Um Survey sobre Processamento de Histórico de Contextos para Inferência de Situações em Sistemas Distribuídos

Pedro A. Tavares, UCPEL
E-mail: pedro.tavares@sou.ucpel.edu.br

Resumo—Sistemas cientes de contexto são aqueles capazes de adaptarem seu comportamento de maneira dinâmica com base nos dados coletados no ambiente. O tempo é um aspecto necessário para modelar a variabilidade contextual, sendo muito importante manter dados de contextos passados para aprimorar a qualidade das inferências de situação no presente. Estes históricos de contexto podem ser armazenados e processados de muitas maneiras pelos sistemas computacionais atuais, desde através de cálculos matemáticos e estatísticos até por meio de algoritmos de aprendizado de máquina. Esse survey apresenta diversos estudos de caso que utilizam o processamento de histórico de contexto como ferramenta nos seus sistemas de inferência de situação, em aplicações de áreas totalmente distintas.

Palavras-Chave—context awareness, context history, context prediction, internet of things (IoT), distributed systems.

1 Introdução

Avanços recentes nas áreas de computação móvel e pervasiva permitiram o advento de um novo modelo de comunicação conhecido como *Internet of Things (IoT)*, que consiste na presença distribuída e conectada de uma grande variedade de objetos e sensores que se comunicam com o objetivo de receber e/ou fornecer informações e interagirem entre si [1].

Essa característica de onipresença da informática no cotidiano das pessoas, trouxe a possibilidade de implementar sistemas computacionais mais inteligentes, capazes de perceber o ambiente e contextualizar informações. Denomina-se esta percepção de *Ciência de Contexto*. Quando os sistemas computacionais passam a armazenar esses dados de contextos já percebidos, mantendo registros históricos, tornam-se capazes de fazer inferências muito mais precisas acerca de determinadas situações.

Este *survey* aborda vários sistemas que exploram a ideia de armazenar e processar históricos de contextos para auxiliar o usuário na tomada de decisões. Nas seções 2 e 3 são apresentados os conceitos de ciência e histórico de contexto. Na seção 4 é feita um estudo de cinco casos que utilizam o processamento de histórico de contexto para inferência de situações. Os sistemas escolhidos para análise são projetos atuais (pós-2018) e possuem aplicabilidades diversas. Ao final do artigo, na secção 5, são apresentadas as conclusões da pesquisa bem como sugestões para ampliá-la.

2 Ciência de Contexto

Segundo [2], contexto em sistemas de computação é o entendimento do ambiente, incluindo parâmetros da própria aplicação, representados através de informações contextuais das entidades em intervalos de espaço e tempo, possibilitando a interpretação de significado e inferências em ações futuras. Outra definição, dada por [3] diz que contexto é qualquer informação que caracteriza a situação de uma entidade, sendo que uma entidade pode ser uma pessoa, um lugar ou um objeto considerados relevantes para a interação entre um usuário e uma aplicação, incluindo o próprio usuário e a aplicação. O contexto é tipicamente a localização, a identidade e o estado das pessoas, grupos ou objetos físicos e computacionais.

Considerando todas as definições pode-se dizer que contexto é a situação em que uma ação ocorre que, quando qualificada e quantificada,

complementa e dá sentido à ação ajudando na sua compreensão. Em computação pervasiva, onde há mobilidade dos nós e as condições de processamento do sistema podem variar, o contexto é um elemento muito importante, tendo grande influência nos quesitos de processamento e da comunicação.

3 Histórico de Contexto

Há sistemas onde a simples análise do contexto atual é insuficiente para fazer inferências precisas sobre situações. Existem casos onde o tempo é um fator determinante, sendo necessário analisar informações passadas para calcular tendências e variações de contexto. Uma leitura única de um termômetro digital, por exemplo, fornece a temperatura instantânea naquele exato instante enquanto que, se o sistema tiver armazenado históricos de leituras anteriores será capaz de calcular a variação da temperatura, que pode ser uma informação muito mais útil para determinar a situação atual. E, caso esse histórico for longo, pode-se calcular variações de temperatura, ao longo do dia, dos meses, dos anos. E ainda relacionar esses contextos com localizações para poder tratar de maneiras diferentes essas informações relacionando-as a diferentes temperaturas-ambiente de diferentes lugares em diferentes épocas do ano. As possibilidades são muito maiores quando se têm acesso aos contextos passados de determinada entidade.

Considerando a grande heterogeneidade de aplicações voltadas para a IoT (*Internet of Things*), há inúmeros casos onde se faz necessário armazenar e tratar os contextos no domínio do tempo. A este conjunto de contextos é dado o nome de Histórico de Contexto.

De acordo com [4], a coleção de contextos passados e as ações do usuário nesses contextos são conhecidas como histórico de contexto. Existem inúmeras possibilidades de melhoria dos serviços oferecidos com essa abordagem. Em [5] os autores argumentam que o armazenamento e o uso do histórico de contexto são críticos, uma vez que os dados históricos permitem comportamentos, preferências, padrões, tendências, necessidades e muito mais a serem entendidos. Em [6] é argumentado que o papel da previsão em sistemas sensíveis ao contexto é frequentemente visto como um facilitador de execução de serviços e aplicativos. Para expandir essa visão, são necessárias novas maneiras de modificar iterações entre usuários e aplicativos. Uma estratégia para isso é o uso da proatividade em sistemas, com base em históricos de contextos armazenados. Isto está de acordo com a visão de [7], que aponta à necessidade de pró-atividade para tornar a computação onipresente mais eficaz.

4 Estudo de Caso: Sistemas Sensíveis ao Contexto

Há inúmeras maneiras de se armazenar e processar históricos de contexto. Nesta seção serão mostrados diversos estudos atuais que utilizam esta técnica no desenvolvimento de seus sistemas para realizar inferências de situações relativas às suas aplicações específicas. Além do uso de histórico de contexto foram selecionados para esta pesquisa apenas artigos recentes, publicados a partir de 2018. A relevância e a diversidade de aplicações foi outro ponto considerado para a inclusão dos projetos neste *survey*.

4.1 Sistema de Detecção de Estresse Mental através da Variabilidade da Frequência Cardíaca

Várias pesquisas vêm sendo feitas no intuito de detectar situações de estresse e relacioná-las às atividades cotidianas. Essa possibilidade permitirá o desenvolvimento de aplicações para evitar e minimizar a ocorrência desses situações, melhorando o bem-estar dos indivíduos e evitando uma série de complicações de saúde física e psíquica causadas por estresse mental.

Neste sentido o projeto proposto por [8], utiliza o processamento de séries históricas de contexto juntamente com algoritmos de aprendizado de máquina para inferir quais são as situações cotidianas que geram estresse mental aos indivíduos.

O Sistema Nervoso Autônomo (SNA) do ser humano, dentre outras atividades é também responsável por coordenar a atividade cardiovascular, digestiva e respiratória. Além disso controla as respostas involuntárias geradas em reposta a estímulos externos. Quando em situação de estresse mental, o corpo gera respostas fisiológicas como o aumento da frequência cardíaca (FC) e variação

Característica	Unid.	Fórmula
Domínio de Tempo		
Average of RR interval (AVRR)	ms	$\overline{RR} = \frac{1}{N}\sum_{j=1}^{N} RR_j$
Standard Deviation of RR intervals (SDRR)	ms	$\sqrt{\frac{1}{N-1}\sum_{j=1}^{N}\left(RR_j - \overline{RR}\right)^2}$
Standard Deviation of the Average RR intervals (SDARR)	ms	$\sqrt{\frac{1}{N-1}\sum_{j=1}^{N}\left(RR_{1_j} - \overline{RR_1}\right)^2}$
Root Mean Square of RR interval Diff. (RMSSD)	ms	$\sqrt{\frac{1}{N-1}\sum_{j=1}^{N}\left(RR_{j+1} - RR_j\right)^2}$
% of RR intervals that differ by more than 50 ms (pRR50)	%	$\frac{NN50}{N-1} \times 100\%$
Domínio de Frequência		
Low Frequency (LF)	ms^2	Potência espectral entre 0,04 e 0,15 Hz
High Frequency (HF)	ms^2	Potência espectral entre 0,15 e 0,4 Hz
Ratio LF/HF	ms^2	Razão entre LF e HF

Figura 1: Características da frequência cardíaca no domínio do tempo e da frequência (SDE)

da frequência cardíaca (VFC). O sistema proposto captura, através de um sensor de monitoramento cardíaco (ECG) conectado a um celular, cada batimento do coração, enviando esses dados para um servidor localizado na nuvem. Para cada dado, é salvo também o contexto em que o usuário se encontra. Fazem parte deste contexto:

- *Identidade*: Identificador único;
- *Fisiologia*: Leitura dos batimentos cardíacos;
- *Atividade*: Tipo de movimentação do usuário, obtida através dos sensores inerciais do *smartphone* e do serviço *Google Location Services*;
- *Localização*: Localização geo-referenciada do usuário, obtida através do GPS do *smartphone*;
- *DataHora*: *Timestamp* da captura do contexto;
- *Avaliação*: Avaliação qualitativa do nível de estresse de 0 a 4 feita pelo usuário;
- *Classificação*: Resultado do algoritmo classificador para inferir se o usuário está ou não sob situação de estresse.

O sistema salva o contexto para cada *heartbeat* e implementou um classificador usando algoritmos de *machine learning* que, para melhorar sua precisão, usou como características de entrada diversas variáveis calculadas através de séries temporais, como pode ser visto na figura 1.

O sistema faz uma segmentação das séries temporais considerando a norma de tempo curto com 5 minutos de duração e aplica uma série de filtros para remoção de ruídos. Só então, esses dados pré-processados são enviados para o algoritmo classificador *SVM (Support Vector Machine)*. Outros tipos de técnica foram testados, mas o SVM foi o que resultou em uma melhor acurácia.

Além de possibilitar a remoção de ruído, os dados sensíveis ao contexto coletados tornaram possível relacionar o estado de estresse computado com a localização em que foram mensurados, conforme pode ser percebido no gráfico da figura 2.

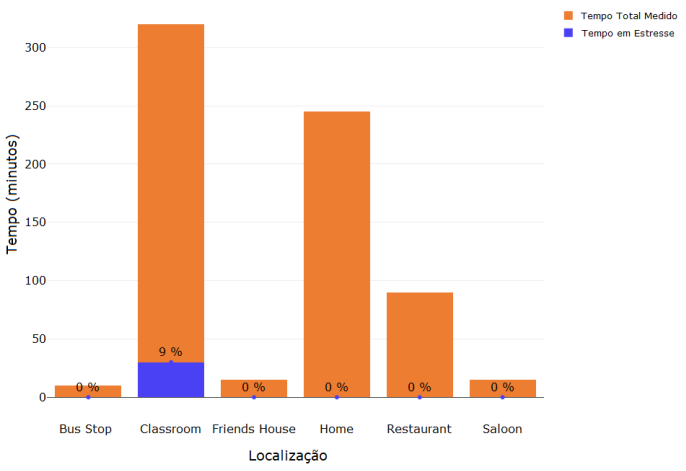

Figura 2: Relação entre nível de estresse e Localização

Este trabalho apresentou a coleta de dados sen-

síveis ao contexto com o intuito da classificação do estresse mental. Estes dados possibilitaram identificar e remover ruído da frequência cardíaca, tais como a movimentação e as atividades físicas, bem como mostrar uma relação dos níveis de estresse e a localização em que foram mensurados.

4.2 CHSPAM

Entre os vários tipos de análise que podem ser realizados utilizando histórico de contexto, uma das mais importantes é o reconhecimento padrões. Ele fornece ferramentas que podem ser usadas para várias aplicações, desde a identificação do comportamento social baseado na fala [9] até a identificação das atividades do usuário, analisando dados de sensores [10]. Em [11] os autores afirmam que o reconhecimento de atividades humanas é baseada na descoberta de padrões e no reconhecimento preciso da própria atividade. O CHSPAM [12] propõe um modelo computacional que descobre padrões sequenciais em bancos de dados de históricos de contextos e monitora a evolução desses padrões ao longo do tempo.

O sistema proposto acessa bancos de dados de aplicativos externos ou arquivos para extrair históricos de contexto periodicamente. Posteriormente, o modelo executa o reconhecimento de padrões nestes dados históricos e segue acompanhando esses padrões, armazenando informações relacionadas a eles em um banco de dados específico. Os serviços do sistema fornecem informações padrão e podem ser consumidos por aplicativos clientes.

A arquitetura proposta é um *MVC (Model-View Controller)* conforme mostrado na figura 3. A camada de visualização possui um grupo de serviços responsáveis pelo fornecimento de informações dos padrões para aplicativos externos. A camada do controlador, dividida em quatro módulos, é responsável pela composição dos históricos de contexto, execução de tarefas de transformação de dados, monitoramento de padrões e consultas. A camada do modelo é composta por um banco de dados que mantém os históricos de contexto armazenados localmente e um banco de dados, que armazena informações relacionadas a padrões.

O módulo *Context History Composition* executa consultas periódicas às bases de dados da

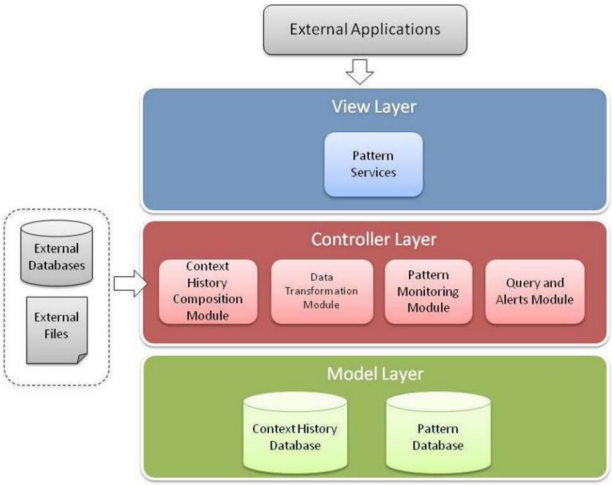

Figura 3: Arquitetura do modelo CHSPAM

aplicação do cliente, extraindo e armazenando informações relativas ao contexto. A representação dos contextos utiliza informações que descrevem a entidade, sua situação e localização.

O módulo *Pattern Monitoring* recebe os dados filtrados do módulo *Data Transformation* e é responsável por executar o reconhecimento de padrões aplicando processos de mineração de dados no banco de dados de histórico de contexto. A análise de padrões é realizada na seguinte sequência:

1) Os históricos de contexto são convertidos em um conjunto de vetores de sequência, utilizando todo o banco de dados como fonte de dados (figura 4). Cada contexto é mapeado para uma tupla, composta pelo identificador da entidade e um vetor de informações de contexto, armazenadas em pares chave/valor. Uma função de redução é aplicada, agregando o vetor de informações contextuais de cada contexto em uma nova sequência para cada identificador de entidade.
2) A partir do conjunto de dados gerado no passo anterior é executado o processo de reconhecimento de padrões. Foram utilizadas bibliotecas *Apache Spark* para a extração de sequência de padrões, onde são gerados como saída, estruturas de dados que armazenam os padrões e suas frequências.
3) Essas estruturas (modelos) são enviados para um algoritmo de monitoramento que verifica se já existem padrões com a mesma

sequência no banco de dados de padrões. Se for este o caso, a frequência do padrão é comparada à frequência anterior a fim de identificar se houve alteração. Se o padrão não estiver presente no banco de dados, um evento de descoberta é gerado. Finalmente, o histórico de padrões é atualizado para armazenar essas informações.

Depois disso, compara-se o resultado da última extração de padrões executada com os padrões atuais para identificar a extinção de padrões. Por fim, armazena-se o resultado dessa extração para ser utilizado na próxima.

Os padrões têm um identificador único, informações sobre o tipo de padrão e a estratégia usada na descoberta e a sequência de contexto que compõe esse padrão. Cada padrão tem seu histórico que contém eventos relacionados e a frequência apresentada nesse *timestamp*. Cada padrão tem seu histórico que contém eventos relacionados e a frequência apresentada naquele instante de tempo.

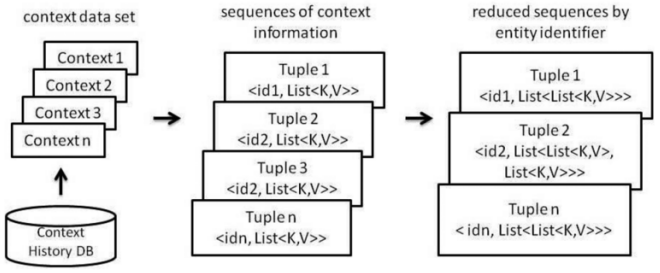

Figura 4: Conversão do histórico de contexto em sequência de vetores (CHSPAM)

O modelo foi testado através da implementação de um sistema de reconhecimento de atividades através do monitoramento de padrões sequenciais utilizando previsão de contexto.

O objetivo do experimento foi comparar a previsão de contextos usando padrões sequenciais comuns (PC) e padrões sequenciais monitorados (MP). O PC possui apenas informações simples do padrão enquanto que o MP também possui informações históricas sobre frequências e eventos anteriores. Foi utilizado o algoritmo *PrefixSpan* para mineração da sequência de padrões.

Uma ferramenta foi implementada para testar e comparar as previsões de atividades feitas utilizando os *datasets* de PC e MP. Em geral, predições utilizando os padrões sequenciais monitorados foram mais precisas, especialmente para previsões onde o número de contextos é pequeno. Os resultados também demonstraram que o uso de padrões sequenciais monitorados apresentaram uma acurácia até 17% maior que utilizando padrões sequenciais comuns.

Este projeto apresentou a criação de um modelo que permite a descoberta de padrões sequenciais em bancos de dados de históricos de contexto usando uma abordagem genérica para representação de contexto e demonstrou a capacidade do modelo para monitorar padrões descobertos ao longo do tempo.

4.3 TELLUS

O uso do contexto em sistemas computacionais pode ser aplicado nas mais diversas áreas, inclusive no setor agrícola. O *Tellus*, proposto por [13], é um modelo computacional aplicado na agricultura de precisão que utiliza os históricos de contexto para predição da fertilidade do solo. O contexto aqui, ao contrário dos sistemas mostrados anteriormente, contém informações relacionados ao meio ambiente, e não mais às atividades humanas. Aqui são considerados para a formação de contexto, os aspectos químicos e físicos que caracterizam os diferentes tipos de solo ao longo do tempo.

A análise do solo é um método amplamente utilizado na agricultura de precisão pois permite, antes do plantio, conhecer a capacidade de nutrientes que o solo poderá fornecer para as plantas. Esse conhecimento prévio viabiliza o uso racional de insumos, aumentando a produtividade e mitigando impactos ambientais. Neste projeto é proposto um modelo computacional denominado *Tellus*, que utiliza paradigma de computação ubíqua e processamento de histórico de contexto para realizar de maneira eficaz a predição da fertilidade do solo.

Sua arquitetura, conforme mostrada na figura 5, é composta por três blocos: um assistente móvel, um *middleware* para os sensores conectados em atuadores e um servidor. O assistente móvel é responsável por mostrar um mapa em tempo real da qualidade de fertilidade do solo. O *middleware* do atuador agrícola coleta os dados do solo e notifica o tratorista se o local onde ele se

encontra no momento está necessitando de tratamento (aplicação de fertilizantes). Os algoritmos de predição são executados no servidor e utiliza regressão linear múltipla para analisar os dados do histórico de contexto.

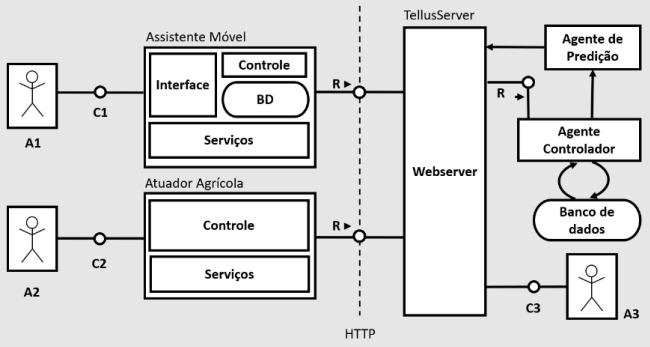

Figura 5: Arquitetura do *Tellus*

As propriedades que formam o contexto contém informações dos dados espectrais coletados do solo, localização, entre outros, conforme mostrado a seguir:

- *Evento*: Data e hora corrente (*timestamp*);
- *Latitude*: Coordenadas da latitude;
- *Longitude*: Coordenadas da longitude;
- *Infravermelho*: Dados espectrais em formato .CSV;
- *Matéria orgânica*: Índice de concentração de matéria orgânica;
- *Argila*: Índice de concentração de argila;
- *IP*: Endereço IP da requisição;

Foram coletadas 355 amostras diversas de diversos locais e, após tratamento de secagem e moagem, foi efetuada a aquisição dos espectros com o auxílio de um espectômetro com fibra óptica. Para gerar o modelo de calibração foram usados os dados de infravermelho dos históricos de contexto, um para matéria orgânica e outro para argila. Após, foi feita uma reanálise dos dados via algoritmo de Regressão por Mínimos Quadrados Parciais, utilizando 18 componentes ou variáveis latentes. A figura 6 mostra a performance do modelo para matéria orgânica.

Comparando os resultados do modelo de calibração gerados pelos dados históricos do *Tellus* com trabalhos de diferentes autores, foram atingidos melhores índices de avaliação preditiva, comprovando que o histórico de contexto pode ser utilizado em cenários onde a atividade humana não faz parte do escopo da análise.

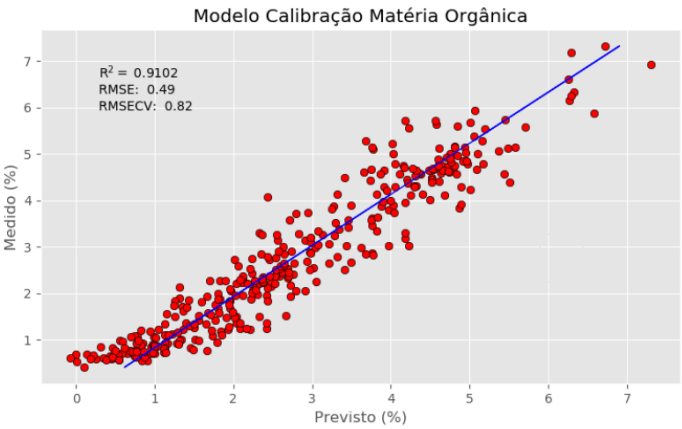

Figura 6: Modelo de calibração para matéria orgânica do *Tellus*

4.4 CMFRAME

O CMFrame [14] é um *framework* cujo principal diferencial é considerar os contextos como entidades que possuam características dinâmicas, no sentido de oferecer recursos que permitam seu uso em ambientes cujos elementos internos se modifiquem constantemente ao longo do tempo. Ele modela um gerenciador de históricos de contextos onde as entidades são organizadas de forma hierárquica. Essa hierarquia tem como objetivo definir como essas entidades se relacionam entre si e com o ambiente. Essas relações poderão ser alteradas dinamicamente conforme a utilização do sistema, por isso as entidades são denominadas Entidades Hierárquicas e Dinâmicas.

O *framework* possibilita mudanças hierárquicas entre os contextos e entidades em tempo de execução. É possível, por exemplo, que uma entidade contenha o contexto de outra entidade. O sistema também permite que entidades se conectem a uma grupo de entidades como nodos filhos, mudem de grupos ou se separem em entidades independentes. Esse dinamismo permite uma melhor representação de ambientes mutáveis onde os elementos realizam trocas internas constantes como no caso de sistemas que possuam muitas entidades móveis.

Uma entidade no CMFrame deve ser capaz de gerar dados (através de sensores) e/ou modificar o sistema através de atuadores. Ela pode representar qualquer elemento de um ambiente: um robô, um *hardware* dotado de sensores, uma sala com componentes de IoT, uma andar inteiro dessas salas ou até mesmo um prédio contendo esses

andares.

A entidade é primeiramente classificada como física ou móvel, para configurar se é necessário enviar dados de localização. Os dados mínimos de contexto exigem que seja informado a data em que a informação de contexto foi coletada, a fim de manter o registro de histórico, algum tipo de identificador utilizado pelo usuário para posterior consulta. Além disso as entidades podem enviar quaisquer outros dados adicionais que acharem relevantes, sem a necessidade de seguir um formato prévio.

O *framework* possui um módulo de banco de dados capaz de gerenciar as funcionalidades de históricos de contextos. Optou-se por uma base NoSQL para armazenamento de dados dos sensores devido à sua alta escalabilidade e disponibilidade, sendo capazes de incluir de maneira dinâmica novos atributos aos registros além de distribuir os dados entre vários servidores de maneira eficiente. As requisições por dados de contexto podem ser feitas para uma entidade passando como parâmetros o ID e quais dados da entidade se deseja adquirir. A busca hierárquica por contextos utiliza os mesmos parâmetros porém nesse caso toda a hierarquia é percorrida a partir da entidade desejada.

Para validar o *framework*, foi implementada uma aplicação de *smart home* capaz de monitorar um ambiente dotado de sensores fixos e agentes móveis humanos e robôs. A avaliação realizada sobre o aplicativo visou analisar os controles de troca hierárquica, e se eles apresentam contribuições para seus utilizadores.

O aplicativo desenvolvido utilizando o *CMFrame* monitorava o estado atual de todos os ambientes de uma casa e ativava atuadores mediante inferência sobre os dados de contexto. A figura 7 exibe alguns dos sensores com base no ambiente em que estão localizados. As mudanças hierárquicas eram solicitadas mediante a análise dos dados de contexto vindos dos sensores de presença, sempre mantendo a localização onde a entidade se encontra através da hierarquia. Todas as trocas de hierarquia eram armazenadas nos históricos das entidades, através do uso dos contextos dinâmicos do *CMFrame*.

Com base nas informações dos sensores fixos e móveis (usuários com *smartphone*) o aplicativo verificava os dados de contexto apropriados

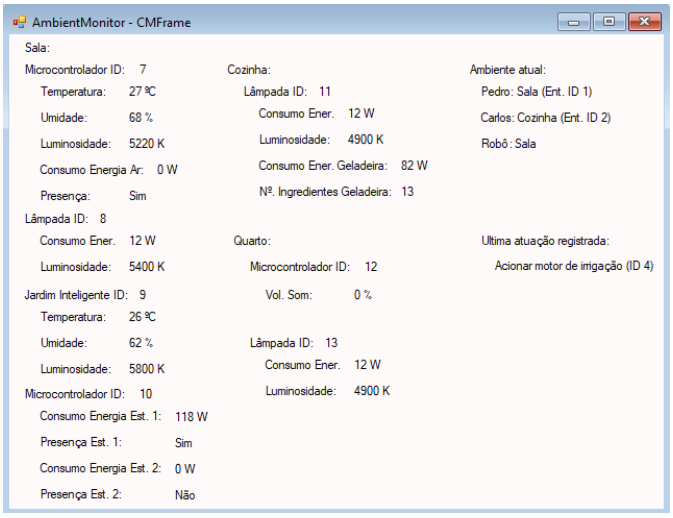

Figura 7: Aplicação de teste do *framework* do *CMFrame*

para detectar os deslocamentos e solicitava tanto as trocas de hierarquia como o armazenamento no histórico da entidade do ambiente em que se encontrava. Desta maneira era possível, centralizando tudo em um aplicativo único, saber informações sobre os deslocamentos e atividades dos usuários e ainda controlar os atuadores dinamicamente. Era possível saber, por exemplo, em qual ambiente o robô de limpeza estava, se estava limpando de maneira eficiente, o número de limpezas diárias, o momento do início e do fim da limpeza, tudo isso através da verificação de contextos do ambiente específico.

Ao final, pôde-se analisar mais uma forma de utilização de processamento de histórico de contextos. Nesta análise se propôs a utilização de entidades hierárquicas e dinâmicas como forma de se obter uma visão mais real e atualizada dos ambientes monitorados/controlados provendo suporte a informações de contexto variadas. O *framework* proposto oferece suporte ao gerenciamento de uma base de dados NoSQL que armazena os dados de contexto bem como uma interface para que aplicações possam utilizar esses dados para tomada de decisões acerca do ambiente e reagir aos usuários que nele interagem.

4.5 TrailCare

O *TrailCare* [15] é um sistema computacional de auxílio a usuários de cadeiras de rodas através do uso de tecnologias ubíquas e móveis. Ele

utiliza sensoriamento para detectar a localização do usuário, tanto em ambientes internos como externos, para desenvolver soluções estratégicas de acessibilidade recomendando rotas seguras.

Os deslocamentos do cadeirante, baseado em sua localização, são armazenados em históricos de contexto chamados de trilhas. Baseado nelas, é possível realizar inferências sobre o comportamento do usuário e recomendar recursos de acessibilidade contextualizados.

A arquitetura do sistema é composta por quatro camadas (figura 8):

1) *Server*: que utiliza *web services* para fornecer recursos de acessibilidade em contextos e executar o gerenciamento das trilhas;
2) *Client*: aplicativo móvel que fornece a interface com o usuário, mostrando os recursos de acessibilidade sensíveis ao contexto baseado em seus deslocamentos. Também envia as informações de localização para que o *Server* armazene as trilhas. As localizações externas são obtidas por GPS e as internas por leituras de cartões *RFID*.
3) *Middleware*: executa em um microcontrolador e é responsável por comunicar com o aplicativo, ler os cartões *RFID* e comunicar via serial com o *firmware* da cadeira de rodas inteligente;
4) *Firmware* da Cadeira de Rodas: envia e recebe mensagens do *middleware* (terceira camada) via comunicação serial;

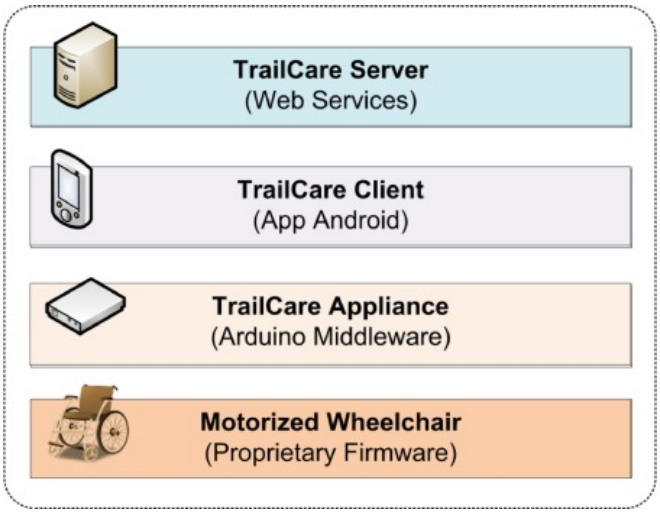

Figura 8: Arquitetura do sistema *TrailCare*

O *TrailCare Server* é dividido em módulos que, juntos, são capazes de mapear recursos de acessibilidade de uma área e utilizá-los para auxiliar os cadeirantes usuários do sistema. O componente que armazena as informações referentes aos usuários é o *Profile System*. Nele, são armazenados dados como nome, nível de incapacidade (por exemplo, paraplégico ou tetraplégico) e preferências (interesse em recursos de acessibilidade específicos como área de estacionamento, telefones públicos adaptados, etc). Também armazenam contatos em caso de situações de emergência. O *Context System* gerencia e armazena os contextos. Os contextos registram os recursos de acessibilidade disponíveis nas regiões e são a base para as sugestões contextualizada de recursos para os usuários do sistema O *Trail Manager* gerencia as trilhas (internas e externas) dos usuários. O *Acessibility Assistant* é o mecanismo que gerencia o servidor e gera a recomendação de recursos contextualizados e o *Service Manager* opera como uma camada de comunicação entre o componente da interface (*TrailCare Client*) e os outros componentes.

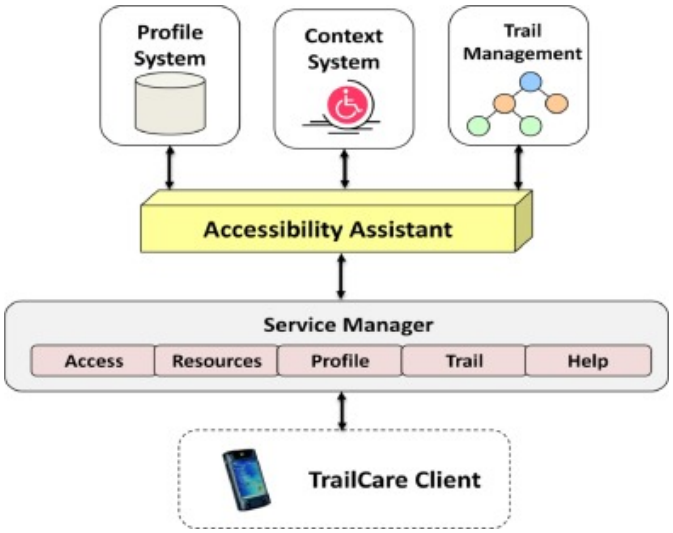

Figura 9: Arquitetura do *TrailCare Server*

Os usuários possuem um identificador único vinculado a um perfil que armazena suas informações pessoais e preferências. O tempo e a localização são monitorados continuamente e adicionados às trilhas. Por fim, os status das entidades-objeto descrevem os recursos de acessibilidade disponíveis nos ambientes (ex. rampas, banheiros para cadeirantes, etc). Baseado na localização e no perfil do usuário o *TrailCare* mostra os recursos de interesse próximos a ele.

As inferências são realizadas na seguinte

Sistema	Banco de Dados	Tratam. dos Históricos	Técnica de Inferência	Propriedades do Contexto	Localiz.?	Aplicação
SDE	Relacional	Séries Temporais	Support Vector Machine	Entidade, Localiz, Atividade, Tempo, Sensores (FC, VFC)	Sim	Saúde (Detecção de Situações de Estresse Mental)
CHSPAM	Genérico	Reconhecim. de Padrões	Genérico	Entidade, Localiz„ Tempo	Sim	Genérica
TELLUS	Relacional	Regressão Mínimos Quadr. Parciais	Regressão Linear Múltipla	Entidade, Localiz., Tempo, Sensores (IR)	Sim	Agro (Qualidade da fertilidade do solo)
CMFRAME	NoSQL	Genérico	Genérica	Entidade, Localiz., Tempo, Sensores	Sim	Genérica
TRAILCARE	Relacional	Não especificado	Não especificado	Entidade, Localiz., Tempo, Trilhas	Sim	Acessibilidade (Rotas e recursos p/ cadeirantes)

Tabela 1: Comparativo dos sistemas com relação ao processamento de histórico de contexto

sequência:

1) O aplicativo envia o *trailpoint* (ponto de trilha) para o *Acessibility Assistant* que registra a informação no *Trail Manager*;
2) O próprio *Assistant* acessa o *Profile Manager* para ler as preferências do usuário;
3) Depois, conecta-se ao *Context System* para identificar os recursos contextualizados disponíveis no local;
4) Finalmente, o *Assistant* informa o usuário através do aplicativo os recursos de sua preferência que se encontram no ambiente.

O aplicativo gera uma lista dos recursos de acessibilidade ordenados por proximidade mediante cálculo de distância estimada. E mostra em mapas a localização desses recursos tanto em ambientes internos como externos. O sistema foi validado utilizando-se 10 usuários cadeirantes em uma região conhecida cujo ambiente interno era dotado de várias *tags* RFID espalhadas em locais estratégicos.

O projeto *TrailCare* implementou um sistema de apoio à acessibilidade para usuários de cadeiras de rodas através da recomendação de acessibilidade contextualizada de recursos, que pode ser utilizado em ambientes internos e externos. Além disso, apresentou uma abordagem simples para o cálculo de reconhecimento de trilhas registrando caminhos já visitados anteriormente, utilizando de maneira simples e eficaz um banco de dados relacional para armazenamento e tratamento de histórico de contexto.

5 Considerações Finais

Com o crescimento da computação móvel e pervasiva, onde os dispositivos são capazes de descobrir no ambiente outros dispositivos móveis, seus serviços e recursos e se conectar a eles, o contexto em que esses dispositivos se encontram é uma peça chave para extrair informações e inferir situações. Armazenar esses dados ao longo do tempo, para gerar um histórico de contexto pode ser utilizado para melhorar a eficácia dessas inferências bem como ampliar a gama de utilização das aplicações.

Esta pesquisa mostrou cinco exemplos de sistemas computacionais ubíquos atuais que utilizam histórico de contexto para inferência de situações nas mais diferentes áreas de aplicação. Cada um desses sistemas, armazena e processa os dados de maneira única conforme mostrado na tabela 1.

Como trabalhos futuros sugere-se ampliar o escopo das aplicações, buscando um número maior de sistemas que utilizam esse conceito, classificando-os por grupos conforme as estruturas que utilizam para gerenciar os históricos de contexto e comparando as vantagens e desvantagens de se utilizar cada estrutura específica.

Referências

[1] L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. *Computer Networks*, pages 2787–2805, 10 2010.

[2] M. R. Endsley. Measurement of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37:65–84, 1995.

[3] A. K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5:4–7, 2001.

[4] Hong J, Suh EH, Kim J, and Kim S. Context-aware system for proactive personalized service based on context history. *Expert Syst Appl*, 36, 2009.

[5] Perera C, Zaslavsky A, Christen P, and Georgakopoulos D. Context aware computing for the internet of things: a survey. *Commun Surv Tutor*, 16, 2014.

[6] Nurmi P, Martin M, and Flanagan JA. Enabling proactiveness through context prediction. *Workshop on context awareness for proactive systems*, page 53, 2005.

[7] Satyanarayanan M. Pervasive computing: vision and challenges. *IEEE Personal Commun*, 8:10–17, 2001.

[8] R. Bavaresco and J. Barbosa. Análise de históricos de contextos para classificação do estresse mental em situações reais por meio da variabilidade da frequência cardíaca. *Simpósio Brasileiro de Computação Ubíqua e Pervasiva*, 11, 2019.

[9] Rachuri KK, Musolesi M, Mascolo C, Rentfrow PJ, Longworth C, and Aucinas A. Emotionsense: a mobile phones based adaptive platform for experimental social psychology research. pages 281–290, 2010.

[10] Wood AD, Stankovic JA, Virone G, Selavo L, He Z, Cao Q, Doan T, Wu Y, Fang L, and Stoleru R. Context-aware wireless sensor networks for assisted living and residential monitoring. *IEEE Netw*, 22:26–33, 2008.

[11] Yürür Ö, Liu CH, Sheng Z, Leung VC, Moreno W, and Leung KK. Context-awareness for mobile sensing: a survey and future directions. *IEEE Commun Surv Tutor*, 18:68–93, 2014.

[12] D. Dupont, J. L. V. Barbosa, and B. M. Alves. Chspam: a multi-domain model for sequential pattern discovery and monitoring in contexts histories. *Pattern Analysis and Applications*, 2019.

[13] G. Helfer, B. Martini, R. ; Santos, A. Costa, and J. Barbosa. Tellus: um modelo computacional para a predição da fertilidade do solo na agricultura de precisão. *Anais do XI Simpósio Brasileiro de Computação Ubíqua e Pervasiva*, 11, 2019.

[14] F. L. Vielitz, M. G. Martins, c. L. V. Barbosa, K. F. de Oliveira, L. P. S. Dias, and A. Wolf. Cmframe: a framework for managing dynamic and hierarchical context histories. pages 1–8, 2019.

[15] J. Barbosa, J. Tavares, I. Cardoso, B. Alves, and B. Martini. Trailcare: an indoor and outdoor context-aware system to assist wheelchair users. *International Journal of Human-Computer Studies*, 116, 04 2018.

Pedro A. Tavares Mestrando do Programa de Pós-Graduação em Engenharia Eletrônica e Computação da Universidade Católica de Pelotas (UCPEL), na linha de pesquisa Sistemas Computacionais Embarcados e Distribuídos. Possui graduação em Engenharia de Computação pela Universidade Federal do Rio Grande (FURG) e MBA em Gerenciamento de Projetos pela Fundação Getúlio Vargas (FGV). Tem experiência na área de Ciência da Computação, com ênfase na área de Sistemas Embarcados.