VoiceRescuer: Interagindo com Aplicações Móveis Utilizando Voz em Situações de Emergências

Fábio Vinícius Pontes Pereira, Renato Lima Novais

Grupo de Pesquisa em Sistemas Distribuídos, Otimização, Redes e Tempo-Real (GSORT)
Instituto Federal de Educação, Ciência e Tecnologia da Bahia (IFBa)
Av. Araújo Pinho, no 39 - Canela - Salvador - BA - CEP: 40.110-150
{fabiopontes, renato}@ifba.edu.br

Resumo – Na área de gerenciamento de emergência, muitas soluções já foram criadas, entretanto no que diz respeito ao pedido imediato de ajuda para os órgãos competentes quando acontece uma situação emergencial, ainda existem lacunas a serem preenchidas. A maior parte das ferramentas existentes disponíveis para esse tipo de circunstância exige o manuseio do aparelho, o que pode ser impossível a depender dos ferimentos sofridos pelas vítimas ou mesmo o fato que elas podem se encontrar impossibilitadas de se moverem, como ocorre em situações de desabamentos, soterramentos, etc. Além disso, a maioria dessas ferramentas não possuem integração com os sistemas utilizados pelos órgãos de apoio como o corpo de bombeiros, por exemplo, o que dificulta o envio imediato do pedido de socorro. O presente trabalho propõe um aplicativo para smartphones capaz perceber a ocorrência de uma possível situação emergencial e permitir que os acidentados possam solicitar apoio através de comandos de voz. Além disso, busca-se utilizar os sensores ubíquos dos dispositivos móveis para prover localização e informações do ambiente para apoiar os centros de comando e acelerar o processo de resgate. Foi realizado um estudo com participantes que destacou o potencial da solução desenvolvida.

Palavras-chave – smartphone, comando de voz, detecçao de queda, emergência

I. Introdução

Diariamente, as pessoas estão propensas a sofrer diversas situações de emergência. Catástrofes naturais como terremotos e tempestades estão constantemente ocorrendo em diversos países. Além dos fenômenos naturais, também existem outras situações que podem acontecer. Como exemplos, podem ser citados: acidentes veiculares, incêndios e até mesmo atos inesperados como os ataques terroristas ocorridos nos Estados Unidos em 11 de setembro de 2001 [1].

Com a constante evolução da computação, muitas soluções ao longo do tempo têm sido desenvolvidas para dar suporte aos processos envolvidos no gerenciamento de emergências. A rapidez com que as tecnologias de transmissão de dados são aprimoradas, bem como a eminência de novos meios de acesso à informação são fatores que contribuem para esse cenário.

Ao longo dos anos, desde o surgimento do telefone, as circunstâncias emergenciais passaram a ser informadas aos centros de comandos dos bombeiros à medida em que iam acontecendo. Com o uso dos celulares, essa comunicação passou a ser ainda mais acelerada, já que para informar uma ocorrência, as pessoas não mais necessitavam procurar por orelhões públicos ou casas que tivessem uma linha disponível para solicitar ajuda.

Com o passar dos anos, os celulares foram se renovando gradativamente. A diminuição de tamanho de componentes computacionais como processadores e memórias e de sensores como acelerômetros, giroscópio, GPS, sensores de proximidade etc, ocasionou o surgimento dos atuais *smartphones* ou telefones inteligentes. Atualmente, tais aparelhos possuem recursos computacionais em franca expansão, sendo que a cada ano os modelos são substituídos por sucessores cada vez mais poderosos.

No que tange a área de *software*, pode-se observar que várias ferramentas foram desenvolvidas para beneficiar o uso dos dispositivos móveis. Na área de apoio e gerenciamento de emergência, muitos avanços são encontrados [2]–[4]. Os aplicativos de chat assim como as redes sociais também representam meios frequentemente utilizados para troca de informações sobre ocorrência de situações adversas. É comum receber notícias nas redes sociais e em aplicações de chat, a divulgação de eventos como incêndios, explosões, desabamentos e alagamentos através de fotos, vídeos e áudios capturados por usuários próximos ao acontecimento. Além disso, os dispositivos móveis também são utilizados para fazer chamadas ao corpo de bombeiros e demais órgãos envolvidos no resgate e salvamento de vítimas.

Apesar da existência das ferramentas citadas, o processo de comunicação envolvido no *report* e solicitação de apoio em emergências ainda possui obstáculos a serem superados. Principalmente no que diz respeito ao pedido de socorro pelas vítmas no momento do incidente, as aplicações convencionais ainda oferecem limitações a serem tratadas.

Utilizando-se dos celulares, há a possibilidade de ligar para órgãos como o corpo de bombeiros. Em situações onde acontecem quedas, como em terremotos ou desabamentos, o aparelho pode ser arremessado para um local inalcançável. Em contrapartida, mesmo quando o aparelho permanece próximo, é possível que a vítima encontre dificuldades em movimentar seus membros, impossibilitando a manipulação do dispositivo.

Outro ponto importante a ser considerado é que boa parte das emergências são percebidas pelas pessoas após um intervalo de tempo do seu início. A depender do local da ocorrência, essa percepção pode demorar algum tempo e assim postergar o pedido de ajuda. Em alguns casos, a demora na solicitação de socorro pode significar a perda da vida pelas vítimas em estado mais grave.

Quando terremotos e tempestades atingem as cidades, observa-se a evidência das dificuldades relatadas anteriormente. Em um desabamento, por exemplo, as vítimas podem sofrer fraturas nos membros e em alguns casos permanecerem conscientes necessitando de um socorro urgente. Há alguns anos, um terremoto no Haiti deixou uma vítima consciente embaixo dos escombros, porém ela não conseguia mover os membros pois estavam cobertos por terra [5]. O resgate demorou a encontrá-la e somente aconteceu graças ao fato de uma equipe de reportagem ter ouvido os seus gritos no local.

Para os órgãos que atuam no resgate, saber da localização exata de acidentados pode acelerar bastante o procedimento. Possibilitar comunicação com os envolvidos em um acidente pode favorecer os processos de tomada de decisão dos órgãos competentes. Desse modo, saber informações importantes como a quantidade de vítimas naquele ambiente e o seu respectivo estado ou as condições em que o local se encontra poderia aumentar a rapidez e eficácia do salvamento. Um aplicativo capaz de perceber e informar uma situação emergencial no momento em que ela ocorre, além de possibilitar uma interação eficiente com o usuário de forma integrada a um sistema de gerenciamento de emergências, pode beneficiar o processo de resgate e salvamento.

Os *smartphones* modernos, tendo em vista suas capacidades computacionais e sensoriais, são meios bastante promissores de aceleração de resgate em situações emergenciais. Através do aproveitamento adequado de tais recursos, esses dispositivos apresentam-se como forma de aumentar a rapidez, eficiência e eficácia do salvamento das vítimas.

No intuito de aproveitar as faculdades sensoriais presentes na maior parte dos dispositivos móveis modernos, bem como as possibilidades de comunicação através das redes sem fio, o presente trabalho propõe o desenvolvimento de uma solução capaz de identificar automaticamente uma possível situação emergencial e fornecer ajuda às vítimas. Trata-se de uma aplicação capaz de perceber um contexto emergencial e solicitar ajuda às equipes de socorristas, graças à integração direta a um sistema de gerenciamento de emergência, o RESCUER [6].

Desse modo, para fazer com que o *smartphone* possa ter os seus recursos devidamente aproveitados e melhorar o processo de apoio às vítimas, propõe-se:

- proporcionar uma maneira rápida e automática de identificação de ocorrências emergenciais logo após o seu acontecimento. Trata-se de uma forma de acelerar o pedido de ajuda, já que quanto antes a crise é percebida, maior será a probabilidade de obtenção do resgate. Consequentemente, quanto mais rápido o socorro estiver disponível para as vítimas, maiores serão as chances de sucesso, uma vez que podem existir casos de ferimentos graves onde a rápida assistência pode ser crucial;
- fornecer interação com as vítimas através de comandos de voz, já que assim, torna-se possível a utilização do dispositivo de forma prática, rápida e intuitiva;
- coletar e reportar informações do local em que o desastre tenha acontecido, de modo a informar as condições ambientais e consequentemente os desafios que as equipes de resgate podem encontrar;
- utilizar os recursos do aparelho para informar a localização onde ocorreu a emergência e da vítima, a fim de se facilitar os processos de busca;
- realizar o envio das informações ao RESCUER [6], um sistema de gerenciamento de emergência capaz de receber esses dados e utilizá-los como insumo para as tomadas de decisão inerentes ao planejamento e execução do resgate.

A ideia é aproveitar ao máximo os sensores disponíveis nos *smartphones* para utilizá-los como uma espécie de alarme portátil e fornecer as informações adquiridas ao RESCUER [6]. A princípio, a aplicação vai se basear no acelerômetro para identificar se ocorreu uma queda e então iniciar um processo de comunicação com o usuário através de comandos de voz. A identificação da queda funciona como um gatilho a ser disparado para todas as possíveis situações em que o usuário possa ter caído. A partir de então, um processo de perguntas e respostas é iniciado entre o aparelho e o acidentado, para que sejam colhidos maiores detalhes a respeito da ocorrência.

A escolha da forma de interação através de comandos de voz se dá ao fato de que ela permite que o usuário possa utilizar o aparelho mesmo que ele esteja a pequenas distâncias. Outro benefício é que o acidentado não precisa necessariamente estar segurando o aparelho para poder manipulá-lo, o que pode ser de importância crucial para aqueles que, devido a ferimentos, não estejam conseguindo movimentar seus membros.

Além dos microfones e auto-falantes, o presente trabalho também fará uso do GPS para informar o posicionamento do seu usuário no momento da interação, facilitando assim a definição de estratégias de salvamento, como por exemplo determinar as melhores opções para se chegar aos pontos onde se encontram as vítimas. As câmeras disponíveis também serão aproveitadas para colher imagens do ambiente no intuito de informar as condições em que o local se encontra.

O fato de ser uma aplicaçao *mobile*, faz com que essa ferramenta esteja disponível para uma quantidade considerável de usuários, considerando a atual popularização dos dispositivos móveis. Desse modo, o aplicativo também é classificado como uma aplicação *crowdsourcing* para compartilhamento de informações, já que a depender da proporção da crise, um conjunto de usuários poderá fornecer dados online sobre a emergência, proporcionando tomada de decisões através da análise dessas informações.

A figura 1, mostra o esquema de funcionamento da aplicação proposta, destacando a integração que é realizada com o RESCUER [6]. Em meio ao funcionamento geral do RESCUER [6], foi acrescentada uma ocorrência de um terremoto percebida pelo aplicativo, o qual atua junto ao usuário tentando confirmar a ocorrência e obter as demais informações em caso positivo. Essas informações, segundo ilustrado em 1 são enviadas ao RESCUER [6] que, por conseguinte, as organiza e as repassa para os centros de comando e demais órgãos competentes.

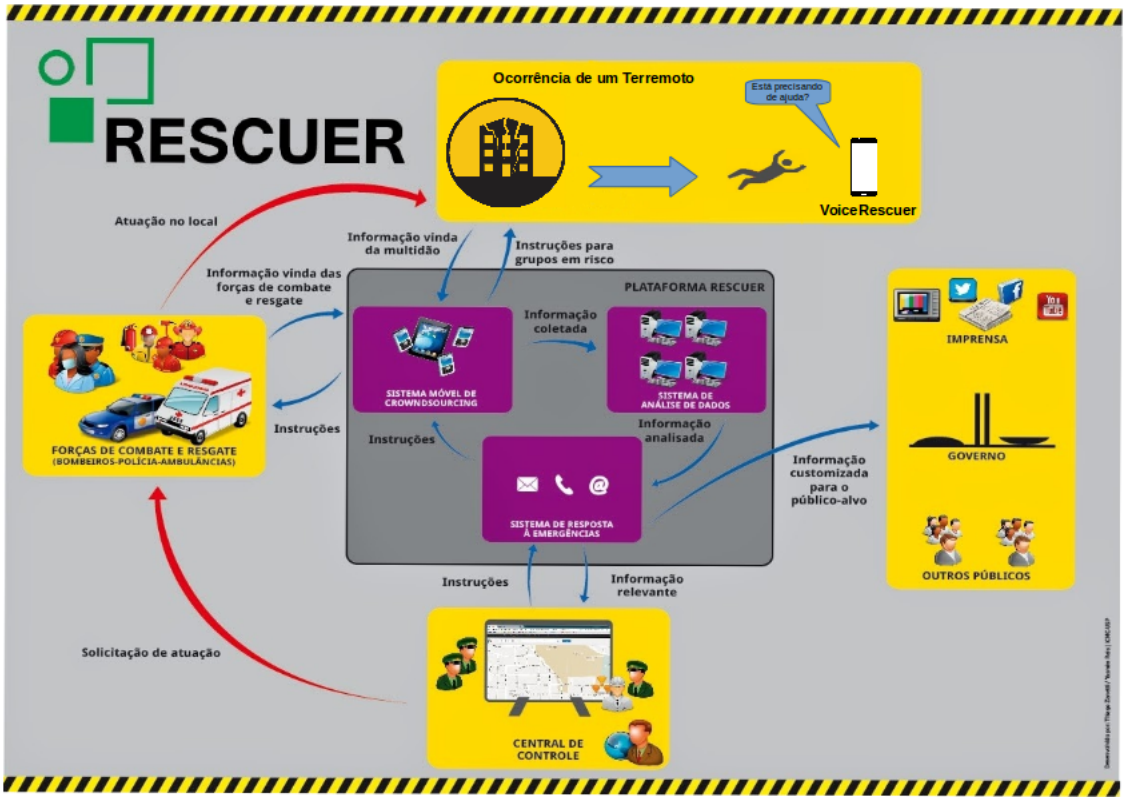

Figura 1: Esquema de funcionamento do VoiceRescuer

Além desta introdução, o texto está organizado da seguinte forma: a seção II apresenta os conceitos chaves que norteiam o desenvolvimento da solução proposta. Em seguida, é feito uma análise dos trabalhos correlatos, evidenciando as semelhanças e diferenças em relação ao projeto desenvolvido. Por fim, é feita uma discussão a respeito do projeto desde sua concepção até a concretização do respectivo *software*.

II. Fundamentação Teórica

Desde o surgimento do primeiro computador nos anos 50 até os dias atuais, a computação tem se tornado cada vez mais inserida na vida dos usuários. Isso se deve à evolução de diferentes áreas em conjunto, tanto no que diz respeito aos componentes de *hardware* como *software*. A cada ano, o mundo da tecnologia presenteia seus usuários com a exponencial miniaturização e empoderamento de componentes como processadores, memórias e sensores. Além disso, a evolução dos sistemas operacionais e linguagens de programação permitem que a computação seja disponibilizada para além dos computadores de mesa. Dessa forma, a tecnologia está se desenvolvendo para além da criação de notebooks com mais memória ou capacidade de processamento. Observa-se a imersão da computação no cotidiano do usuário em diferentes aparelhos com diferentes funções. Estudiosos afirmam que em um futuro próximo os usuários poderão levar consigo a computação para qualquer ambiente, independente de onde se esteja [7]–[11]. Todo esse fenômeno foi introduzido por Mark Weiser em 1991 e será discutido em maiores detalhes na próxima seção.

A. Computação Ubíqua

Segundo Weiser [7], "A tecnologia mais profunda é aquela que desaparece". Com essa frase, ele introduz a ideia da revolução tecnológica onde a computação passa a ser inserida na vida das pessoas sem que elas percebam, ou seja, de forma invisível e indistinguível. Nesse conceito, define-se que a computação passa a estar presente em diferentes dispositivos além do computador pessoal, podendo ser encontrada em objetos como canetas, canecas, utensílios domésticos, etiquetas de roupas, interruptores de luz, etc.

Em [7], Weiser explica o que exatamente significa essa ideia de invisibilidade tecnológica. Segundo ele, tal desaparecimento é uma consequência não da tecnologia, mas da psicologia humana. Sempre que as pessoas aprendem algo suficientemente bem, elas passam a agir inconscientemente, ou seja, de forma automática. Weiser cita o exemplo de que quando se vê uma placa na rua, a informação é absorvida sem que a leitura seja executada conscientemente. Ele previa que a computação seria incorporada às rotinas diárias de tal modo que o seu uso seria inconsciente.

Quando se fala em invisibilidade no contexto da computação ubíqua, tem-se a ideia de criar computadores de todos os tipos e tamanhos que possam estar imersos nas atividades do dia-a-dia de cada pessoa [8]. Esses dispositivos devem trabalhar em colaboração, interconectados através de uma rede de comunicação sem fio adequada. Essa abordagem aponta o computador pessoal *desktop* como um dispositivo isolado das atividades diárias, ou seja, ao invés de seu uso se tornar automático e inconsciente, ele continua requerendo certo nível

de atenção.

Com o passar dos anos, a computação ubíqua tem se tornado cada vez mais próxima da realidade. Gradativamente, computadores estão sendo embutidos no cotidiano e interagem através de diferentes formas. Tudo isso devido ao desenvolvimento das redes sem fio, o contínuo aumento do poder computacional, melhoria no gerenciamento de energia das baterias e o aparecimento de arquiteturas de software flexíveis [9].

O crescimento da computação ubíqua se dá através do desenvolvimento de duas áreas chaves: a computação móvel e a computação pervasiva. A computação móvel diz respeito à capacidade de mover os serviços computacionais junto com o usuário. Como consequência, a computação se torna sempre presente, em qualquer lugar onde o usuário esteja. Desse modo, em qualquer ambiente sempre haverá dispositivos capazes de prover os serviços necessários [9].

A imersão da computação em dispositivos de diferentes tamanhos e utilidades, associada ao acesso às redes sem fio, têm permitido que os serviços computacionais sejam levados a diferentes ambientes como praias e aeroportos, dentre outros. Entretanto, o conceito fundamental defendido pela computação móvel é que o modelo computacional não muda consideravelmente enquanto se move de um ambiente para outro. Tal ideia na prática ainda não é atingida de forma automática, ou seja, atualmente o usuário ainda precisa fazer ajustes manuais em seus dispositivos quando ocorre uma mudança de ambiente.

O princípio da computação pervasiva é tornar o computador invisível [9]. Isso significa que os dispositivos devem obter informações do ambiente no qual estão inseridos e com isso construir modelos computacionais. O ambiente por sua vez também deve ser capaz de identificar os dispositivos que chegam nele e estabelecer colaboração entre si. Os serviços da computação pervasiva podem ser disponibilizados através de modelos específicos para um ambiente, embutidos em computadores dedicados ou adicionar capacidades genéricas a computadores que sejam capazes de criar um modelo computacional para o ambiente.

Na próxima seção, serão discorridos maiores detalhes sobre o nascimento da computação ubíqua e os seus primeiros anos de pesquisa e desenvolvimento.

1) Origem da Computação Ubíqua: A computação ubíqua se originou nos laboratórios do Xerox Palo Alto Research Center (PARC) [10]. Em 1987, Bob Sprague, Richard Bruce, e outros membros propuseram fabricar telas planas que serviriam como dispositivos de entrada. A princípio, essas telas deveriam funcionar como lousas eletrônicas que ofereceriam maior interatividade a seus usuários permitindo inclusive se conectar e interagir com outros dispositivos. Esses quadros eletrônicos foram pensados para serem usados como dispositivos de entrada para canetas eletrônicas e imagens escaneadas.

Rapidamente, membros de outros laboratórios se voluntariaram para projetar o *hardware* e o *software* para esse novo sistema computacional que tinha como objetivo proporcionar facilidade de uso e funcionamento transparente, além de permitir sua operação em rede com outros dispositivos. Esses "computadores de parede" foram projetados visando uma concepção totalmente diferente do paradigma do computador pessoal *desktop*, inspirando os pesquisadores a investirem na ideia de se espalhar computadores por todo o ambiente de forma invisível.

Paralelamente, antropólogos da área de Trabalho, Práticas e Tecnologias (*Work Practices and Technology*), liderados por Lucy Suchman, passaram a observar a forma como as pessoas realmente usavam a tecnologia e não apenas a forma como elas pediam para usar a tecnologia. Suas observações influenciaram os projetistas a pensarem menos sobre os detalhes computacionais como acesso a memória ou número de *pixels* ou *megahertz* e muito mais na situação do uso da tecnologia. Diante de todo esse cenário, no início do ano de 1988, surge a Computação Ubíqua no Laboratório de Ciência da Computação (*Computer Science Laboratory - CSL*) do PARC [10]. No primeiro momento, ela foi vista como uma resposta radical ao que havia de errado com o computador pessoal, isto é, sua dificuldade de uso, sua exigência de atenção e seu isolamento de outras pessoas e atividades.

Um dos objetivos principais dos pesquisadores da computação ubíqua era colocar a tecnologia de volta ao seu lugar, ou seja, em segundo plano. Isso significa concentrar-se mais nas relações estabelecidas entre o homem, o ambiente e as demais pessoas, vislumbrando a imersão da tecnologia para dar suporte a tudo isso.

Os primeiros anos da computação ubíqua geraram alguns produtos inovadores. O projeto inicial da lousa eletrônica resultou no *LiveBoard* [11], um grande sistema interativo e colaborativo de desenho. Foram criados também os chamados computadores menores: o *ParcPad*, do tamanho de um livro e o *ParcTab*, de tamanho semelhante aos atuais *smartphones*, dentre outros. Todos esses dispositivos foram construídos de modo que pudessem funcionar em colaboração, visando uma estrutura computacional flexível que provia o reconhecimento não apenas do dispositivo, mas também da sua localização, situação, uso, conectividade, a quem pertencia, etc. Todos esses produtos marcaram o início de uma nova era da computação, ampliando o uso da tecnologia para além dos problemas solucionados pelo computador *desktop*.

2) Desafios da Computação Ubíqua: O maior desafio da computação ubíqua reside na integração entre a mobilidade em larga escala e a pervasividade [9]. Isso significa que qualquer dispositivo ubíquo seria capaz de incrementalmente construir modelos computacionais dinamicamente enquanto o usuário se move entre os vários tipos de ambientes. Deste modo, tais computadores embarcados seriam capazes de relembrar os ambientes em que o usuário já esteve ou mesmo se adaptarem a outros novos.

O mundo da computação ubíqua traz consigo desafios em diferentes esferas do conhecimento, seja social, tecnológica ou mesmo organizacional. No ramo da tecnologia, existem várias questões a serem resolvidas, principalmente ligadas ao *design* e arquiteturas computacionais capazes de realizar a configuração dinâmica de serviços ubíquos em larga escala.

Ao se desenvolver aplicações ubíquas, os engenheiros de *software* e *hardware* deverão conceber novas arquiteturas ou readaptar as já existentes para que essa nova realidade de serviços ubíquos possam ser disponibilizados. Além disso, novos requisitos deverão ser tratados, tanto funcionais como

não funcionais.

Para descrever as características do ambiente, bem como as informações necessárias para que a computação possa ser adaptável e consiga manter as preferências do usuário, será necessária a criação de modelos semânticos de alto nível [12]. Tais modelos deverão ser capazes de descrever os ambientes das tarefas dos usuários de modo a auferir quais as suas necessidades, permitindo que as aplicações possam se adaptar para prover serviços.

A infraestrutura na qual os ambientes ubíquos devem funcionar representa um enorme desafio. Ela deve estar apta a permitir a busca, adaptação e entrega de aplicações apropriadas ao ambiente computacional do usuário, ou seja, ao contexto no qual está inserido. Além disso, a infraestrutura deve suportar a heterogeneidade de dispositivos, já que no ambiente estarão disponíveis diferentes tipos com diferentes recursos e capacidades.

No que diz respeito a heterogeneidade de dispositivos e serviços a serem oferecidos, ainda existe a questão de se determinar quais são mais importantes para um usuário em um determinado contexto [12]. Mesmo que uma lista de tarefas prioritárias seja auferida, o sistema ainda precisa selecionar os serviços e componentes que devem ser ativados para que as aplicações selecionadas possam rodar tendo em vista suas particularidades e restrições.

No mundo da ubiquidade, um vídeo que estiver sendo visualizado pelo usuário em um *smartphone* pode ter sua exibição enviada para continuar em uma televisão no quarto ou mesmo em um *display* no carro, no momento em que um usuário passageiro está sendo conduzido ao trabalho. Em outras palavras, as aplicações podem ter suas execuções movidas de computador para computador ou mesmo de um ambiente para outro. Nesse caso, a infraestrutura deve estar preparada principalmente no que se refere às tecnologias de transmissão de dados e a escalabilidade dos dispositivos e aplicações.

O presente trabalho faz uso de propriedades da computação ubíqua, uma vez que está inserindo uma solução computacional no dia-a-dia do usuário de forma transparente. Desse modo, a aplicação herda as características da computação pervasiva presentes em um *smartphone*, ou seja, ela poderá ser levada com o usuário para seus diversos ambientes, fazendo com que ele tenha sempre consigo uma ferramenta de apoio caso alguma emergência ocorra.

B. Sensoriamento em Dispositivos Móveis

Os *smartphones* evoluíram tanto que praticamente todas as atividades, antes executadas exclusivamente em um computador pessoal, agora podem ser realizadas tranquilamente com o seu uso. Isso se deve em grande parte à contínua evolução do hardware e das tecnologias de comunicação, que possibilitam incorporar poder de processamento e acesso a rede para tais dispositivos.

O telefone móvel é considerado o primeiro computador verdadeiramente pervasivo [13]. Ele está sempre com seus usuários, apoiando não somente a comunicação como também outras atividades. Além disso, através dos seus poderosos sensores eles podem ver, ouvir e sentir os ambientes nos quais estão inseridos.

Nos *smartphones* modernos, podem ser encontrados diferentes tipos de sensores embutidos [14], tais como: acelerômetro, bússola digital, giroscópio, microfone, Global Position System (GPS) e câmeras. Tudo isso abre caminho para o surgimento contínuo de aplicações voltadas para objetivos diversos como segurança, redes sociais, monitoramento ambiental, assistência médica, etc. Elas utilizam do poder desses sensores para fornecer funcionalidades que não seriam possíveis em um computador pessoal no seu formato tradicional. Estima-se que esse novo ramo de aplicativos tende a crescer incomensuravelmente, impactando nas atividades diárias dos usuários.

Os sensores começaram a ser introduzidos nos celulares à medida em que eles foram se amadurecendo como plataforma computacional. O acelerômetro, por exemplo, passou a ser utilizado inicialmente para determinar a orientação na qual o usuário está segurando o aparelho e consequentemente orientar a interface no modo retrato ou paisagem [14].

Os novos modelos de *smartphones*, como as novas versões do *iPhone*, são dotados de giroscópio, bússola, acelerômetro, sensor de proximidade, sensor de luminosidade, câmeras frontais e posteriores, microfone, GPS, WiFi e *Bluetooth*. O sensor de proximidade é usado para identificar se o usuário está com o celular encostado no rosto para falar, fazendo com que o *touch screen* permaneça bloqueado para economizar energia e evitar que alguma tecla seja pressionada indevidamente.

Os sensores de luminosidade servem para ajustar o brilho da tela. O GPS permite o uso da localização em aplicações como redes sociais, aplicativos de navegação e até mesmo aqueles utilizados para descobrir onde o aparelho se encontra no caso de perda ou roubo. A bússola e o giroscópio apoiam os processos de orientação e direção, ou seja, também contribuem para a localização.

Dentre os sensores relatados, merece destaque o acelerômetro. Ele é responsável por permitir aos *smartphones* identificar os movimentos que o usuário esteja executando enquanto transporta o aparelho. Através das informações advindas do acelerômetro, é possível auferir algumas atividades que o usuário possa estar executando. Exemplos são: corrida, caminhada, queda, etc.

O acelerômetro, os microfones e as câmeras fornecem às aplicações os insumos necessários para a identificação do contexto do usuário [14]. A contínua coleta de áudio através dos microfones permite classificar um conjunto de sons e consequentemente perceber o ambiente onde se encontra (como uma determinada cafeteria, por exemplo) e/ou as atividades do usuário naquele momento (como conversando, dirigindo, fazendo café, etc). As câmeras, além de serem utilizadas para fotografia, também possibilitam a ativação de aplicações com base em um movimento ocular realizado. As informações colhidas do acelerômetro em conjunto com a estimativa de localização proveniente do GPS levam o sistema a identificar a forma de locomoção do usuário, ou seja, se ele está andando, de bicicleta, carro, metrô, ônibus, etc.

O presente trabalho destina-se a utilizar as informações de aceleração obtidas do acelerômetro e assim, oferecer apoio para a maior parte de incidentes que possam ser auferidos

através da detecção de quedas. Desse modo, espera-se proporcionar ajuda em situações como desabamentos, terremotos, ou qualquer tipo de acidente onde a vítima e/ou o aparelho seja lançada ao solo. A próxima seção caracteriza o processo de detecção de quedas em dispositivos móveis e como ele pode auxiliar a aplicação no seu propósito.

C. Detecção de Quedas

A detecção de queda é um fenômeno bastante estudado, já que o mesmo é utilizado em diversas aplicações de assistência a idosos. O aumento desse tipo de população no mundo, bem como o fato dela possuir um alto índice de acidentes envolvendo queda [15] contribuíram para avanços significativos nessa área. Dentre esses avanços, pode ser considerado o surgimento de aplicações capazes de monitorar os idosos para esse tipo de ocorrência.

Muitas situações emergenciais têm como consequência o lançamento das vítimas ao chão. Terremotos, desabamentos, acidentes de automóveis, e pousos forçados de aeronaves são exemplos reais onde isso pode acontecer. Nesses casos, a identificação do ocorrido pode ser feita utilizando as técnicas de detecção de queda utilizadas nas aplicações de assistência médica.

Existem três tipos de abordagens de detecção de quedas [15]: baseado no uso de dispositivos, em sensores do ambiente ou câmeras (visão).

Como o presente trabalho tem como um dos objetivos a utilização dos acelerômetros dos *smartphones*, será enfatizada a primeira abordagem, ou seja, a baseada no uso de dispositivos. Nela serão destacadas as principais técnicas atualmente utilizadas para se interpretar os dados obtidos desses sensores e assim promover a percepção dos fenômenos almejados. As demais poderão ser estudadas em maiores detalhes através da verificação da referência bibliográfica.

A abordagem baseada no uso de dispositivos tem como base a utilização de sensores embutidos para identificar o movimento e a localização do corpo do indivíduo. De acordo com o tipo de uso desses sensores, tal abordagem pode ser dividida nas seguintes categorias [15]:

1) Acelerometria: A acelerometria consiste em medir a aceleração do corpo ou de partes do corpo. Ela se baseia no uso do acelerômetro, que pode ser encontrado na maioria dos *smartphones* comercializados. Nessa técnica, uma queda é detectada quando a aceleração negativa aumenta repentinamente e a orientação do corpo monitorado mudou de vertical para horizontal. Um sensor de pressão barométrica também pode auxiliar nesse processo, através de técnicas baseadas na variação da altitude. Assim, um dispositivo é amarrado à cintura do indivíduo para fazer a colheita da aceleração e da mudança de pressão.

2) Fusão da acelerometria com sensores de postura: Um acelerômetro de dois eixos foi utilizado em [16], combinado a um sensor de postura para detectar quedas. Nesse trabalho, os autores construíram um protótipo para uso no punho que continha um detector de quedas integrado a um dispositivo de motinoração de saúde. O sistema possuia funcionalidade de enviar relatórios com as informações obtidas.

3) Inatividade com acelerometria: O acelerômetro fornece informações detalhadas sobre o comportamento, através das quais é possível indentificar se existe uma atividade física sendo realizada ou não. Além disso, tais informações podem apresentar detalhes como frequência, intensidade e duração dos movimentos. Tendo como base tal característca, em [17] foi proposto um sensor inteligente para detecção de quedas. O software utilizado nesse projeto colhia os dados através do sensor e os transmitia via rede.

4) Acelerometria em três eixos: Acelerômetros de três eixos são capazes de detectar a aceleração em três eixos de direção. Em [18], múltiplos acelerômetros desse tipo foram utilizados em conjunto e aclopados ao corpo para realizar detecção de queda e deduzir quais partes sofreram maior impacto durante o acidente. Tal modelo transmite as informações provenientes de sensores espalhados pelo corpo, determinando uma possível queda quando a aceleração excede os limites habituais.

A detecção de queda baseada em sensores do ambiente utiliza-se da combinação de dados visuais e auditivos para deduzir uma ocorrência. Em [19], por exemplo, uma rede *wireless* emprega o uso de múltiplos sensores e diferentes modalidades de detecção de eventos para monitorar pacientes da terceira idade. Esse sistema utiliza-se de sensores de imagem e raciocínio baseado em visão para analisar os dados obtidos dos outros sensores. Outra forma de se monitorar um ambiente é através de informações vibracionais como em [20], onde faz-se uso de um detector de quedas baseado na vibração do solo. Abordagens baseadas no uso de câmeras também podem oferecer benefícios, pois elas permitem detectar diversos eventos simultaneamente, de forma menos intrusiva.

Para que o aplicativo possa ter uma boa usabilidade, deve-se considerar as situações em que os mesmos estarão inseridos. Tendo em vista as emergências que podem ocorrer, a interação através da voz mostra-se como a opção mais viável. Na próxima sessão, será caracterizada a utilização de comandos de voz nos sistemas, bem como as aplicações desse gênero que são encontradas nos dispositivos móveis.

D. Comandos de Voz em Dispositivos Móveis

A presença de múltiplos sensores nos dispositivos móveis modernos abriu espaço para o surgimento de aplicações baseadas em outros tipos de interface. Atualmente, além dos mouses, apontadores e superfícies multi-toques, os *smartphones* também se beneficiam dos microfones e alto-falantes para proporcionar maior inclusão e interatividade. Todos esses recursos tornam o uso desses dispositivos ao alcance de usuários com necessidades de usabilidade diferentes entre si, como por exemplo, os deficientes visuais que através dos comandos de voz podem acessar a maior parte das funções dos seus aparelhos.

A interatividade através do uso da voz tem sido utilizada em muitos domínios de aplicação ao longo dos anos. Tais sistemas, conhecidos como Sistemas de Dialogo Falado (SDF), permitem aos usuários interagirem com os dispositivos através da fala para obter informações, conduzir transações ou realizar outras tarefas para solução de problemas [21]. Eles variam

desde o sistemas interativos de resposta por voz com reconhecimento de um conjunto restrito de palavras até a interação mais abrangente através da linguagem natural.

Os SDF geralmente operam em um domínio restrito. A principal característica que os distingue é o grau de conversação que eles podem estabelecer. Aqueles que apresentam um diálogo direto tendem a direcionar ao usuário uma série de perguntas. Esses questionamentos tendem a obter respostas curtas como sim ou não, ocasionando maior incidência de sucesso no diálogo.

Uma abordagem alternativa é empregar uma estratégia mista onde o sistema torna-se mais flexível em lidar com as necessidades do usuário, consequentemente processando entradas linguísticas mais complexas. Nesse grupo de sistemas, tenta-se estabelecer um conjunto maior de restrições e características relacionadas a uma tarefa particular. Por permitir maior flexibilidade, pode haver maior incidência de erros e o diálogo pode se tornar mais confuso para o usuário. Como forma de mitigação, muitos sistemas apresentam soluções híbridas, alternando para um domínio mais restrito de diálogo quando um tratamento mais amplo está apresentando problemas.

Os SDF podem ser encontrados em diversos domínios de aplicação. Atualmente, a área que está em franca expansão é o desenvolvimento de interfaces de usuário para dispositivos móveis [22]. Os assistentes virtuais como o Siri da Apple, o Cortana da Microsoft e o Google Now são exemplos de agentes computacionais que auxiliam seus usuários a executarem tarefas simples através de comandos de voz. Assim, esses recursos permitem acionar as funções do aparelho viabilizando atividades como a realização de chamadas, manipulação da agenda ou mesmo uma pesquisa na internet.

Além dos assistentes virtuais dos *smartphones* modernos, os SDF podem ser encontrados inseridos em muitos sistemas robóticos, espalhados por diferentes tipos de ambientes como casas, escolas, escritórios e hospitais. A necessidade desse tipo de sistema com essa capacidade de comunicação é crescente em muitos setores da sociedade.

1) Arquitetura dos Sistemas de Diálogo Falado: No que se refere a arquitetura, os SDF são constituídos por componentes que tratam do reconhecimento da fala, entendimento da linguagem, gerenciamento do diálogo, geração e síntese da fala [21], [22].

O primeiro passo é o recebimento da fala como entrada, a qual deverá ser processada e gerar como resultado um conjunto sequencial de palavras que foram ditas. Essa fase é conhecida como reconhecimento da fala.

Para que as palavras reconhecidas possam ter utilidade, é preciso que além de reconhecê-las, o sistema consiga inferir o seu significado mais próximo naquele contexto. Esse processo corresponde ao entendimento da linguagem.

Sequencialmente, o sistema precisa selecionar uma ação adequada de acordo com o que foi dito e compreendido. Caso haja mais de uma ação que se enquadre como resposta, o sistema poderá realizar mais perguntas a fim de se obter maiores detalhes a respeito do que se pretende. A tomada de decisão é sempre realizada tendo como base tudo aquilo que é compreendido sobre a situação corrente, incluindo o histórico do diálogo e as informações do contexto. O módulo responsável pela escolha da ação de resposta é o gerenciador do diálogo.

O fluxo do diálogo é basicamente determinado pelo seu gerenciador. Por exemplo, ele pode ter mapeado como resposta a um conjunto de palavras recebido, a geração de uma sequência de palavras a serem geradas no módulo de geração de falas, a fim de comunicar algo ao usuário. Nesse caso, a saída do módulo de geração é sintetizada para um sinal de fala, através do módulo de síntese.

O gerenciador de diálogos é o principal componente na arquitetura de um SDF. Ele controla todo o fluxo de interação, sendo responsável pela tomada de decisão do sistema em cada situação.

2) Sistemas de Diálogo Falado em Smartphones: Atualmente, os principais sistemas operacionais para *smartphones* oferecem suporte a comando de voz, através dos assistentes virtuais. Seja com o Google Now, o Siri dentre outros. A maior parte dos dispositivos estão adotando alternativas que estão se aprimorando gradativamente enquanto instrumento de inclusão no uso desses aparelhos.

O Google Now é um assistente virtual pessoal disponível para aparelhos com o sistema operacional Android ou iOS. Ele permite, através do uso da voz, realizar envio de mensagens, manipular agenda, pesquisar endereços e informações na internet [23]. O Google Now oferece acesso a voz a uma gama de serviços desde que o usuário tenha fornecido permissão de acesso. A interface com o usuário é através da linguagem natural. Ele apresenta interface de busca individuais e focadas em imagens, vídeos, localidades, artigos, mídia social em tempo real, etc.

O assistente pessoal virtual nativo do iOS é o Siri. Ele permite fazer uso de praticamente todas as funções do aparelho através de comando de voz. Não requer treinamento inicial e o aprendizado de novas palavras é feito progressivamente à medida em que são ouvidas, utilizando como auxílio as informações do contexto [23], [24]. O Siri foi disponibilizado pela Apple pela primeira vez no iPhone 4S, tornando-se parte integrada às versões posteriores do aparelho e *tablets*.

Uma limitação de ambos os assistentes é que eles necessitam que o aparelho esteja conectado a internet para operarem. Além disso, ambos têm o seu uso restrito a plataformas particulares, ou seja, iOS no caso do Siri e Android ou iOS para o Google Now.

E. Gerenciamento de Emergências

Situações emergenciais estão constantemente acontecendo no dia a dia. Por esse motivo, o gerenciamento de emergências tem sido alvo de inúmeros estudos que tentam aprimorar métodos e tecnologias para gradualmente melhorar a prevenção e atuação para essas ocorrências.

Segundo [25], emergência é um termo utilizado para designar um evento repentino e imprevisto o qual exige que medidas imediatas sejam tomadas para minimizar suas consequências. Como exemplos, podem ser destacados: incêndios, terremotos, maremotos, desabamentos, ataques terroristas, etc.

O gerenciamento de emergências pode ser concebido como um processo de desenvolvimento e implementação de políticas preocupadas com [26]:

- **mitigação:** decidir o que fazer quando existe um risco ao bem estar social, a saúde, a segurança, etc., e implementar um programa de redução do risco;
- **prevenção:** desenvolver um plano de resposta e treinamento aos socorristas para salvar vidas e reduzir os danos do desastre, incluindo a identificação de recursos críticos e o desenvolvimento de acordos necessários entre as instituições de apoio;
- **resposta:** prover apoio emergencial e assistência, reduzindo a probabilidade de impactos secundários e minimizando problemas para operações de recuperação;
- **recuperação:** prover suporte imediado durante o período inicial de recuperação até que a normalidade se reestabeleça.

Tendo em vista o gerenciamento de emergências em áreas industriais e eventos de grande porte como as olimpíadas, a copa do mundo, dentre outros, destaca-se o RESCUER [6]. Esse projeto visa desenvolver uma solução baseada em computador e *crowdsourcing* para apoiar centros de comando de forma rápida [6]. Ele é composto de quatro componentes principais:

- *Soluções de Compartilhamento Crowdsourcing:* o objetivo principal dessas soluções é fornecer apoio a testemunhas, socorristas e ao centro de comando oferecendo informações sobre situações de emergência, considerando os diferentes tipos de *smartphones* que podem ser usados e como as pessoas interagem com os mesmos sob estresse;
- *Soluções de Análise de Dados:* incluem abordagens para a integração de dados de diferentes forças operacionais, bem como para a combinação, filtragem e análise das informações enviadas pela multidão somadas às informações que já estão na rede (dados abertos);
- *Ferramentas de Resposta a Emergências:* fornece ao centro de comando as informações atualizadas e relevantes, no formato adequado, para apoiar a tomada de decisões nas diferentes fases de uma emergência. Este componente tem como objetivo investigar metáforas diferentes de visualização que podem ser aplicadas aos dados de *crowdsourcing* analisados e agregados pela análise soluções de dados, para realizar uma comunicação clara e eficiente, com o centro de comando e controle;
- *Infraestrutura de Comunicação:* apoia o fluxo de informações entre a multidão e o centro de comando, mesmo quando a infraestrutura de comunicação tradicional está sobrecarregada.

Boa parte das informações que alimentam o RESCUER [6] são provenientes do uso de aplicações *Crowdsourcing*, principalmente através do uso dos *smartphones*. O envio das informações pode ser feito de forma interativa, onde o usuário acessa diretamente as aplicações para esse propósito, ou mesmo automaticamente, como ocorre no presente trabalho, por exemplo.

Crowdsourcing pode ser definido como a resolução distribuída de um problema [27]. Atualmente o termo *crowdsourcing* tem sido empregado para designar um novo modelo de negócio baseado na web, onde uma rede distribuída de usuários, colaborando entre si ou não, contribuem para a realização de uma atividade.

Dentre as categorias de aplicações *crowdsourcing* estão os sistemas de compartilhamento de informações [27]. Eles têm como objetivo compartilhar variados tipos de informação. Como exemplo de sites que se encaixam nessa categoria podem ser destacados: o *Wikipedia*, uma enciclopédia online onde qualquer usuário pode contribuir com informações; o *Yahoo! Answers*, um fórum de perguntas; o *Yahoo! Suggestion*, um sistema de sugestões e *feedbacks* em escala de internet, dentre outros.

No que se refere a esse tipo de aplicação para dispositivos móveis, ainda existem lacunas a serem preenchidas. Através do uso dos *smartphones*, as vítimas podem fornecer informações valiosas para o centro de comando, contribuindo para rápida localização e salvamento. Os valiosos recursos de sensores podem oferecer opções vantajosas ao processo de comunicação e consequentemente ao gerenciamento da emergência.

III. Trabalhos Relacionados

Tendo em vista os objetivos do trabalho proposto, bem como suas funcionalidades, foram levantadas algumas aplicações que apresentam características convergentes. Nessa análise, buscou-se verificar aplicativos ligados à detecção de quedas em pessoas que precisam ser monitoradas, como idosos, por exemplo. Além disso, foram analisadas ferramentas que utilizam comandos de voz em situações de emergência ou que oferecem suporte que se aproxime da solução proposta.

A. Disaster Voice Messaging Service

O Disaster Voice Messaging Service é um serviço de comunicação desenvolvido para ser usado em situações de desastres [2]. Ele transmite mensagens de voz entre seus usuários através de pacotes de comunicação e SMS. Esse *software* permite que usuários de dispositivos móveis possam enviar e receber mensagens de voz, conforme as etapas a seguir: primeiramente, o emissor grava uma mensagem de voz e realiza o seu *upload* para o servidor; na sequência, o destinatário recebe uma notificação SMS indicando que uma mensagem está disponível para ser checada; em seguida, o destinatário faz o download e ouve a mensagem disponibilizada; por fim, o emissor recebe uma notificação SMS de que a mensagem foi recebida por seu destinatário.

O sistema possui os seguintes requisitos:

- o emissor pode enviar mensagens de voz semelhantemente a chamadas telefônicas, utilizando um número de telefone;
- O emissor pode enviar mensagens de voz através de operações simples no ‘Disaster Kit‘, aplicação disponíel para smartphones;
- A aplicação pode ser acessada através de dispositivos móveis que utilizem o Android a partir da versão 2.2 ou em modelos terminais *i-mode* mais recentes;
- O usuário pode saber que uma mensagem foi entregue a seu destinatário através de uma notificação de SMS, enviada pelo servidor quando a mensagem tiver sido baixada e verificada.

A aplicação Disaster Kit foi projetada tendo como motivação o fato de que, durante os terremotos ocorridos no Japão em 11 de março de 2011, tornou-se bastante difícil realizar ligações devido ao alto congestionamento das linhas [2]. Ao mesmo tempo, o tráfico de comunicação por pacotes se apresentava sem grandes alterações. Desse modo, a aplicação aproveita a infraestrutura de rede disponível para proporcionar a seus usuários o envio e recebimento de mensagens de voz que em situações de emergência podem se tornar impossíveis de serem recebidas através das chamadas telefônicas.

O Disaster Kit se aproxima do presente trabalho quando propõe uma forma alternativa de estabelecer comunicação em situações de crise. Ambos os trabalhos buscam o uso do envio e recebimento de mensagens de voz através da rede, oferecendo uma alternativa à ligação telefônica convencional. Entretanto, a ferramenta proposta vai além da transferência de mensagens de voz. Com ela, os usuários poderão se beneficiar da sensibilidade ao contexto que poderá inferir que uma crise esteja em curso. O foco dos comandos de voz nesse caso é auxiliar as vítimas para que elas recebam auxílio ágil, rápido e adequado, uma vez que as informações obtidas serão utilizadas por sistemas de gestão de emergências.

B. FallAlarm: Um sistema de detecção de quedas e posicionamento baseado em smartphone

O FallAlarm é uma ferramenta que pode ser utilizada em *smartphones* para perceber quedas de pacientes e/ou pessoas que necessitam de cuidados especiais. Ela envia um sinal para um destinatário para que o mesmo tenha ciência do ocorrido e da respectiva localização da vítima [3].

O sistema utiliza o acelerômetro do smartphone para reconhecer que uma queda tenha ocorrido. O local onde o acidentado se encontra é descoberto através do sinal Wi-Fi no ambiente. Dessa forma, a aplicação não somente detecta a queda como também identifica onde ela ocorreu. Para reduzir o consumo de energia dos módulos Wi-Fi, o sistema dispara um método de posicionamento através de RSSI. Além disso, o sistema envia uma mensagem para um contato da vítima informando que a mesma sofreu um acidente.

O FallAlarm pode ser instalado em um smartphone que possua um sensor acelerômetro e um módulo Wi-Fi. Isso faz com que os usuários possam levar o sistema consigo para onde forem. O FallAlarm sempre permanece em execução coletando dados do acelerômetro. Após um intervalo de alguns segundos, os dados prospectados são extraídos para formar uma amostra. Tendo a amostra como informação de entrada, a aplicação pode identificar o tipo da atividade realizada. Se essa atividade é corriqueira (correr, andar, etc), o FallAlarm a desconsidera e reinicia o processo de amostragem. Caso contrário, o sistema coleta o RSSIs(Received Signal Strength Indiction) dos pontos de acesso próximos à vítima e obtem a sua localização através da posição do módulo. Após a fase de posicionamento, a aplicação envia uma mensagem indicando a localização do acidente para os números armazenados.

C. PerFallD: Um sistema pervasivo de detecção de quedas usado em telefones móveis

Nesse trabalho já se pode perceber claramente aspectos de computação ubíqua. Os autores fazem uso das princpais propriedades do smartphone para a criação de um aplicativo pervasivo de detecção de quedas. Eles justificam que o uso do aplicativo apresenta uma melhor aceitação para os idosos, já que é melhor ter um smartphone sendo levado consigo do que carregar um dispositivo extra para detectar possíveis quedas [28].

As técnicas de detecção de quedas propostas em aplicações podem ser classificadas em três grupos:detecção feita com base na aceleração, no tipo de movimento realizado ou no processamento de imagens. O PerfallD utiliza a verificação da aceleração para determinar a ocorrência de queda.

O aplicativo conta com um módulo de configuração onde é possível configurar uma lista de contatos em caso de emergência. Além disso, o PerfallD permite definir configuração específica para a detecção da queda, como por exemplo um algorítmo padrão de detecção.

O aplicativo permanece rodando em background colhendo informações constantemente. Se uma informação adquirida satisfaz a uma condição pré-determinada no algorítmo, um processo inicia a determinar se uma queda ocorreu. Se a queda não foi detectada, a aplicação volta ao estado de background. Caso contrário, o serviço envia um sinal que dispara um alarme e inicia um temporizador. Se o usuário não desligar o alarme manualmente dentro de um determinado período, o sistema automaticamente liga para os contatos da lista de emergência, de acordo com a prioridade configurada. O *smartphone* liga e envia mensagem recursivamente para 5 contatos da lista.

D. Aplicação sensível ao contexto para smartphone para detecção de eventos por socorristas

No referido trabalho, os autores trazem uma solução que possibilita a utilização de informações fornecidas por vítimas e observadores através das redes sociais como facebook, titter, instagram, etc [4]. Tais informações são utilizadas como forma de se detectar as características do evento que está ocorrendo, de modo que os socorristas possam planejar melhor sua atuação. Através da aquisição e classificação de dados provenientes das mídias sociais em tempo real, os centros de comando serão capazes de avaliar rapidamente as necessidades e situações de muitos cidadãos simultaneamente e reagir de forma adequada, com base na sua proximidade ao local do evento. Os autores acreditam que as mídias sociais podem oferecer dados confiáveis, cuja utilização aumentará a velocidade, eficiência e até mesmo o desempenho em tempos de resposta a situações emergenciais.

O sistema se baseia em dois pressupostos: a existência de um número suficiente de pessoas que possuem dispositivos móveis com acesso à internet e o fato de que tais cidadãos em situações de crise utilizam as redes sociais para postarem informações. Com o número cada vez mais frequente desses dispositivos no mercado, os autores acreditam que a maioria das pessoas vão possuir *smartphones*. Assim, as redes sociais

também tendem a serem cada vez mais utilizadas para informar emergências que estejam ocorrendo.

O sistema tem como base a PSSA, uma arquitetura colaborativa em núvem que permite integração de novos sensores e seus serviços, através do conceito de normalização de eventos.Cada serviço de ingestão é responsável por registrar um novo sensor que publica seus fluxos de dados a todos os serviços subscritos interessados, incluindo disseminação, aplicação e serviços de armazenamento. Tais serviços habilitados no PSSA são distribuídos em várias plataformas computacionais(*desktop*, servidor, *mobile*) de acordo com os requisitos da missão. No PSSA, o sistema possui um serviço chamado OpenLST, que pode ser utilizado pelo centro de comando para visualizar a área onde está ocorrendo a emergência. Esse serviço mostra em um mapa as informações provenientes dos sensores e das mídias sociais, permitindo colaboração e comunicação entre seus usuários.

Esses serviços subscrevem e publicam eventos espaço-temporais tendo como base as informações e notícias provenientes de redes sociais, de veículos de comunicação, ou até mesmo dos próprios socorristas através do uso dos *smartphones*.

Esse sistema possui algumas semelhanças com o presente trabalho. Dentre elas, destaca-se o fato de que ambos utilizam-se da ideia de fornecer informações para facilitar o planejamento e ação dos centros de comando. No caso da aplicação proposta, as informações serão fornecidas aos socorristas pelas próprias vítimas. Assim, os centros de comando poderão ter uma visão de dentro do ambiente da crise, o que pode facilitar ainda mais a sua atuação.

IV. VoiceRescuer

O VoiceRescuer consiste em um aplicativo *mobile* capaz de realizar a detecção automática de uma situação emergencial. Ao identificar tal situação, ele inicia comunicação com a vítima através de comandos de voz e envia um pedido de socorro ao RESCUER [6], informando a localização do evento, bem como as condições em que a vítima e o ambiente se encontram.

Como já foi mencionado, os *smartphones* são dispositivos que estão em todos os lugares para as quais os seus donos se movam. Eles estão presentes na maior parte do dia das pessoas e na grande maioria das vezes conectado à internet, seja através das redes *wireless* ou mesmo utilizando-se das coberturas 3G ou 4G disponibilizadas pelas operadoras. Nesse contexto, imagine uma situação onde alguém possa sofrer uma adversidade a partir de um desabamento, por exemplo. Esse tipo de evento é bastante comum, seja como consequência de abalos sísmicos ou fortes tempestades. Quando uma casa ou edifício sofre desabamento em parte ou na sua completude, aqueles que se encontram no interior de tais estruturas podem sofrer danos físicos. Muitas vezes, as pessoas são lançadas ao solo, permanecendo imobilizadas devido à presença de escombros sobre o corpo. Em outros casos, a vítima permanece caída no chão e não consegue se movimentar devido a ferimentos. Esses acidentes têm em comum a carência de uma ferramenta que possa perceber tais necessidades de ajuda e fornecê-la da forma adequada, aumentando assim a eficiência e a eficácia no resgate.

O VoiceRescuer foi proposto como uma solução propícia para ser usada nesse tipo de situação. O fato de se utilizar comandos de voz para interagir com o usuário, além de executar os demais processos de percepção do ambiente sem necessitar que o mesmo manipule o dispositivo, faz desse aplicativo uma ferramenta poderosa para facilitar o apoio durante tais emergências.

Para melhor compreensão do seu funcionamento, serão mostrados na próxima subseção as decisões de engenharia de software tomadas para o desenvolvimento do VoiceRescuer.

A. Requisitos Funcionais do VoiceRescuer

A Figura 2 retrata o diagrama de estados da aplicação. Ele mostra o conjunto de estados e transições que ocorrem durante o funcionamento da ferramenta. No diagrama, é possível observar que a partir do seu início, o primeiro estado a ser encontrado pela aplicação é o estado de *Monitoramento*. É nesse estado que a aplicação permanece até que uma queda seja identificada.

O VoiceRescuer pode detectar uma emergência, alterando o seu estado para o de *Confirmação*, onde o usuário é questionado através de comandos de voz com a seguinte pergunta:"Está precisando de ajuda?". Obtendo-se a resposta "Não", o sistema retorna para o estado de *Monitoramento*. Se a resposta for um "Sim"ou se o aplicativo não conseguir identificar uma resposta, o aplicativo colhe as informações identificadas até o momento, acrescidas da localização do dispositivo e inicia-se um processo paralelo de *Envio do pedido inicial de ajuda*. O pedido inicial de ajuda é bastante importante para que as equipes possam imediatamente começar a se preparar para atuação.

Em paralelo ao envio do pedido inicial de socorro, verificando-se que existem perguntas configuradas a serem realizadas, atinge-se o estado de *Execução do questionário*. O VoiceRescuer então utiliza-se de um *script* de perguntas pré-configurado de onde cada pergunta é lida e transformada em comando de voz. Para cada pergunta feita, o sistema aguarda a resposta do usuário.

Concluindo-se a execução do questionário, a aplicação inicia o processo de *Obtenção das imagens*, no qual o VoiceRescuer prospecta informações do ambiente para que possa melhor informar às equipes de socorristas sobre o acidente. Assim, ele utiliza-se das câmeras do dispositivo para obter as imagens do ambiente no intuito de perceber as condições em que o local se encontra, bem como as possíveis dificuldades de acesso apresentadas.

A fim de se complementar o pedido de ajuda com as perguntas e respostas efetuadas e com as imagens obtidas no último passo, o VoiceRescuer envia um pedido de atualização do *report* enviado para a emergência. Dessa forma, busca-se atualizar no RESCUER [6] toda a comunicação realizada junto ao usuário.

A seguir, serão detalhados os requisitos funcionais responsáveis por fazer com que o aplicativo permeie cada um dos estados supracitados.

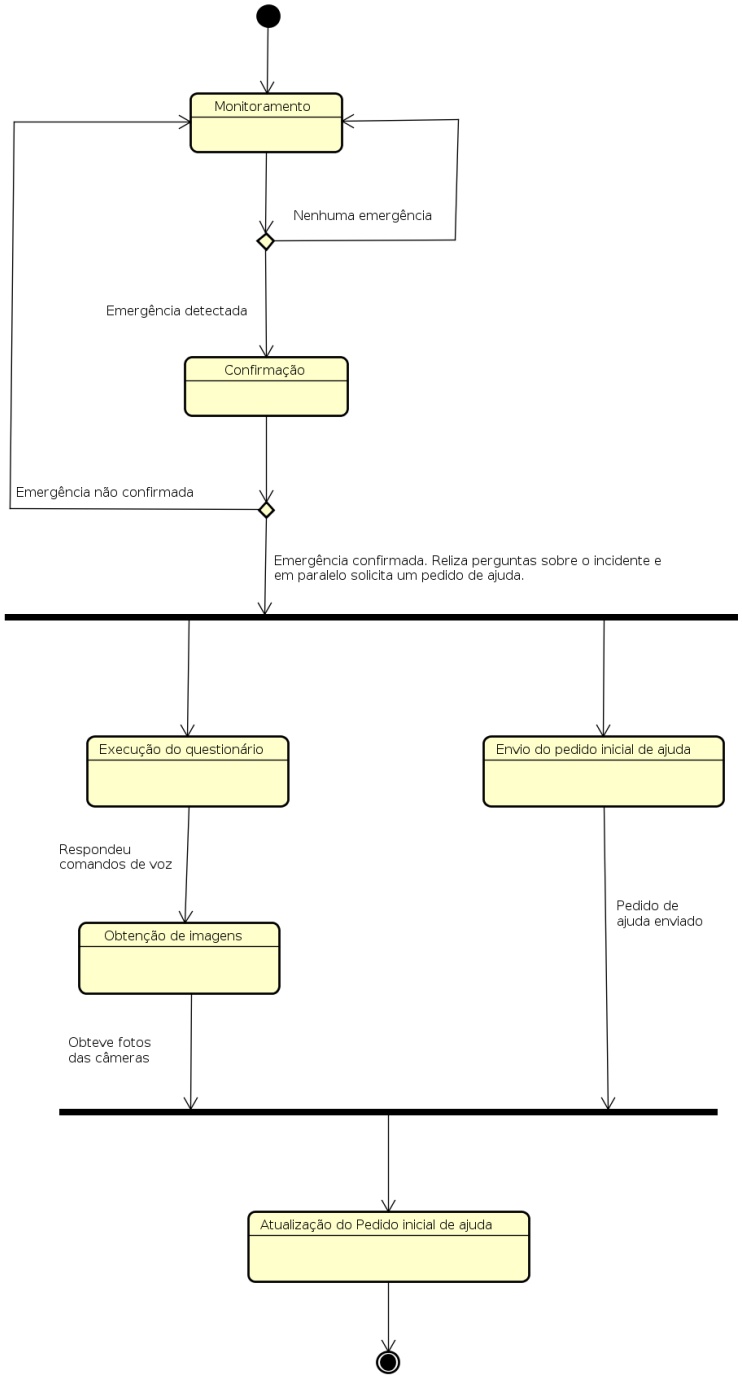

Figura 2: Diagrama de estados da aplicação.

1) Detecção de queda: Como foi relatado na seção II-B, os dispositivos *mobile* em geral possuem acelerômetros. Eles são utilizados para tentar se perceber qual tipo de movimento está sendo executado com o aparelho. O VoiceRescuer utiliza-se do acelerõmetro para identificar se o dispositivo sofreu uma queda e oferecer ajuda em caso de uma possível emergência.

Muitas das situações emergenciais que acontecem podem ter como consequência a queda. Seja em terremotos, fortes tempestades com ventanias, desabamentos ou mesmo acidentes em casa: as vítimas podem ser arremessadas ao chão. Desse modo, o acelerômetro foi escolhido como gatilho para disparar as funcionalidades do VoiceRescuer, pois ele é a forma mais abrangente de auferir emergências, já que grande parte dos incidentes tem como consequência o lançamento das vítimas ao chão.

Em II-C1, foram apresentadas as principais técnicas utilizadas atualmente para identificação de quedas através dos valores lidos pelos acelerômetros. Para o VoiceRescuer, a técnica utilizada é a acelerometria em três eixos, na qual subentende-se que o dispositivo possa ter caído através da análise da

variação dos valores nos três eixos do acelerômetro. Dessa forma, a aceleração é calculada pela seguinte fórmula:

$$|A_T| = \sqrt{A_x^2 + A_y^2 + A_z^2}$$

Logo, o cálculo da aceleração é dado pela raiz quadrada da soma dos quadrados dos valores obtidos nos eixos x, y e z. Um processo de queda é deduzido quando o valor da aceleração sofre uma variação brusca seguida de impacto [18]. A verificação dos valores do acelerômetro acontece de forma automática e em *background*. Assim que o aplicativo é instalado, um serviço de escuta desse sensor é inicializado, permanecendo assim até que uma nova variação repentina seja percebida. Nesse caso, tem-se o início do procedimento de confirmação do incidente através de comandos de voz.

2) Comunicação através de comandos de voz: A partir da detecção de uma possível queda, o VoiceRescuer emite um comando de voz onde efetua a seguinte pergunta para o usuário "Está precisando de ajuda?". Após emitir esse questionamento, a ferramenta aguarda durante 5 segundos para obter uma resposta. Caso ele responda "Não", o serviço retorna à escuta dos valores do acelerômetro de modo transparente. Caso seja identificada uma resposta "Sim", o sistema envia uma notificação ao RESCUER [6] formalizando um pedido de socorro. Caso nenhuma resposta seja identificada, o sistema se comporta semelhantemente à resposta "Sim".

Esse último comportamento foi concebido pensando nos casos em que a vítima sofre um acidente e permanece desacordada. Assim, mesmo impossibilitada de solicitar ajuda, o aplicativo emitirá o pedido de socorro e através das fotos retiradas das câmeras frontal e traseira, a equipe de socorristas poderá verificar as condições do ambiente o que pode ajudar a determinar se uma emergência está em curso.

Após realizar a solicitação inicial de ajuda, o aplicativo retoma a continuação do diálogo com a vítima, a fim de se obter maiores detalhes do incidente. Para cada pergunta realizada, o sistema deverá aguardar 5 segundos e caso uma resposta não seja identificada, será realizado o próximo questionamento disponível até que todas as perguntas tenham sido emitidas. De posse de todas as perguntas e respostas obtidas, a ferramenta enviará um *report* (relato) ao RESCUER agregando as novas informações ao pedido de socorro emitido inicialmente.

O processo de comunicação com o usuário se dá através do uso do comando de voz para prevenir o fato de a vítima estar impossibilitada de manusear o aparelho, por conta de ferimentos que a impeçam de movimentar os braços. Em muitas ocasiões, o acidentado também pode estar imobilizado por conta de entulhos sobre os membros, como ocorreu com uma vítima de terremotos no Haiti no ano de 2010 [5], a qual foi encontrada por uma equipe de resgate após uma semana em meio aos escrombos.

3) Obtenção da localização da ocorrência: A localização GPS serve para auxiliar às equipes de resgate nas buscas pelas vítimas. As fotos retiradas do local ajudam as equipes a visualizarem as condições em que se encontra o lugar onde ocorreu o incidente. Essas informações podem ser utilizadas pelo RESCUER [6] para apoiar os processos de gerenciamento de emergência, como por exemplo, o estabelecimento de rotas de resgate para as equipes de socorristas. Além disso, será possível ter uma noção do número de vítimas disponíveis no local, o que pode ser bastante benéfico para o planejamento do socorro.

4) Obtenção das imagens do ambiente da emergência: Como forma de possibilitar às equipes de resgate uma possível visualização das condições ambientais da emergência, o VoiceRescuer capta fotos tanto da câmera frontal quanto da câmera traseira. Através das imagens, as equipes de resgate podem verificar quais os riscos que o ambiente oferece para o resgate, auxiliando no estabelecimento de estratégias de resgate. As fotos também podem ajudar a verificar se existem outras vítimas próximas ao dispositivo, caso o mesmo tenha caído em uma posição em que consiga captar tal informação.

A seguir, serão explanados os aspectos arquiteturais e de implementação da solução proposta. Serão evidenciadas as principais decisões que nortearam a concepção do aplicativo, considerando suas necessidades de operação.

B. Aspectos Arquiteturais e de Implementação

A aplicação proposta foi projetada e desenvolvida tendo como foco o sistema operacional Android. Segundo [29], o Android é uma plataforma de desenvolvimento para aplicativos móveis para *smartphones* e possui um sistema operacional baseado em Linux. Além disso, ele contém um ambiente de desenvolvimento poderoso, inovador e flexível, onde os desenvolvedores podem utilizar a linguagem Java para construir as aplicações, beneficiando-se de todos os recursos disponíveis no sistema e no aparelho.

O Android surgiu a partir da junção das maiores companhias de celulares lideradas pela Google. Esse grupo é denominado de Open Handset Alliance (OHA) e tem em sua formação empresas como HTC, LG, Motorola, Samsung, Sony Ericsson, ASUS, Intel, Dell, dentre muitas outras. Em sua página [30], encontra-se uma descrição do que consiste essa aliança:"Hoje, existem 1,5 bilhão de aparelhos de televisão em uso em todo o mundo e 1 bilhão de pessoas tem acesso à internet. No entanto, quase 3 bilhões de pessoas têm um telefone celular, tornando o aparelho um dos produtos de consumo mais bem-sucedidos do mundo. Dessa forma, construir um aparelho celular superior melhoraria a vida de inúmeras pessoas em todo o mundo. A Open Handset Alliance é um grupo formado por empresas líderes em tecnologia móvel que compartilham essa visão para mudar a experiência móvel de todos os consumidores[...]".

Desse modo, o Android é o sistema operacional utilizado pela grande parte dos aparelhos das principais fabricantes de *smartphones*. Criar um aplicativo para essa plataforma, significa abranger uma vasta gama de dispositivos espalhados pelo mundo.

A Figura 3 mostra a organização arquitetural do Android [31]. Nela podem ser observados os seguintes componentes:

- ***Application Framework:*** É mais frequentemente utilizada pelos desenvolvedores de aplicação;
- ***Binder Inter-Process Communication (IPC):*** Mecanismo que permite à camada *Application Framework* realizar chamadas ao sistema de serviços do Android (*System Services*);

Figura 3: Arquitetura do Android.

- ***System Services:*** As funcionalidades disponibilizadas pela camada *Application Framework* se comunicam com a camada *System Services* para acessar o *hardware* subjacente. Os serviços são organizados em módulos que possuem componentes como *Window Manager*, *Search Service* ou *Notification Manager*;
- ***Hardware Abstraction Layer(HAL):*** É a camada de abstração do *hardware*. Ela define o padrão de interface que deve ser seguido pelos fornecedores de *hardware*, de modo a se permitir facilidade de integração entre tais componentes e o sistema operacional.

Os aplicativos do Android são desenvolvidos utilizando-se a linguagem JAVA. Nesse processo, as ferramentas Android SDK compilam o código em conjunto com todos os arquivos de dados e recursos, gerando um APK [32]. Nesse arquivo, está todo o conteúdo de um aplicativo desenvolvido para Android e é utilizado para realizar a instalação dos mesmos nos dispositivos.

Para melhor entender como a aplicação foi desenvolvida, é importante ter ciência dos tipos de componentes que são utilizados na construção de aplicativos Android. A arquitetura desse sistema, mais precisamente a camada *Application Framework*, fornece os seguintes tipos de componentes de aplicativos [32]:

- **Atividades:** As atividades representam uma tela única do aplicativo, ou seja, uma interface com o usuário. Por exemplo, um aplicativo de e-mail pode ter uma atividade correspondente a uma tela onde seja feita a listagem de e-mails. As atividades são independentes entre si e são implementadas através de subclasses da classe *Activity*;
- **Serviços:** Os serviços são executados em segundo plano para efetuar operações longas ou trabalhos para processos remotos. Não apresentam uma interface com o usuário. É implementado por uma subclasse da classe *Service*;
- **Provedores de conteúdo:** Os provedores de conteúdo gerenciam um conjunto de dados compartilhados no aplicativo. É o provedor de conteúdo que permite o acesso ou alteração ao conteúdo compartilhado;
- **Receptores de transmissão:** São componentes que respondem a anúncios de transmissão por todo o sistema. Como exemplo de transmissões genéricas, pode-se destacar o anúncio de que uma tela foi desligada ou que a bateria está baixa ou que uma tela foi capturada. Os próprios aplicativos também podem criar os seus anúncios de transmissão.

A figura 4 retrata uma visão geral da arquitetura concebida para o VoiceRescuer. Nela, é possível observar que a foi utilizada na aplicação o estilo arquitetural *Virtual Machine*, onde o software é concebido através de camadas. Cada camada oferece um conjunto de serviços para os programas ou componentes que residem na camada acima. Desse modo, a figura mostra as três camadas utilizadas pelo aplicativo: a camada de visão, a camada de controle e a camada de persistência.

A camada de visão é constituída pelas telas que são mostradas aos usuários. Como a aplicação possui pouca interação direta através de telas, essa camada é formada por apenas dois artefatos: o arquivo main_activity.xml, o qual representa a tela utilizada para efetuar a configuração inicial e o camera_activity.xml, utilizado para habilitar as câmeras para fotografar o ambiente. Esses artefatos são usados e também fazem uso dos serviços disponibilizados pela camada de controle.

A camada de controle, por sua vez, é formada pelos serviços e atividades responsáveis por controlar os fluxos das funcionalidades do sistema. Eles são utilizados pela camada de visão para realizar processamentos a partir das ações dos usuários, ou mesmo disparar novas interações com os usuários, como a percepção da queda ou execução de perguntas através de comandos de voz, captura de fotos do ambiente, etc. Nessa camada é possível observar a presença da atividade MainActivity, a qual interage com a camada de aplicação através do arquivo de tela mani_activity.xml para realizar o

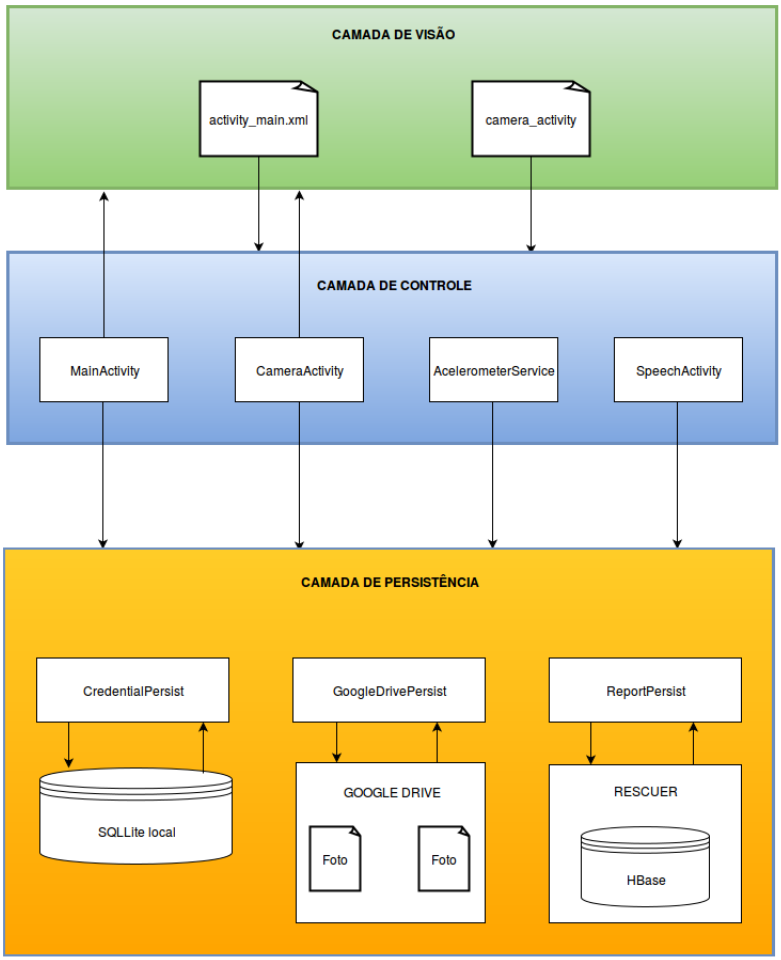

Figura 4: Representação em camadas da aplicação

fluxo de configuração da aplicação pelo usuário.

O AcelerometerService é o serviço responsável por monitorar o acelerômetro. Através dessa monitoração, o aplicativo consegue detectar a ocorrência de uma queda. Esse serviço é um ponto crucial da arquitetura do VoiceRescuer pois ele é o responsável por disparar a criação do serviço SpeechService, o qual, por sua vez, tem como papel iniciar o processo de confirmação da ocorrência de queda, utilizando comando de voz. O SpeechService realiza todo o processo de diálogo com o usuário, desde a confirmação da ocorrência até a execução do questionário, configurado na aplicação para ser executado caso uma emergência tenha sido confirmada. Para tais tarefas, ele utiliza-se dos seguintes componentes do Android:

- **TextToSpeech:** é o serviço do Google disponibilizado no Android que permite ao aplicativo converter texto em fala. Através desse serviço, o VoiceRescuer pode realizar as perguntas configuradas para as vítimas;
- **SpeechToText:** é o componente responsável por propiciar ao VoiceRescuer a possibilidade de captar as respostas faladas pelas vítmas e traduzi-las para texto.

A partir do SpeechService, invoca-se a atividade CameraActiviy, para que as imagens do ambiente possam ser capturadas por ambas as câmeras do dispositivo. Todo o controle de inicialização das câmeras, invocação da tela para habilitar o recebimento de imagens, bem como a própria captura, é executado pela CameraActivity.

As informações recebidas e tratadas na camada de controle precisam ser guardadas pelo aplicativo. Isso acontece em três esferas principais: quando a aplicação precisa manter o registro das chaves de autenticação necessárias para acessar o *Google Drive*; quando é necessário realizar o arquivamento das fotos obtidas do ambiente da ocorrência; quando é necessário realizar a criação e atualização dos *reports* para o RESCUER [6]. Todos essas necessidades foram disponibilizadas através artefatos implementados na camada de persistência. Como pode ser observado na figura 4, a camada de controle solicita aos serviços da camada de persistência para efetuarem o arquivamento e recuperação das informações.

No contexto de persistência da aplicação, o VoiceRescuer utiliza-se do SQLite, uma biblioteca que permite a criação e manutenção de uma base de dados interna, auto-contida e transacional [33]. Desse modo, a aplicação aproveita-se dessa biblioteca para criar e atualizar uma tabela onde são gravadas e consultadas as chaves recebidas e utilizadas para acessar o *Google Drive*. O objeto que possui o papel de orquestrar essa parte da persistência é o CredentialPersist.

Ainda na camada de persistência, existe o objeto GoogleDrivePersist responsável por realizar a conexão com o *Google*

Drive e enviar as fotos capturadas, como está representado em 4. O *Google Drive* disponibiliza um conjunto de apis *restfull*, que possibilitam o correto armazenamento das fotos nas respectivas contas dos usuários, para que posteriormente, suas urls possam ser compartilhadas no RESCUER [6].

Como pode ser observado em 4, na camada de persistência, existe também o componente ReportPersist, responsável por enviar os *reports* para o RESCUER [6]. O envio dos *reports* é feito através de mensagens disponibilizadas usando o RabbitMQ [34], um intermediador para envio e recebimento de mensagens. Assim, o objeto ReportPersist utiliza-se dessa tecnologia para intermediar a conexão a uma fila de mensagens, de modo que para o produtor, nesse caso o VoiceRescuer, a entrega das mensagens é feita de forma transparente. As mensagens, ou seja, os *reports* são enviados na forma de arquivos JSON contendo as informações obtidas do usuário e do ambiente.

A Figura 5 mostra o diagrama de componentes e conectores utilizados pela arquitetura do aplicativo. Os componentes AcelerometerService, SpeechService e CameraActivity estão conectados ao programa principal através de conectores do tipo *procedure call*, já que são inicializados pelo próprio programa, a depender dos estímulos recebidos pelo dispositivo. Além disso, o GoogleDrive é representado como um componente ao qual a aplicação se conecta por um conector do tipo http, ou seja, através de chamadas à API *restfull* desse componente, tendo como intermediador o protocolo http. Na figura 5 também está evidenciado o RESCUER [6] na forma de componente pois ele representa o sistema para onde os *reports* são enviados e utilizados no gerenciamento das emergências. O conector utilizado nessa conexão é dos tipos *data accsses e distributor*, pois ele representa o rabbitMQ [34] e suas propiedades de distribuição dos *reports* através da rede.

O diagrama de sequência mostrado na Figura 6 demonstra como os componentes de aplicativo do Android foram utilizados no desenvolvimento do trabalho proposto. Nele, pode ser observada a utilização da classe *AcelerometerService*, a qual é responsável pela verificação em segundo plano dos valores recebidos pelo acelerômetro nos eixos x, y, e z. Para tal, essa classe é uma subclasse de *Service* para que possa prover tal comportamento. A *AcelerometerService* também implementa a interface *SensorEventListener* e seus respectivos métodos *onSensorChanged* e *onAccuracyChanged*. Através do método *onSensorChanged* o aplicativo pode verificar os valores recebidos nos três eixos do acelerômetro e com isso determinar se houve ou não uma queda com base em uma aceleração obtida, conforme já explanado.

```
/**
* M todo onSensorChanged() definido na classe
    AcelerometerService
*/
@Override
    public void onSensorChanged(SensorEvent
        event) {
        float sensorX = -event.values[0];
        float sensorY = event.values[1];
        float sensorZ = event.values[2];

        Double aceleracaoCorrente = Math.sqrt(
            Math.pow(sensorX, 2)
                + Math.pow(sensorY, 2)
                + Math.pow(sensorZ, 2));

        if (aceleracaoCorrente <= 6.0) {

            aceleracaoInicial =
                aceleracaoCorrente;

        } else if (aceleracaoCorrente >= 12.5
            && aceleracaoInicial != null) {

            aceleracaoInicial = null;

            final Intent it = new Intent(this,
                SpeechService.class);
            it.addFlags(Intent.
                FLAG_ACTIVITY_NEW_TASK);
            startService(it);
            this.stopSelf();

        }
    }
```

Quando a queda é detectada, a classe *AcelerometerService* inicia um serviço chamado *SpeechService* que é responsável por tratar todo o processo de conversação com o usuário, desde a confirmação do incidente até a execução do questionário. Através do método *speak()*, o sistema realiza as perguntas à vítma. Após finalizar cada pergunta, o método *listenToSpeech()* habilita o reconhecimento de voz para capturar cada resposta obtida. Esses métodos vão se alternando até que todas as perguntas são realizadas.

```
/**
* M todo speak() definido na SpeechActivity
*/
public void speak() {

    UtteranceProgressListener
        mProgressListener = new
        UtteranceProgressListener() {
        @Override
        public void onDone(String utteranceId)
            {


            Handler uiHandler = new Handler(
                Looper.getMainLooper());

            Runnable runnable = new Runnable()
                {
                @Override
                public void run() {
                    listenToSpeech();
                }
            };
            uiHandler.post(runnable);
        }

        @Override
        public void onError(String utteranceId
            ) {
        }

        @Override
        public void onStart(String utteranceId
```

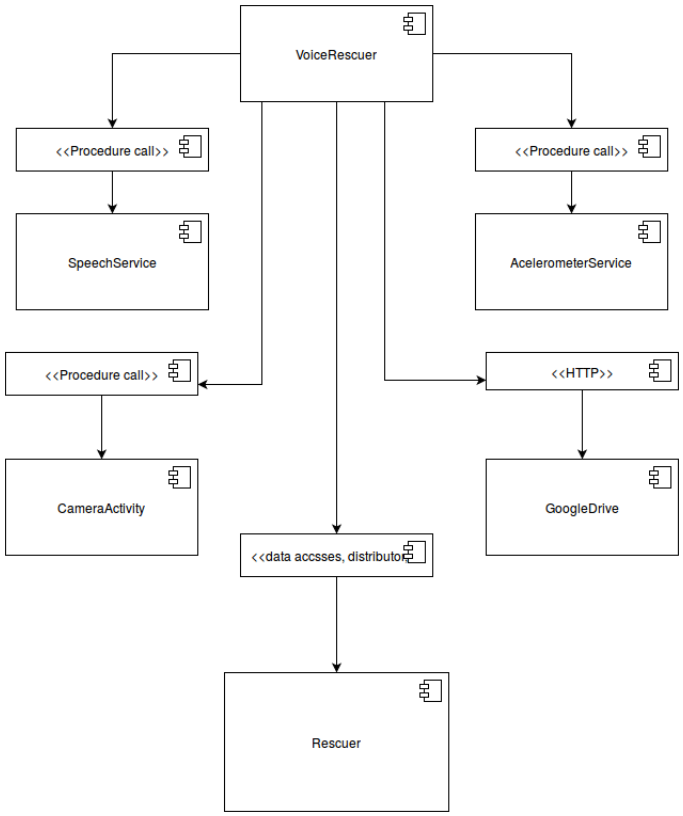

Figura 5: Diagrama de componentes e conectores da aplicação

```
            ) {

        }
    };
    int index = reportIncident.getChatMessages
        ().size();

    if(index < perguntas.length) {

        HashMap<String, String> map = new
            HashMap<String, String>();
        textToSpeech.speak(perguntas[index],
            TextToSpeech.QUEUE_FLUSH, map);
        textToSpeech.
            setOnUtteranceProgressListener(
            mProgressListener);
    }


}
```

Após a primeira pergunta ser executada (“Está precisando de ajuda?”), o aplicativo verifica se a resposta obtida foi “Sim” ou mesmo se não foi identificada uma resposta. Em ambos os casos, a partir do serviço *SpeechService*, é realizada uma chamada à *CameraActivity* para que seja tiradas as duas fotos, uma da câmera traseira e uma da câmera frontal. Ao ser finalizado esse processo, o sistema então executa o método *sendReport* para enviar ao RESCUER [6] o pedido inicial de socorro com base nas informações, fornecendo inclusive a localização GPS (obtida através do método *returnLocalization*). Feito isso, retorna-se ao ciclo de execução de perguntas e obtenção de respostas através dos métodos *speak()* e *listenToSpeech()*. Ao final desse processo, o método *sendReport* é invocado para atualizar as informações anteriormente enviadas para o RESCUER.

```
/**
* M todo listenToSpeech() definido na
    SpeechActivity
*/
private void listenToSpeech() {

    Intent listenIntent = new Intent(
        RecognizerIntent.
        ACTION_RECOGNIZE_SPEECH);

    listenIntent.putExtra(RecognizerIntent.
        EXTRA_SECURE,
            true);

    listenIntent.putExtra(RecognizerIntent.
        EXTRA_CALLING_PACKAGE, getClass().
        getPackage().getName());

    listenIntent.putExtra(RecognizerIntent.
        EXTRA_PROMPT, "Voc  precisa de ajuda?
        ");

    listenIntent.putExtra(RecognizerIntent.
        EXTRA_LANGUAGE_MODEL, RecognizerIntent
        .LANGUAGE_MODEL_FREE_FORM);


    listenIntent.putExtra(RecognizerIntent.
        EXTRA_MAX_RESULTS, 5);

    listenIntent.putExtra(RecognizerIntent.
```

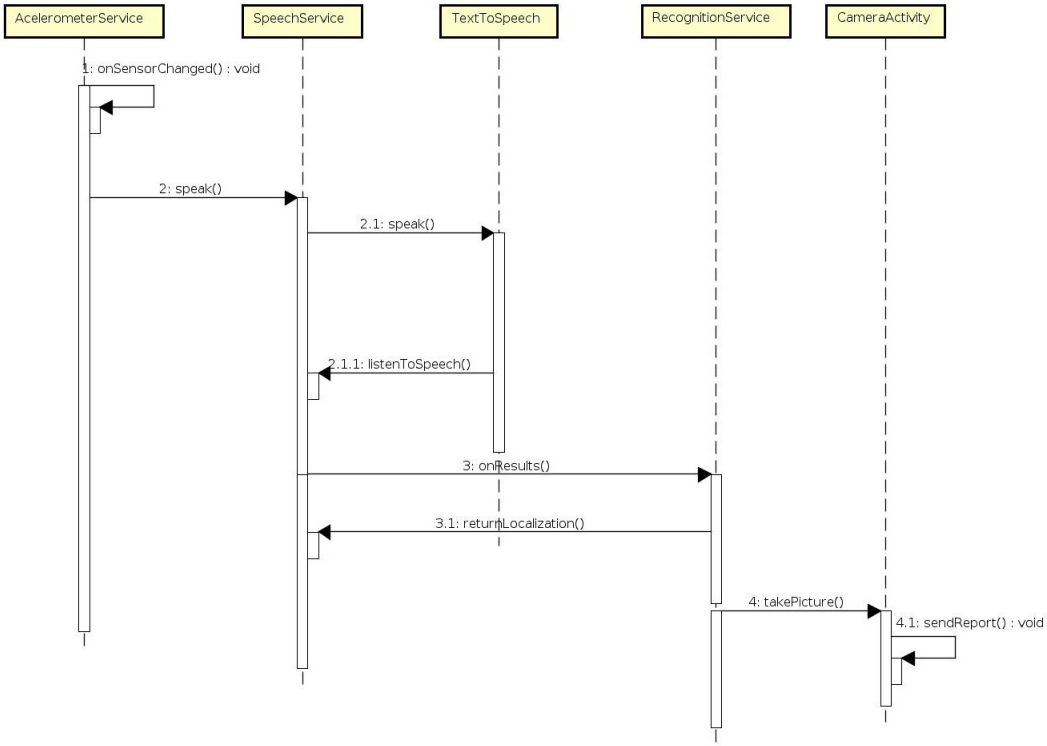

Figura 6: Diagrama de Sequência da aplicação

```
    EXTRA_SPEECH_INPUT_MINIMUM_LENGTH_MILLIS,
        5000);

    speechRecognizer.startListening(
        listenIntent);
}
```

É importante ressaltar que o VoiceRescuer foi projetado para usar o suporte nativo a idiomas do Android. É possível configurar tanto as mensagens mostradas nos componentes de tela, quanto as perguntas em diferentes idiomas. Dessa forma, busca-se manter uma internacionalização do aplicativo, de modo que ele possa ser utilizado em diferentes países. Essa configuração é feita através da alteração de arquivos XML onde são definidas os valores de constantes a serem utilizadas. Nos casos das perguntas, configuramos os valores em um elemento string-array de nome “questions“ onde cada item corresponde a uma pergunta. Esses valores são configurados em um arquivo chamado strings.xml, sendo que cada nacionalidade deve possuir sua respectiva pasta e o seu próprio arquivo.

Na próxima seção, será discutido todo o processo de avaliação da ferramenta, desde as decisões tomadas para sua elaboração até os resultados obtidos junto aos usuários.

V. Estudo de Caso

Para avaliar o VoiceRescuer enquanto ferramenta de apoio a situações emergenciais, ficou clara a necessidade de se executar simulações de situações emergências junto a um grupo de pessoas. A ideia é que cada participante possa compreender o contexto de uso do aplicativo. Além disso, buscou-se estimular os participantes a terem o contato com a aplicação e vislumbrarem qual impacto que a sua utilização poderá trazer.

A avaliação também tem como um de seus objetivos validar com os usuários a forma de interação na qual o aplicativo se baseia. Nesse processo, tentou-se extrair opniões a respeito de como a interação através da voz é utilizada no VoiceRescuer par dar suporte em situações críticas. Além disso, a responsividade da aplicação também constitui um ponto de observação, tendo em vista sua capacidade de responder aos estímulos vindos do ambiente.

Desse modo, buscou-se realizar experimentos junto a usuários munidos de dispositivos que possuem suporte a comandos de voz. Durante os processos de simulação, os usuários deixaram seus aparelhos caírem em superfícies amortecidas como colchões, sofás, etc. Em boa parte dos casos, os experimentos foram executados como se os participantes estivessem passando por desastres como desabamentos, acidentes, terremotos, furacões etc, fazendo com que as respostas para as perguntas realizadas pelo VoiceRescuer fossem baseadas nesses cenários fictícios.

A seguir serão levantados maiores detalhes de como a avaliação foi conduzida para atingir seus objetivos.

1) Protocolo: O processo de avaliação foi conduzido através da execução do VoiceRescuer em aparelhos Android nas versões 5 e 6 que possuem suporte a comandos de voz. Os usuários foram orientados a fazer o donwload da apk correspondente à versão do Android do seu respectivo aparelho, e efetuar a instalação.

Cada usuário foi orientado a realizar uma simulação de emergência que tem como consequência o dispositivo ser lançado. Foi sugerido que após a realização da simulação, um

questionário fosse preenchido a fim de se verificar a aplicação em seus aspectos principais.

Assim, cada simulação executada possui como meta avaliar os seguintes pontos:

- O aplicativo consegue ou não detectar as situações emergenciais quando elas ocorrem através de quedas e oferecer suporte à vítima;
- Em caso de percepção de uma possível ocorrência, o aplicativo consegue ou não tomar a ação pertinente, ou seja, fornecer ajuda através da emissão de uma pergunta falada;
- Após confirmação da emergência, o aplicativo consegue emitir perguntas na forma de comandos de voz e decodificar as respectivas respostas pronunciadas pelos usuários;
- A qualidade do processo de diálogo entre o aplicativo e o usuário, tendo em vista critérios como qualidade sonora dos questionamentos e a quantidade em que os mesmos são realizados;
- Prospecção das vantagens e desvantagens percebidas, bem como oportunidades de melhorias.

O questionário aplicado para avaliação das execuções foi desenvolvido visando extrair as informações necessárias para atingir os pontos relatados anteriormente. Dessa forma, o formulário se inicia com perguntas direcionadas à identificação do usuário. Em seguida, o questionário tenta prospectar informações sobre o contexto em que a simulação foi realizada, considerando fatos como o local em que o *smartphone* estava quando sofreu a queda (bolso, bolsa, mãos, mochila, etc) e a posição que o usuário se encontrava no momento do teste.

O formulário de avaliação também buscou verificar se o aplicativo estava responsivo no momento do teste. Para isso, o usuário informa se o aplicativo disparou a pergunta de confirmação da ocorrência ("Está precisando de ajuda?") e o que aconteceu quando o usuário forneceu uma resposta positiva. Em seguida, os usuários foram conduzidos a responderem questionamentos sobre a forma de interatividade através de comandos de voz, avaliando a clareza, a qualidade sonora e a quantidade das perguntas, o modo como as mesmas são efetuadas pelo aplicativo além do grau de facilidade de uso. Também buscou-se verificar se o aplicativo realizou todo o fluxo conforme o definido, desde a identificação da queda até o envio das imagens e emissão da mensagem final, a qual encerra o processo de interação ("Pedido de ajuda foi enviado com sucesso.O resgate já está a caminho.").

Finalmente, o questionário se propõe a verificar o entendimento da aplicação como um todo no que tange às vantagens e desvantagens obtidas com o seu uso. A avaliação dessas características foi proposta para validar o entendimento do usuário no propósito da ferramenta e se ela consegue desempenhar o papel que lhe foi atribuído.

Na próxima seção, serão analisados os resultados obtidos por meio da realização das simulações e aplicação do questionário para os envolvidos.

2) Resultados: As simulações foram realizadas por dezesseis usuários de *smartphones*. Alguns usuários possuem aparelhos de mesmo modelo e por isso, foram identificadas dez modelos utilizados nas simulações. A Tabela I mostra essa distribuição dos dispositivos. As versões do Android utilizadas foram a 5 (5.0, 5.0.1 e 5.1) e a 6 (6.0 e a 6.0.1). Dos dispositivos utilizados, apenas um estava configurado com o idioma inglês e os demais estavam utilizando o português.

Modelos	**Quantidade**
Moto G 1	2
Moto G 2	2
Moto G 3	2
Moto G3 TURBO	2
ASUS ZenFone 5	1
Moto X 1ª Geração	1
Moto X 2ª Geração	2
Moto X play	1
Samsung Galaxy S5	1
Samsung Note 3	1
LG D337	1

Tabela I: Distribuição dos dispositivos por modelo.

Foram feitas simulações de situações emergenciais onde os usuários realizaram quedas dos aparelhos e em 100% dos casos o dispositivo identificou que uma queda ocorreu e disparou a mensagem de voz "Está precisando de ajuda?". Em todos esses casos foram pronunciadas a resposta "Sim" e o VoiceRescuer conseguiu identificá-la e consequentemente continuar o processo de realização de perguntas e identificação das respectivas respostas. Em todos os casos de simulação, os usuários confirmaram que o aplicativo conseguiu finalizar o fluxo de perguntas e fotografar o ambiente utilizando as câmeras frontal e traseira.

Em relação à avaliação quanto a forma de interação através de comandos de voz, os usuários permaneceram divididos entre o critério 'Ótima' e 'Boa'. Cada uma dessas opções foi escolhida por 50% dos participantes.

No que diz respeito à qualidade sonora dos comandos de voz executados pelo VoiceRescuer, 56,3% dos participantes responderam que a considera 'Ótima'. Além disso, 37,5% dos usuários responderam que consideram a qualidade sonora alcançada como 'Boa'. Consideraram a qualidade sonora 'Ruim' 6,3% dos entrevistados.

No que se refere à clareza das perguntas efetuadas, em 68,8% dos casos ela foi considerada 'Ótima'. Os demais 31,3% a consideraram 'Boa'. Apenas um dos usuários justificou a escolha da opção 'Boa' informando que a clareza das perguntas Ìepende do volume que estiver o celular.

Em relação à facilidade de uso, a aplicação foi avaliada como 'Bom' em 50% dos casos. Outros 43,8% avaliaram esse quesito como 'Ótimo'. Os 6,3% restantes avaliaram a facilidade de uso como ruim. O principal argumento para as avaliações que optaram pela opção 'Bom' está relacionado ao fato do aplicativo necessitar da autenticação no Google Drive para criação da pasta onde serão compartilhadas as fotos obtidas das câmeras do dispositivo. Pelo fato de necessitarem colar na tela inicial do aplicativo o código de autenticação, contribuiu para essa distribuição na escolha das opções.

De um modo geral, no que tange aos benefícios adquiridos com o uso da ferramenta, 62,5% dos participantes a consideraram 'Muito Benéfica'. Além disso, 37,5% dos participantes consideraram a aplicação 'Benéfica'. Ao analisar os comentários atribuídos como justificativa a essas respostas, observa-se que os usuários em geral conseguiram perceber os

benefícios oferecidos pelo VoiceRescuer. Esses comentários recebidos foram bastante positivos, destacando o aplicativo como 'Excelente', 'Prático e eficiente' além de considerações em relação a os benefícios atingidos, como pode ser comprovado no comentário 'A solução tem uma excelente proposta e pode ser muito útil no dia-a-dia, pois traz agilidade no atendimento de vítimas de acidentes'.

No questionário aplicado foi solicitado que os usuários discorressem sobre as desvantagens de ter uma aplicação como o VoiceRescuer instalada no celular. De um modo geral, dois pontos foram levantados: o primeiro, relacionado ao risco de o aplicativo disparar uma mensagem de voz durante uma situação de assalto, na qual haja luta corporal, provocando uma situação de 'descontrole dos meliantes'. Em alguns casos, foi apontado o fato da aplicação poder ser ativada em movimentos bruscos que não representem uma queda consequente de uma emergência. Entretanto, em um dos comentários é apontado o fato dessa situação ser resolvida facilmente, já que nesse caso bastaria o usuário responder 'Não' para a pergunta de confirmação realizada quando a queda é percebida.

Quando o questionário solicitou que os usuários discorressem sobre as vantagens de se ter uma aplicação como o VoiceRescuer no celular, foram destacados: a grande utilidade da aplicação para indivíduos que estejam sozinhos e que precisem solicitar ajuda em uma emergência; a relevância da aplicação para proporcionar um resgate mais rápido e seguro; a possibilidade de se comunicar mesmo estando longe do aparelho e/ou sem conseguir se mover; dentre outras. Tudo isso mostra que os usuários compreenderam o aplicativo como vantajoso e de grande importância. Além disso, ficou claro que a aplicação conseguiu cumprir o seu propósito principal, estimulando os usuários a aderirem ao seu uso.

Como sugestões de melhorias, foram apontadas o desenvolvimento do aplicativo em outras plataformas como IOS e WindowsPhone. Além disso, foram feitas sugestões para a tela inicial de configuração da aplicação, principalmente relacionada à necessidade atual de informar o código de autenticação retornado pelo Google Drive. Foi sugerido também disponibilizar versões para outros dispositivos como *smartwatches*, por exemplo.

3) Riscos à validade: Em um processo de avaliação de uma ferramenta como o VoiceRescuer, voltada para apoiar situações emergenciais, existe o risco ligado ao fato de se estar trabalhando com situações de simulação. Mesmo que se reproduza as condições necessárias para que a aplicação possa realizar seu trabalho, dificilmente se consegue obter comportamento igual a uma emergência real.

Dentre os fatores que contribuem para a disparidade entre os cenários real e fictício, está o componente emocional. Em uma ocorrência real, os usuários podem se atrapalhar por conta das emoções afloradas, o que pode prejudicar a forma como a ferramenta possa perceber o contexto e até mesmo identificar as respostas.

Durante as simulações realizadas, não foi possível verificar, por exemplo, qual a intensidade máxima de impacto que os dispositivos conseguem atingir e como a aplicação se comportaria nesse cenário. Em outras palavras, como os celulares utilizados eram de propriedade dos usuários, não foram realizados testes para verificar o quanto de danos físicos tais dispositivos conseguem suportar e permanecerem funcionando e rodando a aplicação.

VI. Conclusão

Considerando as informações adquiridas nas simulações executadas pode-se entender o VoiceRescuer como uma ferramenta capaz de oferecer apoio a vítimas em situações emergenciais, cuja integração ao RESCUER possibilita o alcance de um resgate mais rápido e eficiente. A aplicação traz consigo uma gama de vantagens que pôde ser percebida diretamente pelos usuários.

Ao longo de toda a concepção do VoiceRescuer, algumas dificuldades foram encontradas. Dentre elas, podem ser destacadas aquelas ligadas ao desenvolvimento propriamente dito, como por exemplo, problemas encontrados para programar o aplicativo para fazer o controle adequado das câmeras, conforme a aplicação exigia. Ainda em relação ao desenvolvimento, foram encontrados alguns desafios ao colocar o aplicativo para disparar o comandos de voz e a captura das imagens quando o *smartphone* se encontra bloqueado.

Como a maioria das aplicações em suas primeiras versões, o VoiceRescuer ainda precisa sofrer algumas melhorias. Dentre elas, tem destaque a necessidade de se buscar uma alternativa que possibilite realizar o armazenamento das fotos sem que o usuário necessite realizar uma configuração prévia, como hoje ocorre no caso da integração com o Google Drive.

Notou-se também a necessidade de ampliar as possibilidades de uso da ferramenta. A primeira medida nesse sentido, é desenvolver versões compatíveis com outros sistemas operacionais para dispositivos móveis e mesmo para versões anteriores à versão 5 do Android. Além disso, pode ser realizado um levantamentos dos dispositivos pervasivos como os relógios inteligentes e se verificar a possibilidade de construir versões da aplicação que possam rodar nesses dispositivos.

Com o aperfeiçoamento dos dispositivos móveis, bem como com a incorporação de novos sensores, novas versões do VoiceRescuer surgirão, contendo novas formas de identificação de emergências. Por exemplo, dispositivos munidos de sensores de temperatura vão exigir do VoiceRescuer que um novo gatilho para o disparo das mensagens de voz seja incorporado para permitir dar apoio a uma outra modalidade de ocorrências.

Espera-se que em um futuro bem próximo o VoiceRescuer possa fazer parte do cotidiano das pessoas proporcionando-lhes maior segurança e acesso rápido a ajuda quando ela se apresenta extremamente necessária.

Referências

[1] V. A. de Angelo.

[2] K. Inoue, S. Kichimi, M. Sekine, and M. Kobayashi, "Disaster voice messaging service," *NTT DoCoMo Technical Journal*, vol. 14, no. 2, 2012.

[3] Z. Zhao, Y. Chen, S. Wang, and Z. Chen, "Fallalarm: smart phone based fall detecting and positioning system," *Procedia Computer Science*, vol. 10, pp. 617–624, 2012.

[4] S. K. Boddhu, R. P. Dave, M. McCartney, J. A. West, and R. L. Williams, "Context-aware event detection smartphone application for first responders," in *SPIE Defense, Security, and Sensing*. International Society for Optics and Photonics, 2013, pp. 874 213–874 213.

[5] UOL. (2010) Mulher é encontrada viva 1 semana após o terremoto no haiti. [Online]. Available: http://noticias.terra.com.br/mundo/america-latina/mulher-e-encontrada-viva-1-semana-apos-o-terremoto-no-haiti,4fa96355ccaea310VgnCLD200000bbcceb0aRCRD.html

[6] W. Knight, "Rescuer," http://http://www.rescuer-project.org/, 2014 (acessado Novembro 5, 2014).

[7] M. Weiser, "The computer for the 21st century," *Scientific american*, vol. 265, no. 3, pp. 94–104, 1991.

[8] ——, "Some computer science issues in ubiquitous computing," *Communications of the ACM*, vol. 36, no. 7, pp. 75–84, 1993.

[9] K. Lyytinen and Y. Yoo, "Ubiquitous computing," *Communications of the ACM*, vol. 45, no. 12, pp. 63–96, 2002.

[10] M. Weiser, R. Gold, and J. S. Brown, "The origins of ubiquitous computing research at parc in the late 1980s," *IBM systems journal*, vol. 38, no. 4, pp. 693–696, 1999.

[11] S. Elrod, R. Bruce, R. Gold, D. Goldberg, F. Halasz, W. Janssen, D. Lee, K. McCall, E. Pedersen, K. Pier *et al.*, "Liveboard: a large interactive display supporting group meetings, presentations, and remote collaboration," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 1992, pp. 599–607.

[12] G. Banavar and A. Bernstein, "Software infrastructure and design challenges for ubiquitous computing applications," *Communications of the ACM*, vol. 45, no. 12, pp. 92–96, 2002.

[13] R. Ballagas, J. Borchers, M. Rohs, and J. G. Sheridan, "The smart phone: a ubiquitous input device," *Pervasive Computing, IEEE*, vol. 5, no. 1, pp. 70–77, 2006.

[14] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *Communications Magazine, IEEE*, vol. 48, no. 9, pp. 140–150, 2010.

[15] M. Mubashir, L. Shao, and L. Seed, "A survey on fall detection: Principles and approaches," *Neurocomputing*, vol. 100, pp. 144–152, 2013.

[16] J. M. Kang, T. Yoo, and H. C. Kim, "A wrist-worn integrated health monitoring instrument with a tele-reporting device for telemedicine and telecare," *Instrumentation and Measurement, IEEE Transactions on*, vol. 55, no. 5, pp. 1655–1661, 2006.

[17] N. Noury, T. Hervé, V. Rialle, G. Virone, E. Mercier, G. Morey, A. Moro, and T. Porcheron, "Monitoring behavior in home using a smart fall sensor and position sensors," in *Microtechnologies in Medicine and Biology, 1st Annual International, Conference On. 2000*. IEEE, 2000, pp. 607–610.

[18] C.-F. Lai, S.-Y. Chang, H.-C. Chao, and Y.-M. Huang, "Detection of cognitive injured body region using multiple triaxial accelerometers for elderly falling," *Sensors Journal, IEEE*, vol. 11, no. 3, pp. 763–770, 2011.

[19] A. M. Tabar, A. Keshavarz, and H. Aghajan, "Smart home care network using sensor fusion and distributed vision-based reasoning," in *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*. ACM, 2006, pp. 145–154.

[20] M. Alwan, P. J. Rajendran, S. Kell, D. Mack, S. Dalal, M. Wolfe, and R. Felder, "A smart and passive floor-vibration based fall detector for elderly," in *Information and Communication Technologies, 2006. ICTTA'06. 2nd*, vol. 1. IEEE, 2006, pp. 1003–1007.

[21] J. Glass, "Challenges for spoken dialogue systems," in *Proceedings of the 1999 IEEE ASRU Workshop*, 1999.

[22] P. Lison and R. Meena, "Spoken dialogue systems: the new frontier in human-computer interaction," *XRDS: Crossroads, The ACM Magazine for Students*, vol. 21, no. 1, pp. 46–51, 2014.

[23] R. Vishnupriya and T. Devi, "Speech recognition tools for mobile phone-a comparative study," in *Intelligent Computing Applications (ICICA), 2014 International Conference on*. IEEE, 2014, pp. 426–430.

[24] T. Geller, "Talking to machines," *Communications of the ACM*, vol. 55, no. 4, pp. 14–16, 2012.

[25] U. DHA, "Internationally agreed glossary of basic terms related to disaster management," *UN DHA (United Nations Department of Humanitarian Affairs), Geneva*, 1992.

[26] W. J. Petak, "Emergency management: A challenge for public administration," *Public Administration Review*, pp. 3–7, 1985.

[27] M.-C. Yuen, I. King, and K.-S. Leung, "A survey of crowdsourcing systems," in *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*. IEEE, 2011, pp. 766–773.

[28] J. Dai, X. Bai, Z. Yang, Z. Shen, and D. Xuan, "Perfalld: A pervasive fall detection system using mobile phones," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*. IEEE, 2010, pp. 292–297.

[29] R. R. Lecheta, *Google Android-3ª Edição: Aprenda a criar aplicações para dispositivos móveis com o Android SDK*. Novatec Editora, 2013.

[30] Android. (2016) Open handset alliance. [Online]. Available: http://www.openhandsetalliance.com/oha_overview.html

[31] ——. (2016) Android interfaces and architecture. [Online]. Available: https://source.android.com/devices/

[32] A. Developers. (2016) Fundamentos de aplicativos. [Online]. Available: http://developer.android.com/intl/pt-br/guide/components/fundamentals.html

[33] D. R. Hipp. (2016) Sqlite. [Online]. Available: https://www.sqlite.org/

[34] Pivotal. (2016) Rabbitmq. [Online]. Available: https://www.rabbitmq.com/features.html