4] também usou SMA com a metodologia Prometheus. Na prática o sistema utiliza um Raspberry Pi, o qual monitora e controla o ambiente de forma descentralizada com agentes autônomos, tendo flexibilidade e proatividade conforme programados. Além disso, conta com a ideia de um sistema Web o qual é responsável por gerenciar os usuários e interligá-los com seus respectivos perfis em cada ambiente.

1) Considerações sobre trabalhos relacionados: O primeiro trabalho [1] tem um destaque superior aos demais, pois seu foco está no auxílio do cuidador (monitoramento e notificação), que é justamente o objetivo deste trabalho. Diferente do segundo trabalho [2], que mostra formas de auxiliar o DA, mas que não se descarta sua contribuição, uma vez que ajudou na construção da ideia do projeto. O terceiro trabalho mostra o impacto negativo quando um cuidador familiar de um DA assume a responsabilidade sozinho. Como no segundo trabalho, o terceiro [3] apresenta análise e discussão dos impactos da doença, que colaboram com a justificativa desta proposta de pesquisa. O simulador projetado e implementado por Santos [4], mostra o uso e a integração de diferentes tecnologias.

Por fim, os trabalhos de Santos [4] e França [1] devem ser referências deste estudo. Uma vez que França [1] tem prioridade no monitoramento e nas notificações, auxiliando cuidadores. No entanto, no texto desse trabalho, não há referências de implementações, somente de tecnologias utilizadas. Por outro lado, Santos [4] usa um sistema multiagentes simulado, fazendo o uso das tecnologias Jason e Raspberry Pi, junto da metodologia Prometheus para definir os atuadores e suas funcionalidades.

III. Metodologia e proposta de trabalho

Este trabalho foi baseado em pesquisa exploratória com desenvolvimento de simulação computacional como recurso de entendimento e avaliação. Já no projeto e desenvolvimento do sistema, foram utilizados a metodologia Scrum [15] com a técnica Kanban para gestão de atividades, prazos e responsáveis.

Os encontros entre aluno e orientador ao longo do semestre foram semanais, seguindo orientações da metodologia, e sempre houve apresentação, e posteriormente discussão, de atividades desenvolvidas. As atividades *Product Backlog* foram geridas pela ferramenta *Web* Trello (técnica Kanban) para controle de prazos.

Vale ressaltar que a equipe de desenvolvimento foi formada unicamente pelo aluno e professor orientador. O aluno, além de desenvolvedor do sistema, comportou-se como *Product Owner* e, neste caso, o professor orientador adquiriu o papel de *Scrum Master*.

As ferramentas utilizadas foram:

- software de Kanban Trello;
- ambiente de diagramação Astah;
- linguagem AgentSpeak(L) e interpretador Jason;
- linguagem Java;
- metodologia Prometheus;
- relé de acionamento Sonoff.

A. Ideia do sistema multiagente

Nesta seção, buscou-se apresentar especificações de alguns aspectos funcionais e estruturais. A Figura 2 ilustra os agentes do sistema e as principais funcionalidades que se buscou modelar. Vale lembrar que esta simulação tem como base o trabalho de Santos [4], que realizou uma simulação e a integração, com tais tecnologias, para automação residencial com a presença de sensores (agentes simulados) e atuadores (acionadores reais).

A Figura 2 tem dois agentes que representam seres humanos: AgentePaciente e AgenteCuidador. O AgentePaciente tem como função gerar aleatoriamente 2 a 3 sintomas mapeados (gerar algum tipo de som ou de movimentação). Já o AgenteCuidador, em algumas situações vai receber notificações de agentes atuadores. O cenário de simulação também conta com os agentes sensores (Som e Movimento), que detectam as ações geradas pelo AgentePaciente. Já os agentes atuadores (Iluminação e *Smart* TV) tem planos para tratar eventos, que são enviados pelos agentes sensores, e planos para acionar os relés (Iluminação ou *Smart* TV). Nesse ponto, os relés são equipamentos, que atuam em um ambiente real, porém, são representados pelos agentes atuadores dentro da simulação.

Decidiu-se que os comportamentos do DA para serem usados na simulação são: gritar, pedir ajuda, levantar-se da cama, agitar-se. Esses comportamentos devem ser gerados aleatoriamente, simulados em períodos do dia (manhã, tarde, noite, madrugada). Cabe ressaltar, que um comportamento, como o gritar no período da madrugada, pode ser sensoreado pelos dois agentes: Som e Movimento.

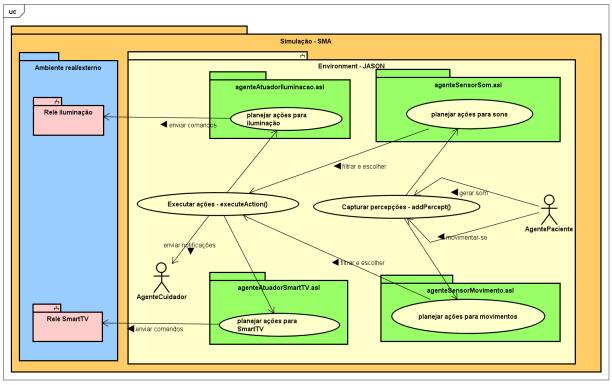

Figura 2. Ideia da arquitetura da simulação.

A Figura 3 apresenta a dinâmica da simulação, envolvendo o ambiente do simulador (Jason) e os relés de atuação (que estão no ambiente real). No diagrama, tem o início a definição dos cenários, seguido por carregar crenças. Após, inicia-se a simulação. É possível notar que os acionadores no ambiente (relés) são ativados pelos agentes da simulação, em especial os agentes que representam os atuadores.

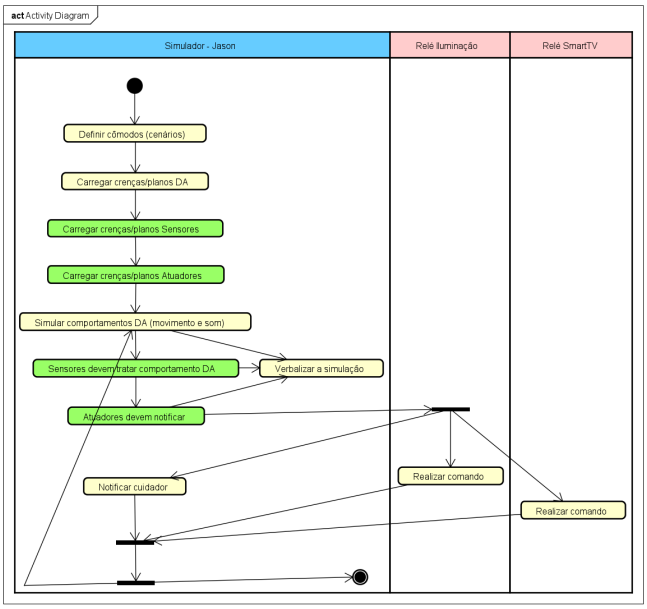

Figura 3. Dinâmica de funcionamento da simulação.

Por fim, tendo como base a metodologia Prometheus, há o diagrama Visão Geral de Análise da fase Especificação do Sistema, apresentado na Figura 4. O agente Paciente gera percepções (aleatórias a partir de uma lista de comportamentos DA) que são notadas ou sensoreadas pelos agentes Som e Movimento. Por exemplo, quando o agente paciente pedir ajuda, o agente Som deve tratar esse evento (percepção), via o plano Tratando Ajuda. Nesse plano há duas diretivas de comunicação, uma para o agente Cuidador e outra para o agente Atuador Iluminação, que por sua vez, aciona o Relé de Iluminação por outra diretiva de comunicação. Esse diagrama apresenta a modelagem do SMA para o ciclo gerar, perceber, planejar e atuar.

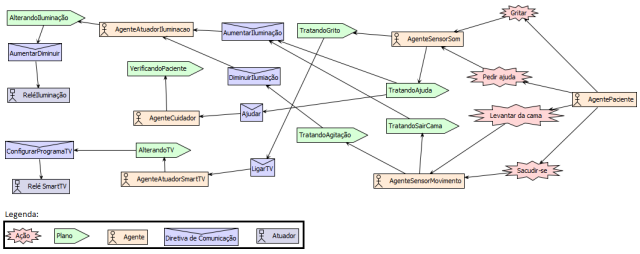

Figura 4. Visão Geral de Análise.

B. Resultados e discussões

A Figura 5 mostra os cenários pré definidos para um ambiente de simulação. O quadro prioriza mostrar as ações do DA, o turno do dia e como os sensores e atuadores se comportariam perante elas. Analisando o quadro, é possível notar que ele não leva em consideração o clima e o evento, dando mais prioridade ao comportamento de acordo com o turno. Foi decido que apenas na percepção de grito o sistema iria acionar o responsável cuidador. Nos casos do paciente levantar da cama em períodos de pouca iluminação, o sistema iria ligar a luz. Esses comportamentos têm o propósito de simular cenários em que normalmente o cuidador seria acionado. A intenção, mesmo que simulada, é distrair o paciente, sem a necessidade de intervenção do cuidador. Acionando o agente cuidador apenas em casos de reincidência do comportamento simulado do paciente.

Percepção paciente (sensoreamento)			Comportamento paciente	Sensor	Atuadores Sonoff		
Turno	Clima	Evento			Iluminação	Smart TV	Cuidador
...	...	...	Gritar	som		Ligar	Chamar
Madrugada	...	...	Sacudir	movimento		Ligar	
Noite	...	...	Levantar	movimento	Ligar	Ligar	
Madrugada	...	...	Levantar	movimento	Ligar	Ligar	
Manhã	...	...	Levantar	movimento		Ligar	
Tarde	...	...	Levantar	movimento		Ligar	
Obs.: Comportamento reincidente do paciente para uma percepção, Cuidador será acionado via o Chamar.							

Figura 5. Quadro de percepções do paciente e do ambiente, com ações dos atuadores.

Já a Figura 6, mostra como a simulação Jason gera situações em que paciente percebe o ambiente, realiza ações (gritar, chamar, levantar, agitar) e os agentes sensores percebem esses eventos, tratando-os via planos (com ativação dos agentes atuadores).

C. Cenários de simulação

O cenário 1 (Figura 7) mostra clima de tempestade, nenhum evento e turno noite, em que o paciente não é acionado. Ou seja, não há planos para tal combinação de percepções, mantendo o paciente em estado de calmo.

O cenário 2 (Figura 8) mostra clima de tempestade, barulho como evento, turno tarde, em que o paciente é acionado, gerando um chamado de um conhecido (grito no ambiente). O

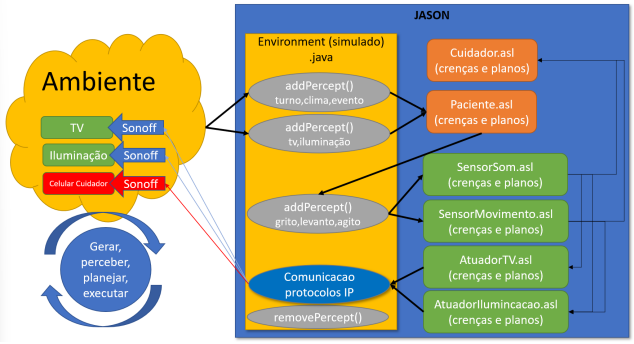

Figura 6. Arquitetura Jason da simulação.

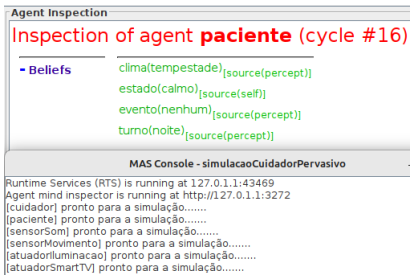

Figura 7. Cenário 1.

sensor de som detectou o grito e acionou o atuador SmartTV, que enviou comunicação ao Sonoff, que ligou a TV e fazendo que o paciente entrasse em estado de calmo.

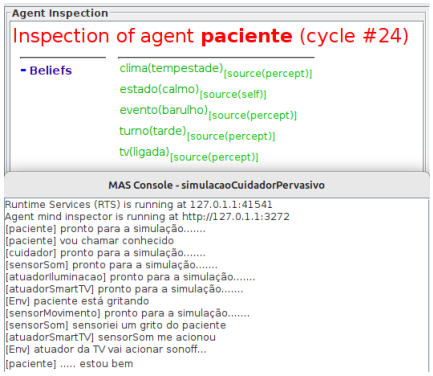

Figura 8. Cenário 2.

O cenário 3 (Figura 9) mostra clima normal, barulho como evento, turno madrugada, em que o paciente inicialmente estava calmo, porém foi acionado com o movimento levantar. O sensor de movimento detectou o paciente levantando e acionou os atuadores de iluminação e SmartTV, que enviaram comunicação ao Sonoff, que ligou a luz e a TV, como mostrado na tabela da (Figura 5).

O cenário 4 (Figura 10) mostra clima normal, visita como evento, turno noite, em que o cuidador recebeu uma visita. Com isso, o paciente foi acionado com o movimento levantar. O sensor de movimento detectou o paciente levantando acionou os atuadores de iluminação e SmartTV, que enviaram

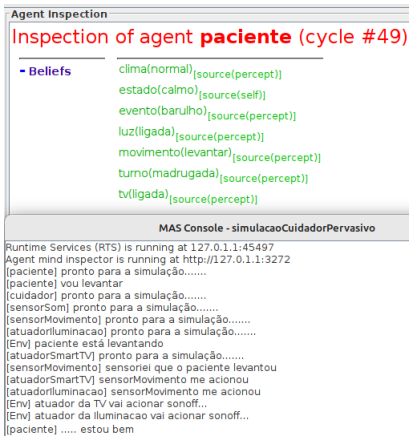

Figura 9. Cenário 3.

comunicação ao Sonoff, que ligou a luz e a TV para acalmar o paciente, como mostrado na tabela da (Figura 5).

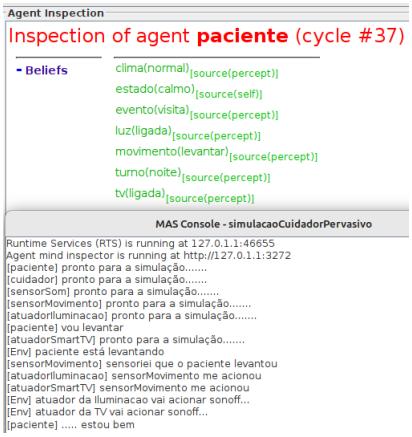

Figura 10. Cenário 4.

Por fim, o cenário 5 (Figura 11) mostra clima tempestade, barulho como evento, em que o paciente é acionado, gerando um chamado de um conhecido (grito no ambiente). O sensor de som detectou o grito e acionou o atuador SmartTV, que enviou comunicação ao Sonoff, que ligou a TV e fazendo que o paciente entrasse em estado de calmo.

Para a execução da simulação, foram implementados: i) ambiente (Figuras 12 e 13); ii) classe Comunicacao (Figura 14); iii) agente paciente (Figura 15); iv) agentes sensores (Figuras 16 e 17), v) agentes atuadores (Figuras 18 e 19).

O ambiente é representado por um código Java que gera aleatoriamente turno, eventos externos e situações de clima. Nesse ambiente, há o método *init* que dá a partida da simulação. Também no código Java, há o método *executeAction*, responsável por tratar todas as ações pretendidas pelos agentes, ou seja, uma fez que um agente planeja algo, é no método que as ações são implementadas ou direcionadas a sistemas/programas externos, como Sonoff.

A classe comunicação também é um código Java com recursos para realizar a integração entre a simulação Jason

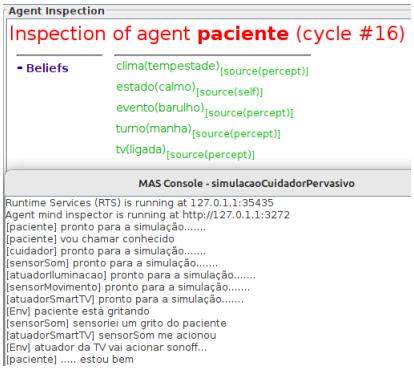

Figura 11. Cenário 5.

```java
package simulacao;
import jason.asSyntax.*; import jason.environment.*; import jason.asSyntax.parser.*;
import java.util.Random; import java.util.logging.*;

public class Env extends Environment {
    private Logger logger = Logger.getLogger("simulacaoCuidadorPervasivo."+ Env.class.getName());
    public String endereco_tv = "http://10.0.0.101:8081/zeroconf/switch";
    public String endereco_iluminacao = "http://10.0.0.101:8081/zeroconf/switch";

    public String gerarTurno() {…

    public String gerarEventoExterno() {…

    public String gerarClima() {…

    /** Called before the MAS execution with the args informed in .mas2j */
    @Override
    public void init(String[] args) {
        super.init(args);
        try {
            addPercept(ASSyntax.parseLiteral(gerarTurno()));
            addPercept(ASSyntax.parseLiteral(gerarEventoExterno()));
            addPercept(ASSyntax.parseLiteral(gerarClima()));
        } catch (ParseException e) {
            e.printStackTrace();
        }
    }
```

Figura 12. Código Java com primeira parte do Environment.

```java
@Override
public boolean executeAction(String agName, Structure action) {
    if (agName.equals(anObject:"paciente") && action.getFunctor().equals("som") && action.getTerm(0).toString().equals("gritar")){
        logger.info(msg:"paciente está gritando");
        try {
            addPercept(ASSyntax.parseLiteral("som(gritar)"));
        } catch (ParseException e) {
            e.printStackTrace();
        }
    } else if (agName.equals(anObject:"paciente") && action.getFunctor().equals("movimento") && action.getTerm(0).toString().equals("levantar")){
        logger.info(msg:"paciente está levantando");
        try {
            addPercept(ASSyntax.parseLiteral("movimento(levantar)"));
        } catch (ParseException e) {
            e.printStackTrace();
        }
    } else if (agName.equals(anObject:"atuadorSmartTV") && action.getFunctor().equals("ligarTV")){
        logger.info(msg:"atuador da TV vai acionar sonoff...");
        //chamada do sonoff para ligar TV
        Comunicacao.Sonoff(endereco_tv, "on", 12, null);
        try {
            addPercept(ASSyntax.parseLiteral("tv(ligada)"));
        } catch (ParseException e) {
            e.printStackTrace();
        }
    } else if (agName.equals(anObject:"atuadorIluminacao") && action.getFunctor().equals("ligarIluminacao")){
        logger.info(msg:"atuador da Iluminacao vai acionar sonoff...");
        //chamada do sonoff para ligar Luz
        Comunicacao.Sonoff(endereco_iluminacao, "on", 0, null);
        try {
            addPercept(ASSyntax.parseLiteral("luz(ligada)"));
        } catch (ParseException e) {
            e.printStackTrace();
        }
    } else logger.info("tentando executar : "+action+", mas ainda não implementado!");
    try {
        Thread.sleep(millis:5000);
        removePercept(ASSyntax.parseLiteral("som(gritar)"));
    } catch (InterruptedException e)  {
        e.printStackTrace();
    }  catch (ParseException e) {
        e.printStackTrace();
    }
    return true;
}
```

Figura 13. Código Java com segunda parte do Environment.

e componentes externos, como Sonoff.

Já a implementação dos comportamentos do paciente Alzheimer são realizados na linguagem AgentSpeak(L), em que é possível gerar deliberação ou raciocínio para tratar eventos percebidos no ambiente Java (simulado).

```java
package simulacao;
import java.io.OutputStream; import java.net.HttpURLConnection; import java.net.URL;

public class Comunicacao {
    public static void Sonoff(String query_url, String operador, int volume, String iluminacao) {
        String data = "{ \"data\" : { \"switch\" :  \"" + operador + "\" } }";
        try {
            URL url = new URL(query_url);
            HttpURLConnection conexao = (HttpURLConnection) url.openConnection();
            conexao.setConnectTimeout(timeout:100);
            conexao.setRequestProperty(key:"Content-Type", value:"application/json; charset=UTF-8");
            conexao.setDoOutput(dooutput:true);
            conexao.setDoInput(doinput:true);
            conexao.setRequestMethod(method:"POST");
            conexao.connect();
            OutputStream os = conexao.getOutputStream();
            os.write(data.getBytes(charsetName:"UTF-8"));
            os.close();
            conexao.getResponseMessage();
            conexao.disconnect();
        } catch (Exception e) {
            System.out.println(x:"Erro de comunicacao.");
        }
    }
}
```

Figura 14. Código Java para a classe de Comunicação entre Jason e Sonoff.

paciente.asl

simulacaoCuidadorPervasivo > src > agt > paciente.asl

```
//crenças iniciais do paciente
estado(calmo).

!start.
+!start : true  <-
    .print("pronto para a simulação.....").

+evento(barulho): turno(madrugada) & clima(ventania)
    <-
        .print("vou gritar de medo");
        -estado(calmo);
        +estado(agitado);
        som(gritar).

+evento(barulho): turno(madrugada)
    <-
        .print("vou levantar");
        movimento(levantar).

+evento(visita): turno(noite)
    <-
        .print("vou levantar");
        movimento(levantar).

+evento(barulho): true
    <-
        .print("vou chamar conhecido");
        som(gritar).

+tv(ligada): estado(calmo)
    <-
    .print("..... estou bem").

+tv(ligada): estado(agitado)
    <-
    som(gritar);
    .send(cuidador,tell,paciente(grito)).

+evento(barulho): turno(noite)
    <-
        .print("vou sacudir");
        movimento(sacudir).
```

Figura 15. Código AgentSpeak(L) para a simulação do comportamento do paciente.

sensorMovimento.asl M

simulacaoCuidadorPervasivo > src > agt > sensorMovimento.asl

```
!start.
+!start : true  <-
    .print("pronto para a simulação.....").

+movimento(levantar): (turno(noite) | turno(madrugada))
    <- .print("sensoriei que o paciente levantou");
       .send(atuadorSmartTV,tell,levanta);
       .send(atuadorIluminacao,tell,levanta);
       -tv(desligada).

+movimento(levantar) : true
    <- .print("sensoriei que o paciente levantou");
       .send(atuadorSmartTV,tell,levanta);
       -tv(desligada).

+movimento(sacudir) : true
    <- .print("sensoriei que o paciente sacudindo");
       .send(atuadorSmartTV,tell,sacode);
       -tv(desligada).
```

Figura 16. Código AgentSpeak(L) para a simulação do comportamento do sensor de movimento.

Assim como no agente paciente, os agentes que representam

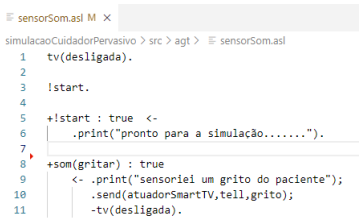

Figura 17. Código AgentSpeak(L) para a simulação do comportamento do sensor de som.

sensores de movimento e som, têm planos de comportamento implementados em AgentSpeak(L). Entretanto, toda a percepção ou sensoreamento desses agentes tem como foco o comportamento do paciente, ou seja, não percebem o ambiente, mas o paciente.

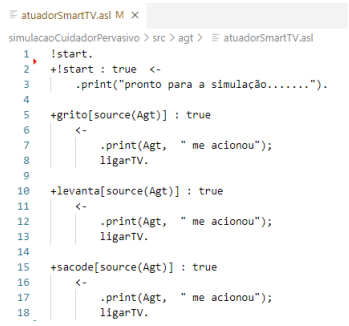

Figura 18. Código AgentSpeak(L) para a simulação do comportamento do atuador da SmartTV.

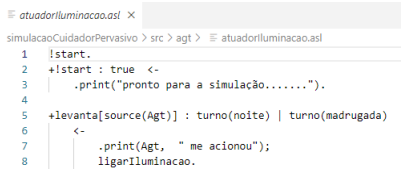

Figura 19. Código AgentSpeak(L) para a simulação do comportamento do atuador da iluminação.

Por fim, os agentes que representam os atuadores (nesta simulação os elementos reais), refletem o ´cérebro´ desses elementos reais, uma vez que são os agentes que planejam o que se deve fazer.

IV. Conclusões e perspectivas

O texto apresentou uma revisão bibliográfica com os assuntos base desta pesquisa: doença Alzheimer, IoT, SMA, Prometheus e Jason. Também mostrou trabalhos relacionados ao tema e às tecnologias a serem utilizadas.

O trabalho apresentou a modelagem de aspectos estruturais e funcionais da simulação, que foi justamente o recurso para avaliar a proposta. O ambiente Jason e a linguagem AgentSpeak(L) se mostraram ferramentas adequadas, não só para a especificação de ambiente e agentes, mas útil para simular e avaliar 5 situações de comportamento do paciente.

Este trabalho mostra que a integração de serviços como Jason e atuadores Sonoff pode ser um recurso interessante e plausível para implementar monitoramento de pessoas doentes em cômodos e planejar e executar automaticamente ações que minimizem a intervenção de cuidadores, uma vez que SmartTV e iluminação, por exemplo, podem ser acionados modificando o estado emocional do paciente.

Registra-se que esta simulação realizada teve como base a simulação criada e integrada por Santos [4], uma vez que o trabalho simulou, usou sensores, atuadores em ambiente real e integrou o Jason com a o relé de atuação Sonoff.

A. Trabalhos futuros ou perspectivas

A pesquisa que vem sendo realizada na instituição, envolvendo Sistemas Multiagentes e Automação está em fase embrionária. Por isso, sugere-se alguns trabalhos futuros:

- projetar e implementar um istema Web para gestão de preferências do paciente vindas do cuidador. Esse sistema deve ser integrado ao simulador, como realizado por Costa e Zamberlan [16], por exemplo;
- realizar a implementação de novos agentes atuadores para ar condicionado e janela. Juntos a um sensor de temperatura com intuito de tornar o ambiente mais dinâmico e agradável;
- desenvolver um sistema fora do ambiente de simulação para uso em situação real, que contará com uma uma rede neural artificial (RNA) para armazenar os dados obtidos no sensoreamento e, possivelmente, implementar novas melhorias base nos resultados.

Referências

[1] F. P. D. França *et al.*, *Uma Arquitetura para o Monitoramento de Pessoas com a Doença de Alzheimer*. Goiânia, GO, Brasil.: Tese de Doutorado. Universidade Federal de Goiás, 2019.

[2] E. G. Carmo, M. S. Zazzetta, G. F. Junior, P. N. Micali, P. F. Moraes, and J. L. R. Costa, "A utilização de tecnologias assistivas por idosos com doença de Alzheimer," *Revista Kairós-Gerontologia*, vol. 18, no. 4, pp. 311–336, 2015.

[3] A. dos Santos Vasconcelos and R. S. d' Alencar, "O impacto socioemocional da doença de alzheimer sobre o cuidador familiar do idoso." *Memorialidades*, vol. 6, no. 12, pp. 107–127, 2009.

[4] A. K. D. Santos and A. Zamberlan, *Sistemas Pervasivos Integrados Por Agentes Inteligentes Em Jason E Raspberry Pi*. Santa Maria, RS, Brasil. Disponível em https://tfgonline.lapinf.ufn.edu.br: Trabalho de Conclusão de Curso Ciência Da Computação - Universidade Franciscana (UFN), 2021.

[5] J. C. Maia, J. F. V. Coutinho, C. R. d. Sousa, R. G. B. Barbosa, F. R. d. N. Mota, M. B. Marques, R. d. R. L. Silva, and R. B. d. S. Lima, "Tecnologias assistivas para idosos com demência: revisão sistemática," *Acta Paulista de Enfermagem*, vol. 31, pp. 651–658, 2018.

[6] B. P. Santos, L. A. Silva, C. Celes, J. B. Borges, B. S. P. Neto, M. A. M. Vieira, L. F. M. Vieira, O. N. Goussevskaia, and A. Loureiro, "Internet das coisas: da teoria à prática," *Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuudos*, vol. 31, p. 16, 2016.

[7] R. F. Perozzo, *Framework para integração entre ambientes inteligentes e o sistema brasileiro de TV digital*. Porto Alegre, RS, Brasil.: Tese de Doutorado em Engenharia Elétrica - Universidade Federal do Rio Grande do Sul (UFRGS), 2011.

[8] H. Mattede. (2023, abril) O que é relé? como funciona um relé? [Online]. Available: https://www.mundodaeletrica.com.br/o-que-e-rele-como-funciona-um-rele/

[9] L. F. C. Cardoso and U. Dias, "Sistema de automação residencial via rede celular usando microcontroladores e sensores."

[10] G. Evolução. (2023, junho) Acionadores. [Online]. Available: https://grupoevolucao.com.br/livro/Tecnologia_Assistiva/acionadores.html

[11] J. F. Hübner, "Um modelo de reorganização de sistemas multiagentes." Ph.D. dissertation, Universidade de São Paulo, 2003.

[12] I. G. L. d. SILVA, "Projeto e implementaçao de sistemas multi-agentes: O caso tropos," Master's thesis, Universidade Federal de Pernambuco, 2005.

[13] R. H. Bordini, J. F. Hübner, and M. Wooldridge, *Programming multi-agent systems in AgentSpeak using Jason*, M. Wooldridge, Ed. Wiley, 2007.

[14] V. M. Berny, F. M. Jeannes, A. C. da Rocha Costa, and A. R. Du Bois, "Prometheus: Metodologia de modelagem utilizada para a simulação de agentes da construção naval," *Anais SULCOMP*, vol. 4, 2008.

[15] J. Sutherland, *Scrum: a arte de fazer o dobro do trabalho na metade do tempo*. Leya, 2016.

[16] Éderson Guterres Costa and A. Zamberlan, *Webservice Para Integração De Dados Entre A Ferramenta De Simulação Maspn E Seu Portal Web*. Santa Maria, RS, Brasil. Disponível em https://tfgonline.lapinf.ufn.edu.br: Trabalho de Conclusão de Curso Sistemas De Informação - Universidade Franciscana (UFN), 2021.