UNISINOS

Programa Interdisciplinar de Pós-Graduação em

Computação Aplicada

Mestrado Acadêmico

Daniel Ambrosi Dupont

CHSPAM: Um Modelo Multi-domínio para Acompanhamento de Padrões em Históricos de Contextos

São Leopoldo, 2017

Daniel Ambrosi Dupont

CHSPAM: UM MODELO MULTI-DOMÍNIO PARA ACOMPANHAMENTO DE PADRÕES EM HISTÓRICOS DE CONTEXTOS

Dissertação apresentada como requisito parcial para a obtenção do título de Mestre pelo Programa de Pós-Graduação em Computação Aplicada da Universidade do Vale do Rio dos Sinos — UNISINOS

Orientador:
Prof. Dr. Jorge Luis Victória Barbosa

São Leopoldo
2017

D938c Dupont, Daniel Ambrosi.
CHSPAM: um modelo multi-domínio para acompanhamento de padrões em históricos de contextos / Daniel Ambrosi Dupont. – 2017.
97 f. : il. color. ; 30 cm.

Dissertação (mestrado) – Universidade do Vale do Rio dos Sinos, Programa de Pós-Graduação em Computação Aplicada, 2017.
"Orientador: Prof. Dr. Jorge Luis Victória Barbosa".

1. Computação ubíqua. 2. Banco de dados. 3. Mineração de dados (Computação). 4. Históricos de contextos. I. Título.

CDU 004.75.057.5

Dados Internacionais de Catalogação na Publicação (CIP)
(Bibliotecária: Carla Maria Goulart de Moraes – CRB 10/1252)

Daniel Ambrosi Dupont

Título: CHSPAM: Um Modelo Multi-Domínio Para Acompanhamento De Padrões Em Históricos De Contextos

Dissertação apresentada à Universidade do Vale do Rio dos Sinos – Unisinos, como requisito parcial para obtenção do título de Mestre em Computação Aplicada.

Aprovado em 21 de março de 2017

BANCA EXAMINADORA

Prof. Dr. Fabiano Hessel – PUC/RS

Prof. Dr. Cristiano André da Costa – UNISINOS

Prof. Dr. Jorge Luís Victória Barbosa (Orientador)

Visto e permitida a impressão
São Leopoldo,

Prof. Dr. Sandro José Rigo
Coordenador PPG em Computação Aplicada

RESUMO

A Computação Ubíqua estuda o desenvolvimento de técnicas que visam integrar perfeitamente a tecnologia da informação ao cotidiano das pessoas, de modo que elas sejam auxiliadas pelos recursos tecnológicos no mundo real, de forma pró-ativa, enquanto realizam atividades diárias. Um dos aspectos fundamentais para o desenvolvimento deste tipo de aplicação é a questão da Sensibilidade ao Contexto, que permite a uma aplicação adaptar o seu funcionamento conforme o contexto no qual o usuário se encontra. Com o desenvolvimento de sistemas que utilizam informações de contextos armazenados anteriormente, foram surgindo bases de dados que armazenam os Históricos de Contextos capturados ao longo do tempo. Muitos pesquisadores têm estudado diferentes formas para realização de análises nestes dados. Este trabalho aborda um tipo específico de análise de dados em históricos de contextos, que é a busca e acompanhamento de padrões. Deste modo, é proposto um modelo denominado CHSPAM (*Context History Pattern Monitoring*) que permite a realização de descoberta e acompanhamento de padrões sequenciais em bases de Históricos de Contextos, fazendo uso de técnicas de mineração de dados já existentes. O diferencial deste trabalho é o uso de uma representação genérica para o armazenamento de contextos, permitindo sua aplicação em múltiplos domínios. Outro diferencial é que o modelo realiza o acompanhamento dos padrões descobertos durante o tempo, armazenando um histórico da evolução de cada padrão. Um protótipo foi implementado e, a partir dele, foram realizados três experimentos. O primeiro foi utilizado para avaliar as funcionalidades e serviços oferecidos pelo CHSPAM e foi baseado em dados sintéticos. No segundo, o modelo foi utilizado em uma aplicação de predição e o acompanhamento de padrões proporcionou ganhos na precisão das predições quando comparado ao uso de padrões sem acompanhamento. Por fim, no terceiro experimento, o CHSPAM foi utilizado como componente de uma aplicação de recomendação de objetos de aprendizagem e a aplicação foi capaz de identificar objetos relacionados aos interesses de alunos, utilizando como base o acompanhamento de padrões.

Palavras-Chave: Computação Ubíqua. Descoberta de Padrões. Históricos de Contextos. Mineração de dados.

ABSTRACT

Ubiquitous computing aims to make tasks that depend on computing, transparent to users, thus, providing resources and services anytime and anywhere. One of the key factors to the development this type of application is the matter of Context Awareness, which enables an application to adjust its operation as the situation in which the user is. Thus, several authors have presented formal definitions of what is a context and how to represent it. With the development of systems that use Context information previously stored, databases have emerged that store Historical Contexts captured over time. Many researchers have studied different ways to analyzes this data. This paper addresses a specific type of data analysis in historical contexts, which is the discovery and monitoring of patterns in Context Histories. For this purpose, a model called CHSPAM (Context History Pattern Monitoring) is proposed, that allows the discovery of patterns in Context History databases and keeps track of these patterns to monitor their evolution over the time. Ubiquitous computing aims aim to integrate information technology perfectly into people's daily lives, so that people are aided by technological resources in the real world, proactively, while performing daily activities. One of the fundamental aspects for the development of this type of application is the issue of Context Awareness, which allows an application to adapt its operation according to the context in which the user is. With the development of systems that use information from previously stored contexts, databases have emerged that store captured Context Histories over time. Many researchers have studied different ways to perform analyzes on these data. This work addresses a specific type of data analysis in context histories, which is the search for sequential patterns. With this purpose, a model called CHSPAM (Context History Pattern Monitoring) is proposed that allows the discovery of sequential patterns in Context Historical databases, making use of existing data mining techniques. The main contributions of this work are the use of a generic representation for the storage of contexts allowing its application in multiple domains. Another contribution is that the model monitors the patterns discovered over time, storing history pattern evolution. A prototype of the model was implemented, and from it three experiments were carried out for its validation. The first experiment was used to evaluate the functionalities and services offered by CHSPAM and was based on simulated data. In the second experiment, the model was used in a prediction application and the use of monitored sequential patterns provided accuracy improvement on predictions when compared to the use of common patterns. Finally, in the third experiment, CHSPAM was used as a component of a learning object recommendation application and the application was able to recommend objects related to students' interests based on monitored sequential patterns extracted from users' session history.

Keywords: Ubiquitous Computing. Sequential Pattern Discovery. Context History. Data Mining.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

AI	Artificial Intelligence
CCD	Charged-couple Device
CRF	Conditional Random Field
FS	Feature Selection
GSP	Generalized Sequential Patterns
HMM	Hidden Markov Model
KDD	Knowledge Discovery in Databases
NBR	Normas Brasileiras de Regulação
PST	Probabilistic Sufix Tree
PTM	Probabilistic Topic Model
RFID	Radio Frequency Identification
SVM	Support Vector Machine
TMAP	Temporal Mobile Access Pattern
UP	User Profiling
JSON	Javascript Object Notation
REST	Representational State Transfer
NOSQL	Not Only SQL

SUMÁRIO

1 INTRODUÇÃO

A área da Computação Ubíqua estuda o paradigma de diversos componentes de hardware e software trabalhando de forma transparente e comunicando-se através de redes sem fio (WEISER, 1991). O valor agregado através da interação inteligente de múltiplos dispositivos de forma autônoma tem um potencial muito maior do que o originado de um dispositivo simples e isolado. Um objetivo estratégico deste tipo de sistema é simplificar drasticamente a interação entre usuários e computadores, fazendo uso de sensores, dispositivos computacionais e atuadores na estrutura das atividades diárias dos usuários, de modo que a sua presença e complexidade permaneça oculta.

Para Weiser (1991), a Computação Ubíqua estuda o desenvolvimento de técnicas que visam integrar perfeitamente a tecnologia da informação ao cotidiano das pessoas, de modo que as pessoas sejam auxiliadas pelos recursos tecnológicos no mundo real, de forma pró-ativa, enquanto realizam atividades cotidianas.

Dey, Abowd e Salber (2001) caracterizam a percepção dos sistemas computacionais ao meio onde o usuário está inserido como um aspecto vital para que a visão de Weiser (1991) se torne realidade. A esta característica Dey, Abowd e Salber (2001) dão o nome de Sensibilidade ao Contexto, onde Contexto é qualquer tipo de informação que permita caracterizar a situação de entidades que sejam relevantes para a interação entre um usuário e um sistema. Isto significa, em geral, informações sobre a situação, identidade e localização espaço-temporal de pessoas, grupos e objetos físicos ou computacionais. Através do conhecimento de dados contextuais, uma aplicação pode ajustar seu próprio funcionamento ou ainda agir de maneira proativa, alertando o usuário de algum perigo ou ajudando-o a realizar suas atividades de forma mais eficiente.

A capacidade de reconhecer apenas a situação atual de um usuário é vista por Dey, Abowd e Salber (2001) como algo trivial. A captura e o armazenamento de informações contextuais ao longo do tempo possibilita uma série de análises mais avançadas. O histórico de contextos pode ser utilizado, por exemplo, para estabelecer tendências e até prever valores de futuros contextos.

Com o desenvolvimento de sistemas que utilizam informações de contextos armazenadas anteriormente, alguns modos de representação formal foram criados para que estas informações fossem armazenadas, gerando, assim, bases de dados que armazenam as sequências de contextos capturados ao longo do tempo. Estes dados também são comumente chamados Históricos de Contextos (HONG, SUH, *et al.*, (2009) DEY, ABOWD e SALBER, 2001). Muitos pesquisadores têm estudado diferentes formas para realização de análises nestes dados (SILVA, ROSA, *et al.*, 2010; WIEDEMANN, 2014; WAGNER, BARBOSA e BARBOSA, 2014; DA ROSA, BARBOSA e RIBEIRO, 2016).

Este trabalho aborda um tipo específico de análise de dados em históricos de contextos, que é a busca e o acompanhamento de padrões sequenciais. Com este propósito, é proposto um modelo denominado CHSPAM (*Context History Pattern Monitoring*), que permite a descoberta e o acompanhamento de padrões em bases de históricos de contextos.

1.1 Motivação

A sensibilidade ao contexto em sistemas é um aspecto essencial para integração da tecnologia ao cotidiano das pessoas. Deste modo, o desenvolvimento de sistemas sensíveis ao contexto é um tema de pesquisa estratégico da área de Computação Ubíqua.

Satyanarayanan (2001) afirma que um sistema de Computação Ubíqua que pretenda atender ao requisito de agir de forma proativa e não intrusiva deve possuir recursos para reconhecimento do contexto atual. Esta capacidade permitiria ao sistema identificar informações relevantes para o usuário. Se, naquele determinado momento, não for possível interromper o usuário para notificá-lo, o sistema poderá tomar a decisão de aguardar um momento mais oportuno.

Segundo Dey, Abowd e Salber (2001), a área de computação sensível ao contexto ainda está no início de suas pesquisas. De acordo com eles, o armazenamento de dados históricos de contexto está ligado diretamente à necessidade de disponibilidade constante de aquisição de contexto, que é um dos requisitos básicos para um *framework* que suporte o desenvolvimento de aplicações sensíveis ao contexto.

Com o desenvolvimento de sistemas como estes, houve a necessidade de representar formalmente dados de contextos e armazená-los, gerando, assim, bases de dados que armazenam as sequências de contextos capturados ao longo do tempo. Desde então, diversos pesquisadores têm estudado as oportunidades que podem surgir a partir da análise destes dados.

Silva *et al.* (2010) chamam estas sequências de trilhas e propõem um modelo chamado UbiTrail, que pode ser acessado por aplicações clientes que necessitem obter e analisar dados sobre as trilhas. Wiedemann (2014) apresenta o SIMCOP, que é um *framework* que possibilita a realização de cálculos de métricas de similaridade entre contextos simples e entre duas sequências de contextos.

Wagner, Barbosa e Barbosa (2014) propõem um modelo que permite às aplicações registrarem ações de entidades em trilhas e inferir informações de perfis a partir delas. Mais recentemente, Da Rosa, Barbosa e Ribeiro (2016) desenvolveram um modelo chamado Oracon, que prevê os contextos futuros de uma entidade a partir de inferências sobre seu histórico de contextos utilizando múltiplas análises de inferência.

Segundo Yurur, Liu, *et al.* (2014), é sabido que, introduzindo inteligência e consciência situacional no reconhecimento de padrões de processos humanos, pode-se obter um melhor entendimento do comportamento humano e também permitir o auxílio proativo a indivíduos para melhorar sua qualidade de vida. Yurur, Liu, *et al.* (2014) ainda afirmam que o entendimento das atividades humanas baseia-se na descoberta de um padrão de atividade e no reconhecimento preciso da própria atividade.

Pode-se observar que a análise de dados de históricos de contexto também tem se tornado um tema de pesquisa estratégico na área de Computação Ubíqua. Entre os diversos tipos de análises que podem ser realizados, a busca por padrões destaca-se, por prover ferramentas para identificação de atividades e comportamento humano. Tais ferramentas auxiliam desde a identificação de padrões de comportamento social de pessoas, analisando padrões de fala e interação (Rachuri, Musolesi, *et al.*, 2010), até a identificação de suas atividades, analisando padrões em dados de sensores (Wood, Stankovic, *et al.*, 2008). Mais algumas aplicações para busca de padrões em dados históricos de contexto são as seguintes:

- Descoberta de comportamento de usuários para melhorar a recomendação personalizada de serviços móveis (Lee, Paik, *et al.*, 2007);
- Reconhecimento de rotinas complexas, com múltiplas de atividades executadas diariamente por usuários (Huynh, Fritz e Schiele 2008);
- Reconhecimento e previsão de ações em ambientes de casas inteligentes (Fatima, Fahim *et al.*, 2013);
- Reconhecimento de atividades humanas e de estilo de vida, com foco em sistemas de assistência a saúde (Yin, Tian, *et al.* 2014, Jung e Chung 2015);
- Identificação de perfis, analisando a causa entre os padrões de comportamento dos usuários, com foco em saúde, segurança, finanças e mídias sociais (Chikhaoui, Wang, *et al.*, 2014);
- Aumentar a precisão de sistemas de rastreamento (Niu, Li 2014);

Embora existam trabalhos que utilizam técnicas de mineração de dados para realizar descoberta de padrões em dados contextuais, a mineração de padrões sequenciais é uma técnica pouco utilizada neste tipo de análise. Porém, tais padrões podem ser utilizados para melhorar a eficiência de sistemas de recomendação, auxiliar em predições, aumentar a usabilidade de sistemas, detectar eventos e, de forma geral, auxiliar a tomada de decisões estratégica em produtos (GUPTA e HAN, 2012). Deste modo, a descoberta de padrões sequenciais em históricos de contextos é uma questão relevante de pesquisa.

Porém, a descoberta de padrões sequenciais em históricos de contextos não é uma tarefa trivial, pois a natureza dos dados contextuais tende a ser heterogênea, combinando dados de diversas fontes, com formatos e significados diferentes. Apesar da existência de trabalhos que pretendam possibilitar aplicações em domínio genérico, as opções são reduzidas pelo fato de representarem as entidades como usuários. Neste trabalho, o conceito de entidade é usado no lugar do conceito de usuário. Enquanto geralmente o usuário compreende a pessoa que está utilizando o aplicativo, uma entidade refere-se a qualquer objeto (físico ou não) capaz de controlar um aplicativo. Deste modo, a pesquisa realizada demonstrou que há oportunidade para o desenvolvimento de um modelo que pode ser aplicado em múltiplos domínios, utilizando uma abordagem genérica para representação de contextos e entidades.

Outro aspecto de destaque dentro da análise de padrões é, além de realizar a simples descoberta dos padrões, também acompanhar a evolução dos mesmos, com o passar do tempo, enquanto a base de históricos vai recebendo mais registros.

1.2 Definição do Problema e Questão de Pesquisa

Os fatores apontados na seção anterior apresentam os motivos pelos quais o tema de “descoberta e acompanhamento de padrões em históricos de contextos” é uma questão desafiadora de pesquisa.

Diversos problemas devem ser tratados em softwares que realizem este tipo de análise, como por exemplo:

- Qual é a definição de padrão para históricos de dados contextuais?

- Como utilizar informações extraídas de padrões para melhorar a eficácia de sistemas ubíquos?
- Como lidar com a natureza heterogênea dos dados contextuais?
- Quais as melhores estratégias de descoberta de padrões para dados sequenciais de históricos?
- Como acompanhar e representar a evolução de um padrão ao longo do tempo?

Estes questionamentos justificam a especificação e implementação de um modelo capaz de descobrir e acompanhar padrões em históricos de contextos, fornecendo ferramentas para configuração de descoberta e tratamento de dados.

Neste sentido, pode-se indicar como a questão de pesquisa deste trabalho: “Como seria um modelo que possibilitasse a descoberta e o acompanhamento de padrões em históricos de contextos para aplicações em múltiplos domínios?”.

1.3 Objetivos

Este trabalho de pesquisa tem como objetivo geral especificar, implementar e avaliar um modelo de descoberta e acompanhamento de padrões em históricos de contextos. Através do modelo, dados de históricos podem ser processados, e padrões podem ser detectados, armazenados e acompanhados em uma linha de tempo.

O modelo apresentado é genérico, ou seja, um modelo que permite trabalhar com qualquer tipo de dado de histórico de contextos, que será convertido em uma representação específica para a extração de padrões. Desta maneira o modelo pode ser utilizado para detecção e acompanhamento de dados de históricos de contextos de diversas aplicações.

Para alcançar esse objetivo geral, são definidos os seguintes objetivos específicos:

- Estudar os fundamentos teóricos da área;
- Identificar e comparar os trabalhos relacionados;
- Avaliar os modelos de representação para Históricos de Contextos;
- Especificar um modelo para representação de Padrões e seu acompanhamento;
- Especificar o modelo CHSPAM;
- Desenvolver um protótipo do modelo;
- Realizar a validação do protótipo através de experimentos utilizando dados sintéticos e reais.

1.4 Metodologia

Este trabalho foi desenvolvido a partir da investigação de tópicos que pudessem oferecer apoio teórico para a criação de um modelo para atender os objetivos descritos. Áreas

relacionadas ao tema foram pesquisadas, bem como áreas que dessem apoio tecnológico para a solução a ser desenvolvida.

Foi realizada uma pesquisa por trabalhos relacionados com os temas e objetivos propostos. A pesquisa resultou em uma análise comparativa dos trabalhos para avaliar as características e abordagens utilizadas pelos outros trabalhos e compor a especificação final do modelo. A partir do estudo efetuado, especificou-se um modelo inicial, detalhando seus componentes principais, assim como as interações entre os mesmos.

Objetivando-se avaliar o modelo proposto, foi desenvolvido um protótipo e a partir dele foram realizados três experimentos. O primeiro experimento foi utilizado para avaliar as funcionalidades e serviços oferecidos pelo modelo e foi baseado em dados sintéticos. No segundo, o modelo foi utilizado em uma aplicação de predição. Por fim, no terceiro experimento, o modelo foi utilizado em uma aplicação de recomendação de objetos de aprendizagem.

1.5 Organização do Texto

Esta dissertação está dividida em seis capítulos. No segundo capítulo são discutidos os tópicos de pesquisa que são relevantes para o trabalho proposto. No terceiro capítulo são apresentados os trabalhos relacionados e realizado um estudo comparativo de suas características, funcionalidades e limitações. O modelo proposto, com sua arquitetura e funcionalidades são apresentados no quarto capítulo. O quinto capítulo discute a implementação do protótipo e o resultado dos experimentos realizados. Por fim, o sexto e último capítulo contém as considerações finais e aponta os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

O presente capítulo apresenta na seção 2.1 os conceitos e tecnologias inerentes à computação ubíqua. As seções 2.2 e 2.3 trazem conceitos relacionados à sensibilidade e históricos de contextos. A seção 2.4 elenca os principais conceitos e tecnologias relacionadas à mineração de dados. Por fim, a seção 2.5 apresenta as principais técnicas para descoberta de padrões em dados sequenciais.

2.1 Computação Ubíqua

Em 1991 Mark Weiser apresentou ao mundo o conceito de Computação Ubíqua, enquanto trabalhava como pesquisador no Laboratório de Ciência Computacional da Xerox (Palo Alto Research Center). No clássico artigo, 'The Computer for the 21th century', Weiser previu ambientes onde a computação seria altamente distribuída e integrada, com diversos dispositivos computacionais, dando origem a esta nova área de estudo (WEISER, 1991).

Para Weiser (1991), as tecnologias que causam mais impacto na vida das pessoas são aquelas que de certa forma se tornam invisíveis. Tais tecnologias se integram tão bem ao cotidiano que são utilizadas de forma inconsciente. O autor compara o caminho que a computação deve percorrer ao caminho traçado pela tecnologia da escrita, uma tecnologia que permite transmitir ideias por séculos e que supera a capacidade de memória humana.

Ainda segundo Weiser (1991), a escrita é uma tecnologia ubíqua, uma vez que está disponível a todo o momento e em qualquer lugar. Além disso, ao fazer uso da escrita toda a atenção do leitor está focada no conteúdo do texto e não no processo de tradução de símbolos em ideias. Ou seja, o foco está na atividade ao invés da ferramenta, e a execução de tarefas deve ser feita sem a necessidade de atenção e ciência da tecnologia que está operando por trás dos panos.

Ishii e Ullmer (1997) apresentam uma visão de Computação Ubíqua ligeiramente diferente de Weiser, onde a ideia não é tornar os computadores ubíquos, mas sim proporcionar o despertar de objetos, instrumentos, superfícies e espaços da era pré-computador utilizando recursos computacionais. Para eles o esforço maior deve se dar em estudos para o desenvolvimento de interfaces naturais, que suportem a comunicação e a consciência entre humanos e computadores.

Abowd e Mynatt (2000) afirmam que estamos apenas começando a entender as implicações de uma imersão contínua em computação, como na visão provida por Weiser. Para eles o futuro iria trazer grande disponibilidade de ferramentas para auxiliar em tarefas computacionais. Tendo computação sendo vestida em nossos corpos ou integrada com nosso ambiente, teríamos também a habilidade de computadores alterando a nossa percepção do mundo físico, suportando conectividade constante a pessoas e locais distantes para nos trazer informações ao alcance da ponta dos dedos. Isso proporcionaria muito mais do que um aplicativo matador (*killer app),* proporcionaria uma existência matadora *(killer existence).*

Satyanarayanan (2001) vê a Computação Ubíqua como uma evolução de uma linha de pesquisa de Sistemas Distribuída iniciada na década de 1970. Inicialmente, preocupava-se com questões como tolerância a falhas, segurança, alta disponibilidade e processamento remoto. A partir do surgimento da Computação Móvel, outras questões importantes surgiram, como o consumo de bateria e miniaturização dos equipamentos.

Caceres e Friday (2011) salientam os avanços da comunidade de pesquisa de Computação Ubíqua, nos últimos 20 anos, que produziram e avaliaram inúmeros protótipos bem sucedidos, que demonstraram a utilidade de sistemas ubíquos em diferentes aspectos e áreas. Neste mesmo período a tecnologia digital também sofreu grandes avanços que possibilitaram a criação de produtos e serviços que vão de encontro com a visão da Computação Ubíqua e se tornaram parte do cotidiano de bilhões de pessoas em todo o mundo.

Caceres e Friday (2011) também elencam os principais desafios que dificultam a evolução de sistemas ubíquos até o ponto que eles se tornem realmente imperceptíveis. Entre estes desafios estão problemas de infraestrutura para garantir baixa latência na iteração entre os usuários e o ambiente, a falta de ferramentas de desenvolvimento para sistemas ubíquos, e o impacto da utilização constante de sensores no consumo de energia dos dispositivos.

Abowd (2012) afirma que o sucesso da Computação Ubíqua como a terceira geração da computação não pode ser negado: as realizações do passado devem ser celebradas e há uma previsão de rápidos avanços na área de ferramentas que finalmente vão auxiliar designers e especialistas de domínio na criação e desenvolvimento de aplicações ubíquas. Abowd (2012) ainda conta que Computação Ubíqua deixou de ser um tema de pesquisa de nicho, e agora é vista como o domínio intelectual da computação como um todo.

Abowd (2012) também salienta que, desde 2005 as linhas de pesquisa da Computação Ubíqua foram englobadas em pesquisas muitas comunidades cientificas diferentes, de tal modo que é impossível saber onde procurar os melhores resultados de pesquisas. Este é um diferencial que não pode ser revertido, resultando no desaparecimento desta linha de pesquisa, tendo ela se infiltrado em quase todos os aspectos das linhas de pesquisa de Computação.

Martínez-Torres, *et al* (2015) afirmam que os serviços de Computação Ubíqua constituem uma nova era da tecnologia da informação que pode ter milhares de aplicações e ambientes em potencial. Segundo eles, a computação ubíqua também está mudando o paradigma clássico da tecnologia da informação, pois está forçando mudanças sociais e culturais.

Com as tecnologias de Computação Ubíqua dos dias de hoje, atividades diárias de usuários estão sendo rastreadas pelos *smartphones* em seus bolsos e muitos de seus objetos utilizados diariamente estão agora conectados a internet. Este fenômeno vem mudando o modo em que vivemos, trabalhamos e interagimos. Isto cria, não apenas oportunidades tecnológicas para cidades inteligentes, mas também oportunidades de iteração entre os habitantes (SALIM e HAQUE, 2015). Contudo, o *design*, desenvolvimento e aplicação de projetos de Computação Ubíqua em ambientes urbanos, fora de áreas controladas de pesquisa, é desafiador devido à complexidade dos ambientes urbanos e das pessoas que os habitam.

Um conceito vital para a Computação Ubíqua é a questão da sensibilidade ao contexto. A possibilidade de uma aplicação conseguir coletar dados sobre a situação atual do ambiente na qual opera, e a partir do conhecimento do usuário, ser capaz de adaptar seu funcionamento.

2.2 Sensibilidade ao Contexto

Para Schilit e Theimer (1994) uma aplicação sensível ao contexto é aquela que é capaz de perceber e reagir a mudanças no ambiente onde o usuário está situado. Nesta visão, contexto está associado diretamente à localização geográfica do usuário e auxilia a aplicação a localizar pessoas e equipamentos computacionais próximos.

Segundo Brown, Bovey e Chen (1997) seria importante se na área da Ciência da Computação pudéssemos classificar as aplicações de maneira fácil. Mas não é tão simples. Assim, aplicações sensíveis ao contexto se misturam a outros tipos de aplicações e alguém poderia argumentar que qualquer aplicação é sensível ao contexto, caso leve em consideração o usuário. Porém, na prática apenas aquelas que possuem seu funcionamento orientado principalmente pelo contexto atual do usuário podem ser classificadas como sensíveis ao contexto.

Brown, Bovey e Chen (1997) ainda classificam este tipo de aplicações em duas categorias: contínuas e discretas. Em aplicações contínuas as informações apresentadas ao usuário são atualizadas constantemente. Um exemplo citado é um sistema de localização que exibe ao usuário uma seta indicando a direção que ele deve tomar para chegar a um determinado destino. A direção apontada pela seta varia conforme o usuário se movimenta, porém é exibida continuamente. Já em aplicações discretas, categoria onde a maioria das aplicações se encaixa, informações separadas estão associadas a diferentes contextos, tais informações são exibidas quando o usuário troca de contexto.

Para Schilit, Adams e Want (1994), aplicações sensíveis ao contexto se adaptam de acordo com a localização do uso, grupos de pessoas e hosts próximos e dispositivos acessíveis, bem como as mudanças que ocorrem nesses grupos. Segundo eles, os aspectos importantes para o contexto são: onde você está, com quem você está e quais são os recursos que estão próximos.

Dey, Abowd e Wood (1998) definem contexto como qualquer informação sobre o usuário ou ambiente que possa ser utilizada para melhorar a experiência do usuário. Isto inclui dados com os quais o usuário está trabalhando, hora do dia, localização do usuário, estado emocional do usuário, ambiente social e objetos próximos.

Dey, Abowd e Salber (2001) tentam estabelecer uma definição operacional de contexto, que ajude a determinar exatamente o que ele é. Desta maneira será possível estabelecer quais são as características comuns de contextos, que podem ser suportadas por abstrações e ferramentas. A definição final proposta pelos autores é a seguinte: “qualquer informação que possa ser usada para caracterizar a situação de entidades que são consideradas relevantes para a interação entre um usuário e uma aplicação, incluindo o próprio usuário e a aplicação. Contextos são, tipicamente: a localização, identidade e o estado de pessoas, grupos e objetos físicos e computacionais”.

Ainda segundo Dey, Abowd e Salber (2001), a definição proposta é bastante genérica, isto se deve ao fato de sensibilidade ao contexto poder ser uma característica de qualquer aplicação. O foco é a utilizar tais informações para descoberta do objetivo do usuário. Como é difícil determinar diretamente quais são os objetivos do usuário, dados do contexto podem ajudar a inferir esta informação. Os autores propõem uma classificação das informações, baseada nas entidades envolvidas no contexto. As entidades mais identificadas como mais relevantes são: lugares, pessoas e coisas. Lugares são regiões de espaço geográfico como salas, escritórios, construções ou ruas. Pessoas podem ser tanto indivíduos quanto grupos, em um mesmo local ou dispersos. Coisas podem tanto ser objetos físicos quanto componentes de software.

Dey, Abowd e Salber (2001), também introduzem o conceito de quatro características essenciais para informações de contexto:

- Identidade: atribui um identificador único para uma entidade no domínio da aplicação;

- Localização: é mais do que simplesmente a posição atual da Entidade em um espaço bidimensional. Pode abranger também a dados como a direção e elevação ou qualquer informação que permita deduzir as relações espaciais entre as entidades.
- Atividade: refere-se a características da entidade vinculada ao contexto. Para lugares, podem ser informações como temperatura, luminosidade, umidade, ou nível de ruído. Para pessoas pode significar o estado emocional, seus sinais vitais ou atividades. Para software, podem ser qualquer atributo do software que possa ser requisitado, como por exemplo uso de memória ou processador.
- Tempo: identifica o momento da ocorrência de cada evento ou captura de dados contextuais, permitindo a análise histórica destes eventos.

Segundo Knappmeyer, Kiani, *et al.* (2013), contexto é a informação sobre uma localização e seus atributos ambientais (por exemplo, nível de ruído, a intensidade da luz, temperatura e movimento) e as pessoas, dispositivos, objetos e agentes de software estão contidos nesta localização. Contexto também pode incluir as capacidades do sistema, serviços oferecidos e procurados, as atividades e tarefas em que as pessoas e entidades computacionais estão envolvidos, e seus papéis situacionais, crenças e intenções. Ciencia de contexto é um dos elementos fundamentais para facilitar o apoio pró-ativo aos usuários em sua situação atual.

Knappmeyer, Kiani, *et al.* (2013) ainda afirmam que, os usuários não devem definir a sua situação de forma explícita, utilizando dispositivos de entrada lentos e não intuitivos, esta infromação deve ser implicitamente reconhecida pelo ambiente inteligente. A identidade do usuário e sua localização são os parâmetros de contexto mais amplamente utilizados em vários serviços baseados em localização.

Com base no exposto, é adotada nesta dissertação a definição de contexto apresentada por Dey, Abowd e Salber (2001), onde contexto é “qualquer informação que possa ser usada para caracterizar a situação de entidades relevantes para a interação entre um usuário e uma aplicação, e que em geral é composto por dados que descrevem a situação, identidade e localização espaço-temporal de pessoas, grupos e objetos físicos ou computacionais”.

A capacidade de armazenar os dados contextuais vinculados a uma entidade cria o conceito de histórico de contextos, que é discutido na próxima seção.

2.3 Históricos de Contextos

Em uma das primeiras tentativas de explorar contextos armazenados de usuários, Clarke e Driver (2004) propuseram a criação de uma estrutura chamada trilha e a utilizaram para captura de atividades diárias de um usuário. De forma geral, uma trilha é uma coleção de localizações, associada com informações sobre as elas e uma ordem recomendada de visita a tais localizações.

Mayrhofer (2005) apresenta o tópico de predição de contexto, como uma possibilidade de explorar históricos de contexto. Ainda afirma que históricos de contexto, especialmente quando armazenados por longos períodos oferecem muitas possibilidades de melhoria para serviços providos por sistemas. Tais possibilidades incluem a inferência de ações de usuários, sejam elas atuais ou passadas. Entretanto, a predição de contextos futuros baseado em contextos armazenados a priori é comumente reconhecida como o desafio definitivo na exploração de históricos.

Segundo Hong, Suh, *et al.* (2009), a coleção de contextos passados e ações de usuários nesses contextos têm sido conhecidas como histórico de contexto. Existem inúmeras

possibilidades de aperfeiçoamento de serviços oferecidos com esta abordagem. Quando histórico de contexto pode ser utilizado por um sistema, pode haver a extração de conhecimento destes históricos e serviços inteligentes podem oferecer aos usuários inúmeras possibilidades personalizadas. Hong, Suh, *et al.* (2009) acreditam que uma das limitações dos sistemas sensíveis a contexto anteriores era o fato de que só consideravam o contexto atual. Um cenário de exemplo é citado, onde um usuário costuma assistir o noticiário na televisão, todos os dias às 21h. Em um ambiente inteligente, o sistema deve permitir que o usuário assistisse o noticiário, de acordo com o seu padrão.

Nurmi, Martin e Flanagan (2005) afirmam que o papel da predição de em sistemas sensíveis ao contexto é frequentemente visto como capacitador de execução automática de serviços e aplicações. Para expandir esta visão novas maneiras de modificar as iterações entre usuários e aplicações são necessárias. Uma estratégia para tal é o uso de pró-atividade em sistemas, baseando-se em históricos de contextos armazenados. O que vai ao encontro da visão de Satyanarayanan (2001), que aponta a necessidade de pró-atividade para tornar a computação ubíqua mais eficaz.

Recentemente, há exemplos de aplicações que utilizam informações de dados de históricos de contexto. Rosa, Barbosa, *et al.*, (2015) propõem *MultCComp*, um sistema de gerenciamento de competências de empregados, que utiliza seus dados de contexto passados e do presente para auxiliar no desenvolvimento de suas competências.

Barbosa, Martins, *et al.*, (2016) apresentam *TrailTrade*, um sistema sensível a trilhas que provê suporte à comércio. O modelo utiliza trilhas de revendedores para promover oportunidades de negócio. Gomes Cardoso, Mota, *et al.*, (2016) propõem Vulcanus 2.0, um sistema de recomendação para acessibilidade, que utiliza análise de similaridade para comparar trilhas de usuários, com aperfeiçoamento em cenários de caso médio.

Com base no exposto, será adotado nesta dissertação o termo “histórico de contextos” para se referir à lista cronológica de contextos que foram visitados por uma entidade identificada no passado.

A seção a seguir apresenta conceitos de *data mining* e *KKD* e discute técnicas comumente utilizadas para análise de dados de históricos de contexto.

2.4 Data Mining e KDD

Segundo Han, Kamber e Pei (2011) *data mining* pode ser vista como o resultado natural da evolução da tecnologia de informação. A indústria de bases de dados evoluiu o desenvolvimento de algumas funcionalidades criticas: (1) coleta de dados e criação de bases de dados; (2) gerenciamento de dados, incluindo armazenamento, recuperação e processamento de transações; (3) análise avançada de dados, incluindo *data warehousing* e *data mining*.

Durante a década de 60 a tecnologia de bases de dados evoluiu sistematicamente, partindo de processamento de arquivos primitivos para sistemas de bases de dados sofisticados e poderosos. Na década de 80, os esforços de pesquisa se voltaram para sistemas avançados de bancos de dados. Esses sistemas incorporavam novos e poderosos modelos de dados, como: relacional estendido, orientado a objetos, objetos relacionais e modelo dedutível. Além disso, novos bancos de dados orientados a aplicações também surgiram, como: bases de dados espaciais, temporais, multimídia, científicas, entre outras. Problemas relacionados com a

distribuição, diversificação e compartilhamento de dados foram estudados extensivamente (HAN, KAMBER e PEI, 2011).

Os avanços da indústria de hardware de computadores do final da década de 80 levou a grande disponibilidade de computadores poderosos e de preço acessível, equipamentos avançados de coleta de dados e mídias de armazenamento. Estes fatores permitiram a criação de inúmeras bases de dados e repositórios de informações. Dados agora podiam ser armazenados em inúmeros tipos de bases e repositórios diferentes. Essa grande quantidade de dados unida a uma necessidade de ferramentas de análise de dados mais poderosas foi descrita como: situação rica em dados, mas pobre em informação. Tais situações trazem a necessidade da evolução de ferramentas de data mining que possam transformar dados em conhecimento (HAN, KAMBER e PEI, 2011).

Segundo Fayyad, Piatetsky-Shapiro e Smyth (1996), *data mining* é apenas um dos nomes historicamente dados à noção de busca por padrões úteis em dados. Na visão dos autores, o termo *Knowledge Discovery in Databases*, se refere ao processo geral de busca por conhecimento útil em dados, enquanto *data mining* é a aplicação de algoritmos específicos para extração de padrões dos dados. Além disso, afirmam que KDD evoluiu e continua evoluindo da intersecção dos campos de pesquisa de aprendizado de máquina, reconhecimento de padrões, estatísticas de bases de dados, AI, aquisição de conhecimento para sistemas especialistas, visualização de dados e computação de alto desempenho.

Fayyad, Piatetsky-Shapiro e Smyth (1996) ainda definem *data mining* como um paço dentro do processo de KDD, que consiste na aplicação de algoritmos de análise e descoberta a fim de produzir padrões ou modelos em dados. Enquanto KDD é definido como um processo não trivial de identificação de padrões validos, incomuns, potencialmente úteis e completamente entendíveis em dados.

Fayyad, Piatetsky-Shapiro e Smyth (1996) dividem as funções de modelagem mais comuns em *data mining* nas seguintes categorias:

- Classificação: mapeia ou classifica um item em uma de várias classes de categorias predefinidas;
- Regressão: mapeia um item a uma variável de predição que satisfaça a função que mais se ajuste aos dados observados anteriormente;
- Agrupamento: mapeia ou classifica um item em uma de várias classes de categorias que provém de dados. Classes de categorias são definidas utilizando métricas de similaridade ou matrizes de probabilidade;
- Sumarização: provê uma descrição simples para um subconjunto de dados;
- Análise de links: determina relações entre campos nas bases de dados, por exemplo, associações de regras para descrever quais itens são mais comprados com outros itens em lojas;
- Análise de sequencias: modela padrões de sequencias em dados com dependência de tempo;

Há exemplos de uso de técnicas de *data mining* para classificação de dados contextuais. Este tipo de análise geralmente busca classificar dados provenientes de sensores para identificação de atividades ou contextos de maiores níveis. Técnicas de aprendizado supervisionado são amplamente utilizadas neste tipo de problema. Para tal, inicialmente dados de treinamento são coletados e rotulados de acordo com os resultados esperados. Em seguida é

derivada uma função que pode gerar os resultados esperados usando os dados de treinamento. Árvore de decisão é uma destas técnicas, que cria uma estrutura de árvore com base nos registros do conjunto de treinamento. A partir desta árvore, pode-se classificar a amostra desconhecida. Esta técnica é utilizada para reconhecimento de atividades e (HUANG, WU, *et al.*, 2008; RIBONI e BETTINI, 2009).

Redes Bayesianas recebem este nome por ser baseado no teorema de probabilidade de Bayes (VAPNIK e VAPNIK, 1998), seu processo baseia-se na construção de uma rede que represente uma distribuição de probabilidades, em um determinado domínio de aplicação. Em seguida, deve-se especificar como a distribuição de probabilidades de cada nó será representada e, por fim, realiza-se uma estimativa das probabilidades condicionais com a base de dados, sendo importante observar valores de variáveis que devem ser considerados ou desconsiderados. As redes bayesianas são comumente usadas na combinação de informações de um grande número de sensores para inferência de contextos de alto nível (KO e SIM, 2008; PARK, OH e CHO, 2011).

Support Vector Machines (HEARST, DUMAIS, *et al.*, 1998) é outro método de aprendizado supervisionado amplamente utilizado no reconhecimento de atividades de pacientes em ambientes hospitalares (DOUKAS, MAGLOGIANNIS, *et al.*, 2007) e ambientes de casas inteligentes (BRDICZKA, CROWLEY e REIGNIER, 2009).

Já técnicas de aprendizado não supervisionado, são capazes de identificar estruturas escondidas nos dados sem o uso de dados rotulados. Também há exemplos de utilização deste tipo de técnica em dados contextuais. O método de classificação baseado no vizinho mais próximo, ou KNN (BOHM e KREBS, 2004), utiliza a técnica de descoberta baseada em instâncias, que requer uma medida de distância para classificar os objetos. Tal método é comumente utilizado para classificar dados de sensores de posicionamento indoor (LIN e LIN, 2005). Métodos de redes neurais não supervisadas como o Kohonen Self-Organizing Map, ou KSOM (KOHONEN, 1998), são utilizados para classificação de dados de sensores em tempo real (VAN LAERHOVEN, 2001).

Regras de associação (HIPP, GUNTZER e NAKHAEIZADEH, 2000) também são amplamente utilizadas para conversão de dados de sensores para contextos mais significativos. Exemplos podem ser encontrados em detecção de eventos (BARBERO, DAL ZOVO e GOBBI, 2011) e classificação de dados de sensores (KONSTANTINOU, SOLIDAKIS, *et al.*, 2007).

Modelos Markovianos, ou HMM (RABINER e JUANG, 1986), modelam processos que tipicamente emitem sinais observáveis em cadeias de estados. Tais modelos são utilizados no reconhecimento de atividades em ambientes inteligentes (BRDICZKA, CROWLEY e REIGNIER, 2009).

Normalmente, técnicas de *data mining* empregadas em dados contextuais buscam obter informações contextuais mais significativas a partir de dados de sensores. Um tipo de técnica pouco utilizado em aplicações relacionadas a dados contextuais é a mineração de padrões sequenciais, também chamada de mineração de sequências frequentes. Esta técnica trata de dados representados como sequências ordenadas cronologicamente, formadas por conjuntos de itens classificados (LI, XU, *et al.*, 2005). Padrões sequenciais são aplicáveis em uma grande variedade de domínios, uma vez que diversas aplicações possuem históricos organizados cronologicamente (MASSEGLIA, TEISSEIRE e PONCELET, 2005). Dados de históricos de contextos apresentam a mesma característica, por apresentarem os diferentes contextos visitados por entidades no passado. Deste modo, a mineração de padrões sequenciais em históricos de contextos tem um grande potencial de aplicações.

Com base no exposto, o modelo proposto fará uso de ferramentas de *data mining* para realizar as tarefas relativas à descoberta de padrões. Mais especificamente o tipo de padrões suportados pelo modelo é padrões sequenciais, também conhecidos como sequências frequentes.

A seção a seguir apresenta a definição do problema da descoberta de padrões sequenciais e algumas das estratégias comumente utilizadas para sua mineração.

2.5 Reconhecimento de padrões sequenciais

Fayyad, Piatetsky-Shapiro e Smyth (1996) apresentam duas definições importantes:

- Primeiramente definem dados como um conjunto de fatos, por exemplo, registros em uma base de dados.
- Em seguida, eles definem padrão como uma expressão em alguma linguagem, descrevendo um subconjunto de dados ou um modelo, aplicável ao subconjunto. Consequentemente extrair um padrão também implica em alinhar um modelo aos dados, encontrar estrutura nos dados ou, no geral, fazer qualquer descrição de alto nível de um conjunto de dados.

Segundo Gupta e Han (2012), métodos de reconhecimento de padrões em dados sequenciais são aplicáveis a um grande número de domínios. Dados sequenciais são onipresentes, estão em todos os tipos de aplicações e domínios. Citam como exemplo, dados de consumidores, dados de tratamentos médicos, dados relacionados a desastres naturais, dados de aplicações científicas ou de engenharia de processos, dados de ações de bolsa de valores, ligações telefônicas, sequencias de DNA do genoma, etc. Técnicas de mineração de padrões sequenciais têm sido extensivamente utilizadas para analisar dados sequenciais e identificar padrões. Tais padrões são utilizados para melhorar a eficiência de sistemas de recomendação, que agora podem recomendar baseado em padrões observados, auxiliar em predições, aumentando a usabilidade de sistemas, detectar eventos e, de forma geral, auxiliar a tomada de decisões estratégica em produtos.

Gupta e Han (2012) definem padrão como uma subsequência de dados sequenciais, que possui uma métrica de suporte. Tal métrica se baseia no numero de vezes que a subsequência está contida nas sequencias de uma base de dados. Uma subsequência cujo suporte seja maior do que uma variável de limite é chamada de padrão frequente. Um algoritmo de mineração de padrões frequentes deve:

- Encontrar o conjunto completo de padrões, quando possível, que satisfaçam o suporte mínimo estabelecido;
- Ser eficiente e escalável, realizando o menor número possível de leituras da base de dados;
- Ter a capacidade de incorporar vários tipos de restrições configuradas pelo usuário;

Esta dissertação adota o termo “padrão”, segundo a definição de Gupta e Han (2012): “uma subsequência de dados sequenciais, que possui uma métrica de suporte. Tal métrica se baseia no numero de vezes que a subsequência está contida nas sequencias de uma base de dados”.

As seções seguintes descrevem as algumas das estratégias mais utilizadas para mineração de padrões sequenciais (MOTEGAONKAR e VAIDYA, 2014), (MOONEY e RODDICK, 2013).

2.5.1 Métodos *Apriori*

A partir da propriedade *Apriori* das sequencias, temos o fato de que, se uma sequencia S não é frequente, então nenhuma das sequências que contém S podem ser frequentes. Por exemplo, se a sequencia $< ab >$ não é frequente, então nenhuma das sequencias $< xab >, < yab >, < abz >$ são frequentes.

O algoritmo *Generalized Sequential Pattern*, apresentado em (SRIKANT e AGRAWAL, 1996), busca todas as sequencias candidatas de tamanho 1, e as ordena de acordo com o seu valor de suporte, ignorando aquelas que não satisfazem o suporte mínimo. A partir daí, o algoritmo repete o processo, verificando todas as sequencias de tamanho 2, 3, 4 e sucessivamente até o tamanho máximo de sequencias que se deseja minerar.

O ponto negativo desta abordagem é que o algoritmo gera um vasto conjunto de sequencias candidatas. Além disso, múltiplas leituras do banco de dados são necessárias, o que o torna ineficiente para mineração de longas sequencias.

2.5.2 SPADE

Sequential Pattern Discovery using Equivalent Class (SPADE), foi apresentado por Zaki (2001). O algoritmo usa propriedades combinatórias de decomposição para reduzir o problema original em subproblemas menores. Os subproblemas podem ser resolvidos de forma independente em memória.

Esta estratégia geralmente requer apenas três leituras na base de dados, ou apenas uma leitura caso haja pré-processamento.

Primeiramente o algoritmo mapeia a base de dados em formato de conjunto de itens $< SID\ (ID\ da\ Sequencia), EID(ID\ do\ Evento) >$. A busca por padrões é realizada expandindo um item por vez através da geração de candidatos usando *Apriori*.

2.5.3 FreeSpan

O *FreeSpan*, apresentado por Han, Pei, *et al.*, (2000), utiliza itens frequentes para projeção de bases de dados de sequencias em conjuntos de bases de dados menores e então minera cada um destes bancos para encontrar os padrões frequentes.

Primeiramente o *FreeSpan* realiza uma leitura completa da base de dados de sequencias, coleta a métrica de suporte para cada item e encontra o conjunto de itens frequentes. Uma lista é criada com os itens frequentes em ordem descendente na forma $< Item: suporte >$, por exemplo, $[a: 5, b: 5, c: 4, d: 4, e: 4, f: 3]$. De acordo com a lista, o conjunto completo de padrões sequenciais pode ser dividido em seis subconjuntos distintos: (1) os que contêm apenas o item a, (2) os que contêm o item b, mas nenhum item após b na lista, (3) os que contêm o item c,

mas nenhum item após *c* na lista, e assim por diante. Por fim, o conjunto (6) com aqueles que contêm f. Os subconjuntos de padrões sequenciais podem ser minerados com o uso de bases de dados projetadas.

Este processo é realizado recursivamente nas bases de dados projetadas. O maior custo desta abordagem é lidar com o grande número de bases de dados projetadas.

2.5.4 CloSpan

Closed Sequential Pattern Mining (*CloSpan*), apresentado por Yan, Han e Afshar (2003) é um método que tem como objetivo apenas a identificação das maiores sequencias. Ou seja, o algoritmo minera apenas as frequências que não estão contidas em sequencias maiores, apenas sequencias fechadas.

A mineração de sequencias fechadas reduz expressivamente o número de sequencias candidatas redundantes e aumenta a eficiência. O *CloSpan* é similar ao algoritmo *PrefixSpan* (PEI, HAN, *et al.*, 2001), que também usa a técnica de projeção, como o *FreeSpan*, porém apenas examinando sequencias prefixadas.

2.5.5 PrefixSpan

O *PrefixSpan* (*Prefix-projected Sequential Pattern Mining*), apresentado por Han, Pei, *et al.* (2001) explora a projeção de prefixos na mineração de padrões sequenciais. Sua ideia principal é que, ao invés de projetar sequências de bancos de dados considerando-se todas as ocorrências possíveis de subsequências frequentes, a projeção seja baseada apenas em prefixos frequentes, já que qualquer subsequência crescente pode sempre ser encontrada pelo crescimento de um prefixo crescente.

PrefixSpan é um algoritmo eficiente para a mineração de sequências (que preserva a ordem temporal de ocorrência), mas que é incapaz de gerar regras de associação e, portanto, não gera regras de implicação com um possível valor para a força do padrão encontrado.

2.5.6 FP-Growth

O *FP-Growth* (*Frequent Pattern Growth*), apresentado por Han, Pei e Yin (2000), é um método eficiente e escalável para a mineração de padrões frequentes sejam eles curtos ou longos. Este algoritmo extrai conjuntos frequentes sem geração de candidatos.

A estratégia utilizada é dada através de uma estrutura de árvore chamada *FP-Tree* para representar a base de dados minerada de forma compactada. Cada nó da árvore representa um item na base de dados. Os nós formam ramos que representam conjuntos de itens presentes em uma ou mais transações. O algoritmo funciona utilizando a estratégia dividir e conquistar, sendo necessárias duas leituras completas do banco de dados.

2.6 Considerações sobre o Capítulo

Neste capítulo foram apresentados os conceitos teóricos que serviram de embasamento para a pesquisa realizada. Inicialmente foi apresentada a área de pesquisa da Computação Ubíqua e seus desafios. Em seguida foi introduzido o conceito de aplicações sensíveis ao contexto e sua importância para a Computação Ubíqua. Tais conceitos (Computação Ubíqua, Contextos e Sensibilidade a Contexto) são uma base importante para a construção do modelo proposto nesta dissertação.

Além disso, foram discutidos os conceitos de trilhas, sequencias e históricos de contextos e como eles podem ser utilizados para melhorar a sensibilidade ao contexto. Neste sentido, foi adotado o emprego do termo “histórico de contextos” para se referir a lista cronológica de contextos visitados por uma identidade no passado, tema que esta dissertação aborda.

Em seguida foram apresentados os conceitos de mineração de dados e descoberta de conhecimento em bases de dados. Alguns dos métodos mais utilizados para mineração de dados contextuais foram discutidos. “Padrões sequenciais” foram identificados como tipo de padrão suportado pelo modelo proposto nessa dissertação. Deste modo o emprego do termo “padrão” foi adotado para identificar “uma subsequência de dados sequenciais, que possui uma métrica de suporte”.

Por fim, foram estudadas as técnicas mais comuns utilizadas para mineração de padrões sequenciais. Tais técnicas são utilizadas pelo modelo proposto para a extração e acompanhamento de padrões.

3 TRABALHOS RELACIONADOS

Neste capítulo são apresentados os trabalhos encontrados que abordam o tema de descoberta de padrões em sequências de dados contextuais, seguindo a metodologia descrita na seção 3.1.

São descritos a metodologia e os critérios para busca e seleção dos trabalhos. Por fim são apontadas as contribuições deste modelo em relação aos trabalhos analisados. O capítulo está dividido em nove seções. A primeira seção apresenta um panorama das pesquisas relacionadas, bem como a metodologia adotada para a escolha dos trabalhos. As seções de 3.2 a 3.8 descrevem os trabalhos selecionados. Por fim, na seção 3.9 é realizada uma análise comparativa entre os trabalhos.

3.1 Metodologias para a escolha dos trabalhos

A pesquisa foi realizada considerando a definição de contexto apresentada por Dey, Abowd e Salber (2001), onde contexto é "qualquer informação que possa ser usada para caracterizar a situação de entidades relevantes para a interação entre um usuário e uma aplicação, e que em geral é composto por dados que descrevem a situação, identidade e localização espaço-temporal de pessoas, grupos e objetos físicos ou computacionais". A partir desta definição foi realizada uma pesquisa por artigos que abordassem o tema da descoberta de padrões em sequências de dados que pudessem ser classificados como dados contextuais.

Para classificar os dados avaliados por cada trabalho como contextuais foram levadas em consideração as seguintes categorias:

- Tempo: os modelos apresentados deve considerar informação temporal para cada situação das entidades a serem acompanhadas;
- Identidade: os dados avaliados devem conter informação que proporcione a identificação de entidades acompanhadas;
- Situação e/ou Localização: os dados avaliados pelos trabalhos devem conter pelo menos um dos seguintes: (1) informação sobre a localização das entidades, por meio coordenadas geográficas ou em ambientes fechados; (2) informação sobre a situação da entidade, podendo descrever um estado ou atividade.

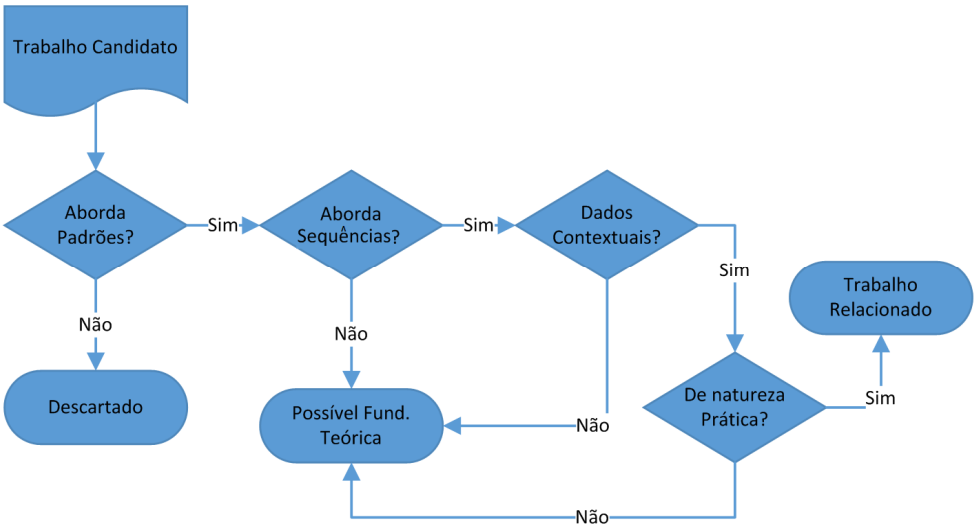
As palavras-chave utilizadas para a pesquisa foram: "*Ubiquitous Computing*", "*Pattern Recognition*", "*Sequential Patterns*" e "*Data Mining*". A pesquisa foi realizada nas bases de dados de pesquisas EBSCOhost[1] e portal de periódicos da CAPES[2].

O processo completo de aplicação dos critérios de seleção dos trabalhos relacionados está ilustrado na Figura 1. Inicialmente foram selecionados os trabalhos que abordassem o tema de descoberta de padrões, trabalhos sem este foco foram descartados, e trabalhos que abordavam padrões de forma geral e não especificamente sobre padrões em sequências de informações contextuais foram considerados como possíveis referências para a fundamentação teórica.

[1] Disponível em: https://www.ebscohost.com/

[2] Disponível em: http://periodicos.capes.gov.br/

Figura 1 – Fluxograma dos Critérios de Seleção dos Trabalhos Relacionados



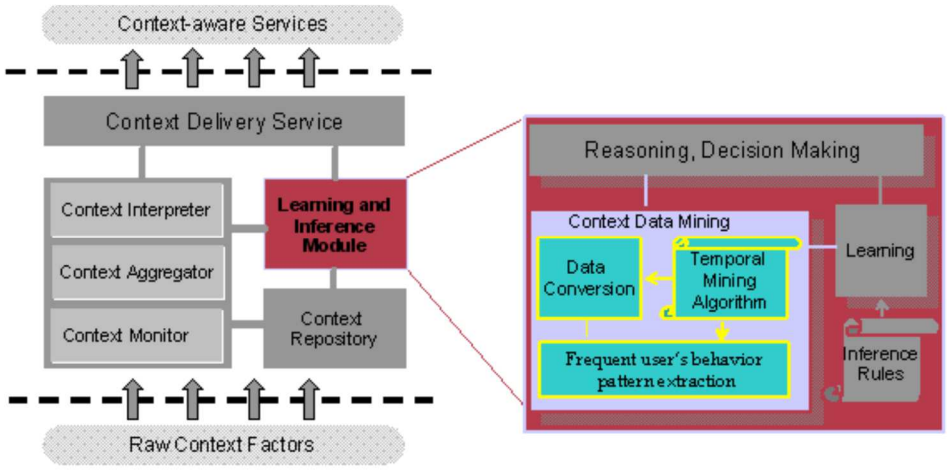
Fonte: Elaborado Pelo Autor

Como o objetivo final deste trabalho é de natureza prática, como último critério de seleção, os trabalhos relacionados deveriam apresentar um modelo que fosse diretamente implementável, a fim de permitir a comparação com o modelo desenvolvido. Isto inclui, mas não se restringe a, algoritmos, estruturas de dados, *frameworks*, sistemas e propostas de arquiteturas. Foram descartados como trabalhos relacionados de natureza teórica. As seções seguintes resumem os trabalhos relacionados que atenderam a estes critérios.

3.2 Efficient Mining of User Behaviors by Temporal Mobile Access Patterns

Lee, Paik, *et al.*, (2007) propõem um modelo para construção de padrões de comportamento de usuários, utilizando regras de associação temporais em sistemas de agentes móveis. O modelo proposto por Lee, Paik, *et al.*, (2007) pode ser visualizado na Figura 2, utiliza o *middleware* XML-ECDM (PAIK, DONG e KIM), uma plataforma inteligente utilizando mineração de arquivos XML. Primeiramente, dados de contexto são monitorados, agregados, interpretados e finalmente, armazenados. Os dados de contextos armazenados ficam disponíveis para o modulo de inferência e aprendizado. O objetivo deste módulo é dar suporte ao processo de tomada de decisão e adaptação do *middleware*, porém Lee, Paik, *et al.*, (2007) afirmam que a precisão pode ser afetada por inundações de informações de contexto sem valor ou escondidas.

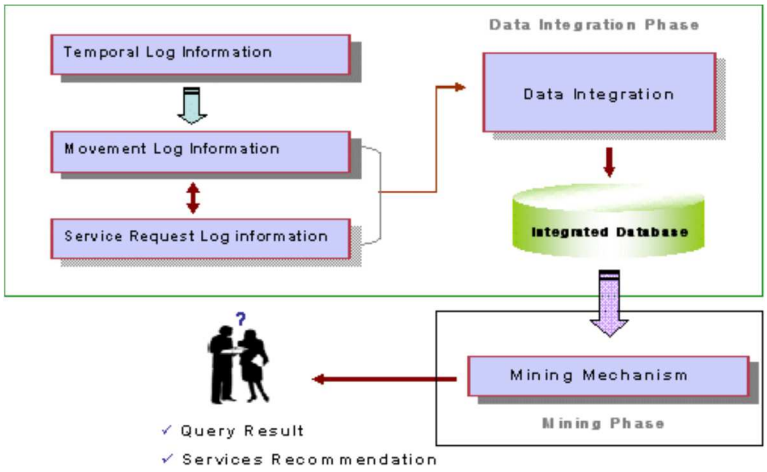
Figura 2 - Modelagem proposta do *middleware* e módulo de mineração



Fonte: Lee, Paik, *et al.*, (2007)

A Figura 3 ilustra os detalhes do módulo de mineração de contextos proposto pelos autores. O fluxo de execução do sistema é dividido em duas fases. Primeiramente os dados de diferentes sistemas são integrados. Assim, *logs* de movimento do usuário e de acesso a serviços são coletados e integrados em um *Dataset* único para um acesso mais eficiente.

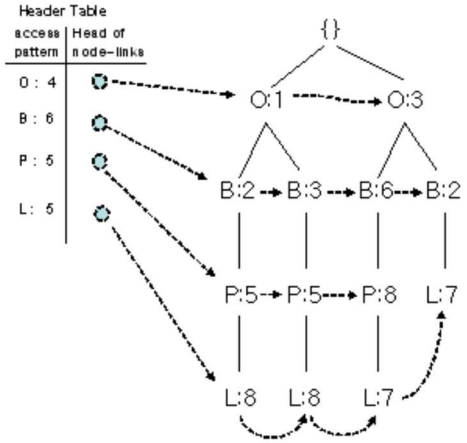
Figura 3 – Módulo de mineração de contextos



Fonte: Lee, Paik, *et al.*, (2007)

A segunda fase é onde ocorre a mineração, utilizando uma técnica de descoberta de padrões de acesso chamada *T-Map*, que utiliza uma tabela de cabeçalhos e busca agregar os padrões de forma compacta em memória utilizando uma estrutura de árvore, como pode ser observado na Figura 4. O trabalho não apresenta nenhum experimento para validação do modelo, os autores apenas afirmam que o modelo apresentado tem melhor consumo de memória e realiza menos consultas ao banco de dados do que algoritmos como o *Apriori*, uma das estratégias mais populares para busca de conjuntos frequentes.

Figura 4 – T-Map para dados de log integrados

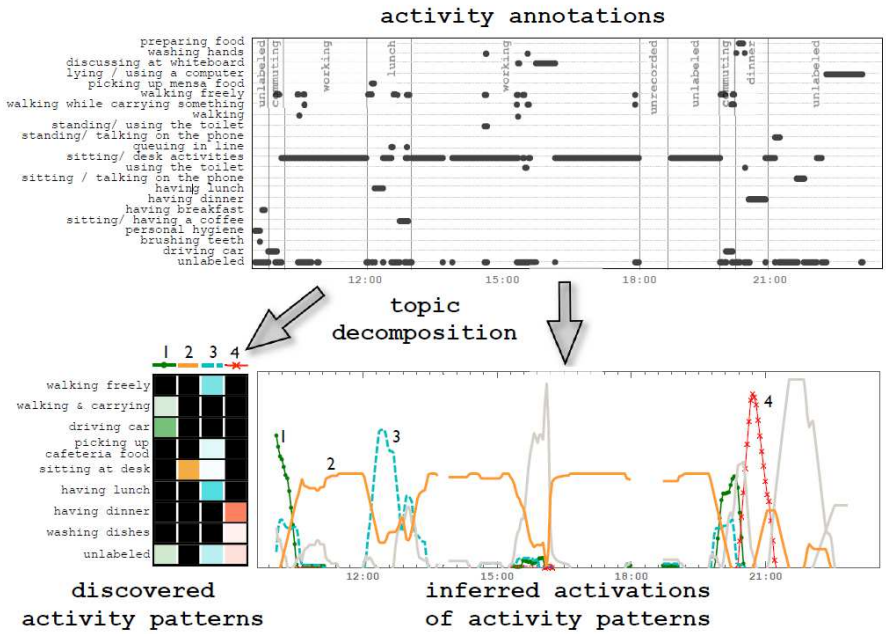


Fonte: (LEE, PAIK, *et al.*, 2007)

3.3 Discovery of Activity Patterns using Topic Models

Huynh, Fritz e Schiele (2008) apresentam um novo método para realizar o reconhecimento de rotinas diárias: uma combinação probabilística de padrões de atividades. Além disso, fazendo uso de modelos de tópicos probabilísticos (PTM), é possível realizar descoberta de padrões de atividades automaticamente.

Figura 5 – Estratégia do uso de PTM para descoberta de padrões de atividades



Fonte: Huynh, Fritz e Schiele (2008)

As atividades diárias podem ser segmentadas em diferentes níveis de granularidade. Para algumas aplicações já é suficiente reconhecer e classificar atividades simples a partir de uma pequena janela de tempo de leitura de sensores. Mas, se for realizada uma análise mais profunda para identificação de rotinas, uma estratégia mais natural seria agrupar estas

atividades em rotinas. Um modelo para reconhecer tais rotinas precisa ser capaz de capturar padrões que apresentam muitas atividades, podem ocorrer durante longos períodos e podem variar significativamente de acordo com suas instancias. Os autores apontam uma família de modelos probabilísticos que apresenta resultados excelentes para este tipo de aplicação, os modelos de tópicos. Assim, a modelagem das rotinas diárias é realizada utilizando PTM. A Figura 5 ilustra o resultado dessa abordagem, a parte superior apresenta as anotações das atividades simples, enquanto a parte inferior demonstra a composição de rotinas a partir da utilização de PTM.

Para avaliação do modelo, Huynh, Fritz e Schiele (2008) gravaram o dia a dia de uma pessoa por um período de 16 dias através do uso de dois sensores vestíveis. O primeiro sensor foi carregado no seu bolso direito e o segundo no seu pulso direito. Ao usuário também foi solicitado que realizasse anotação das atividades que estavam sendo realizadas. As atividades anotadas foram filtradas e organizadas em 34 anotações discretas. O *Dataset* gerado contendo as atividades anotadas de 7 dos 16 dias passou pelo processo de mineração de tópicos para o reconhecimento de padrões de atividades e foi avaliado e demonstrou que os padrões descobertos correspondem a comportamentos de alto nível do usuário que estavam altamente correlacionado com sua rotina. Além disso, os padrões eram baseados em atividades significativas, então um set destas atividades é legível e facilmente compreensível para uma pessoa analisar.

3.4 A Unified Framework for Activity Recognition-Based Behavior Analysis and Action Prediction in Smart Homes

Fatima, Fahim, *et al.*, (2013) propõem um framework para apoiar os habitantes de casas inteligentes. Seu principal objetivo é superar a limitação dos métodos existentes, através da introdução de uma estrutura unificada para análise do comportamento de habitantes e reconhecimento de atividade a fim de apoiar habitantes no desempenho de suas tarefas diárias e prestar serviços personalizados adaptados às suas necessidades.

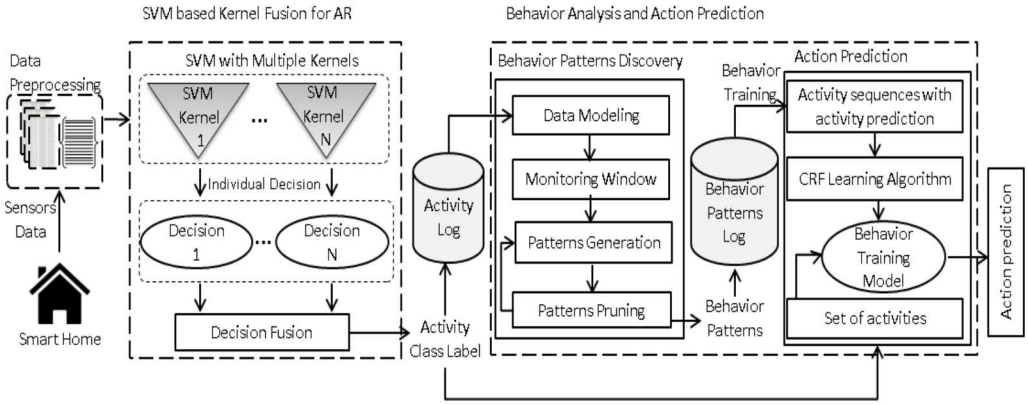
Na estratégia proposta pelos autores, uma atividade é definida como um conjunto de sensores que são ativados em um determinado tempo quando uma tarefa é executada em um ambiente de casa inteligente. Pode ser reconhecida entre dados coletados de sensores e a notação pode ser tanto de nível micro (ex.: leitura de um livro) quanto de macro (ex.: atividade de lazer). O framework é composto por três módulos, como pode ser observado na Figura 6:

- *Data Preprocessing*: Dados coletados de sensores ubíquos baseados na iteração de pessoas são armazenados em *logs* de sensores e arquivos de anotação, com atributos como tempo de inicio, tempo de fim, id dos sensores e valores de leitura dos sensores. Para fins de reconhecer as atividades executadas um *Dataset* é pré-processado na forma de associações entre vetores resultantes da leitura de sensores e o tipo de atividade executada. Além disso, informações excessivas são removidas dos *logs* e arquivos de anotação.
- *SVM based Kernel Fusion for Activity Recognition: Support Vector Machine* (SVM) é um método de aprendizagem de máquina utilizado para classificar as atividades através da determinação de um conjunto de vetores de suporte e minimização da taxa de erro média. As atividades reconhecidas e classificadas são armazenadas em um repositório de *logs* de atividades e podem ser efetivamente utilizadas por

provedores de serviço para auxiliar os habitantes do ambiente adequadamente depois da análise de seu estilo de vida.

- *Behavior Analysis and Action Prediction:* Este módulo se subdivide em dois outros módulos. O módulo de *Behavior Patterns Discovery* busca identificar os conjuntos de ações que ocorrem frequentemente juntos. Para tal tarefa é utilizada a técnica de *Sequential Pattern Mining* (SPM). E por fim, o módulo de *Action Prediction* que busca a predição das próximas ações para conjuntos de atividades. Para predição, é utilizada a técnica de aprendizado de máquina chamada *Conditional Random Field* (CRF).

Figura 6 – Arquitetura do framework proposto



Fonte: Fatima, Fahim, *et al.*, (2013)

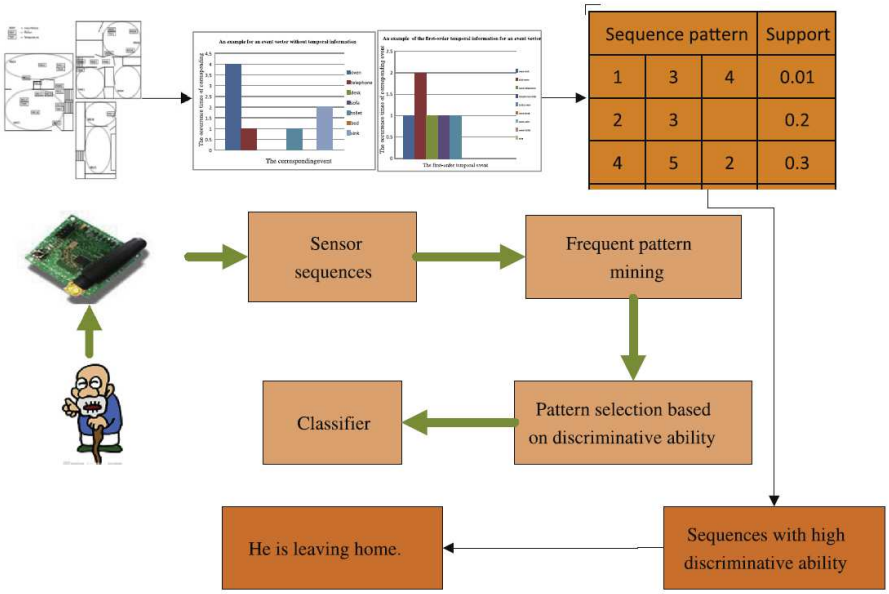
Os autores realizaram uma série de experimentos, utilizando dois *Datasets* coletados em projetos de ambientes *Smart Home*, contendo dados de sensores de movimento, temperatura e de abertura de portas. Para testar a eficiência da predição do framework, os *Datasets* utilizados são separados, deixando sempre um dia fora. Os dados do ultimo dia são utilizados para o teste, enquanto os dos dias restantes para o treino do modelo. Os resultados obtidos são então comparados a diferentes funções de distribuição probabilísticas. A precisão do modelo proposto (*kernel fusion*) foi de 94.11% e 92.70% para os *Datasets* de Milão e Aruba (Cook, Crandall, *et al.*, 2013), o que é significativamente melhor do que cada uma das funções avaliadas pelos autores.

3.5 Human activity recognition based on multiple order temporal information

Yin, Tian, *et al.* (2014) propõem um framework para reconhecimento automático de atividades humanas, com foco em sistemas de assistência a saúde. Os autores afirmam que o reconhecimento de atividades humanas é crítico para o desenvolvimento de sistemas robóticos de assistência. Deste modo, o reconhecimento de atividades se torna um passo importante para, por exemplo, auxiliar idosos a viver de forma independente no ambiente de casa.

Para tirar proveito das informações temporais de ordem múltipla, Yin, Tian, *et al.* (2014) propõem um *framework*, ilustrado na Figura 7, para classificação de atividades combinando técnicas de mineração de dados chamadas *Sequence Pattern Mining* (SPM) e *Feature Selection* (FS).

Figura 7 - Ilustração do funcionamento geral do framework



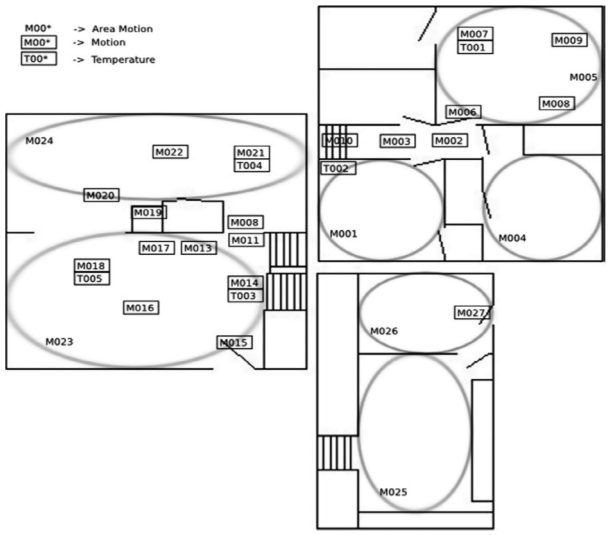
Fonte: Yin, Tian, *et al.* (2014)

Os autores apresentam a definição de *Activity* como sequencias de contextos, extraídas de sensores. Quando uma pessoa realiza suas atividades diárias, dados podem ser capturados por sensores distribuídos pelo ambiente. A Figura 8 ilustra o layout dos sensores em um ambiente. Os sensores com *IDs* iniciando com a letra "M" são sensores de movimento, enquanto os iniciados por "T" são de temperatura.

Um exemplo de *Activity* capturado pode ser observado na Figura 9. É uma atividade nomeada *Night_Wandering*, e inclui quatro informações: o *TimeStamp* da ocorrência, o ID do sensor (sid), o status do sensor (sst) e a descrição da ocorrência.

Yin, Tian, *et al.* (2014) também introduzem a definição de *Event*, que é composto pelo identificador do sensor e seu valor de leitura. Todos os sensores tem um id único e o valor de leitura segue a notação Sv, $Sv \epsilon \{1,2,3, \dots, M\}$. Onde M é o número máximo de leitura de um sensor. Deste modo é possível compor uma matriz contendo todas as possibilidades de leitura para todos os sensores do ambiente.

Figura 8 - layout de instalação de sensores em um ambiente



Fonte: Yin, Tian, *et al.* (2014)

Figura 9 - Um exemplo de sequencia de atividade

2009-06-10 03:20:59.087874	M006	ON	Night Wandering begin
2009-06-10 03:21:01.038931	M002	ON	
2009-06-10 03:21:03.001745	M002	OFF	
2009-06-10 03:21:03.092281	M006	OFF	
2009-06-10 03:21:04.000884	M002	ON	
2009-06-10 03:21:05.009842	M002	OFF	
2009-06-10 03:21:08.033939	M009	ON	
2009-06-10 03:21:10.027285	M009	OFF	Night Wandering end

Fonte: Yin, Tian, *et al.* (2014)

A segunda definição apresentada por Yin, Tian, *et al.* (2014) é a de *Multiple Order Temporal Information*: sequencias de eventos que compõem uma atividade podem possuir várias ordens. Tomando por exemplo os eventos 1, 2 e 3, as sequencias 1-2, 1-3, 2-3, 2-1, 3-1 são sequencias de primeira ordem. As sequencias 1-2-3, 3-2-1 são segunda ordem, e os autores consideram cada um dos eventos 1, 2, 3 como uma sequencia de ordem zero.

Segundo Yin, Tian, *et al.* (2014) informações temporais de ordem zero retêm algumas úteis de atividade, mas se a informação de relação temporal entre os eventos não é levada em consideração pode haver ambiguidade nos dados. Quando sequencias de primeira ordem são utilizadas para a mesma análise há redução na ambiguidade da informação.

Para extrair dados relevantes entre as informações temporais, os autores utilizam a técnica de *Sequence Pattern Minign* (SPM). Levando em consideração algumas características para as sequencias de atividades: (1) as sequencias coletadas são muito mais longas que as sequencias que compõem uma atividade; (2) há muito ruído, pois o status dos sensores pode ser afetado por condições ambientais; (3) pessoas podem realizar atividades de forma arbitrária,

com intervalos e concorrência; (4) a busca deve retornar as sequencias mais discriminatórias, e não os padrões mais frequentes.

Deste modo, Yin, Tian, *et al.* (2014) especificam o tamanho das sequencias a serem mineradas, para lidar com o problema (1). Para solucionar os problemas (2) e (3) mineração de padrões frequentes é utilizada para remover o ruído e extrair informações de relacionamento temporal entre os padrões regulares de atividades. E por fim, é proposta a utilização de *Feature Selection* para classificação de sequencias a serem consideradas relevantes.

O algoritmo proposto foi implementado e três diferentes *Datasets* foram utilizados nos experimentos. Os resultados obtidos foram comparados com outros algoritmos comumente utilizados para reconhecimento de padrões temporais como HMM. Os resultados do algoritmo proposto demonstraram aumento na precisão quando comparados a HMM.

3.6 Pattern-based causal relationships discovery from event sequences for modeling behavioral user profile in ubiquitous environments

O modelo proposto por Chikhaoui, Wang, *et al.*, (2014) baseia-se em análise de causalidade para construção de perfis de usuários baseados em comportamento. O modelo é genérico para ambientes externos ou internos, e introduz um novo método de otimização na profundidade de árvore probabilística de sufixo para detecção de padrões de comportamento significativos. Além disso, há a introdução de um novo método de classificação baseado em padrões significantes de comportamento em sequencias.

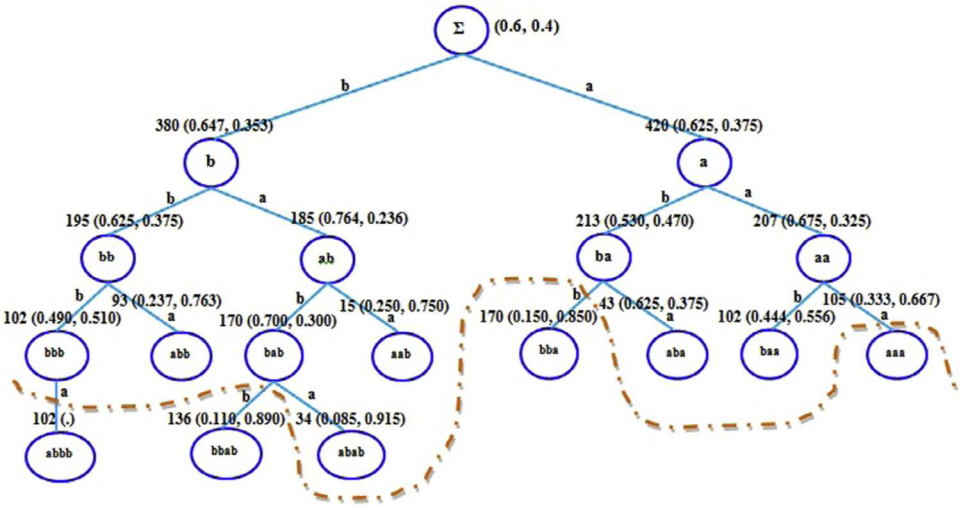
A partir de uma base de dados contendo sequencias, o objetivo é a descoberta de relacionamento de causalidade entre padrões significativos. O primeiro passo é a busca por padrões significativos de comportamento, porém antes de realizar esta busca há uma categorização das sequencias entre grupos similares.

Chikhaoui, Wang, *et al.*, (2014) apresentam as seguintes definições:

- Padrão significante: um padrão ρ, extraído de uma base de dados $\mathcal{D}$ que consiste em uma sequencia de eventos ordenados por tempo, onde o número de sequencias do padrão ρ em $\mathcal{D}$ seja maior ou igual a $\mathcal{R}$, onde $\mathcal{R}$ é um limitador especificado pelo usuário.
- Causalidade: Sendo $\mathcal{X}$ e $\mathcal{Y}$ duas variáveis aleatórias. $\mathcal{X}$ causa $\mathcal{Y}$ assumindo três condições: (i) $\mathcal{X}$ ocorre antes de $\mathcal{Y}$, (ii) $P(\mathcal{X}) \neq 0$, e (iii) $P(\mathcal{X}|\,\mathcal{Y}) > P(\mathcal{Y})$.
- Grafo de causalidade: um grafo de causalidade CG é um grafo direcionado, onde os nodos representam padrões significantes e os arcos representam as dependências causais entre os padrões.
- Perfil de usuário: é um conjunto finito de grafos de causalidade, onde cada grafo representa um comportamento do usuário.

Para realizar a descoberta de padrões significantes, Chikhaoui, Wang, *et al.*, (2014) utilizam uma técnica de mineração de dados chamada *Probabilistic Suffix Tree* (PST). Uma PST, ilustrada na Figura 10, é uma estrutura de árvore, que parte da raiz onde cada nodo é rotulado com um padrão de sequencia, acompanhado pelo numero de ocorrências do padrão em uma distribuição condicional de probabilidade.

Figura 10 – Árvore de sufixo probabilística (PST)



Fonte: Chikhaoui, Wang, *et al.*, (2014)

A extração de correlação entre os padrões significantes é realizada através da aplicação de técnicas de agrupamento (*Clustering)*. O método utilizado para a classificação de grupos é a medida de similaridade de Jaccard. Após a classificação, um perfil de usuário é extraído de cada *cluster* e um novo PST é construído para cada sequencia.

Para descoberta de relação de causalidade Chikhaoui, Wang, *et al.*, (2014) utilizam a estratégia *Transfer Entrophy* (TE) para inferir informações. A TE considera informações compartilhadas a partir de um histórico comum das variáveis utilizando probabilidade condicionada a transições.

Uma série de experimentos foi realizada, utilizando *Datasets* com informações de dois diferentes domínios: (i) atividades diárias de pessoas, e (ii) meios de transportes contendo coordenadas GPS. O modelo proposto foi comparado com modelos comumente utilizados para predição de atividades e identificação de usuários. Os resultados dos experimentos demonstram que o uso de relações causais pode melhorar significativamente a predição de atividades de usuários, alem de produzir perfis de usuário mais precisos.

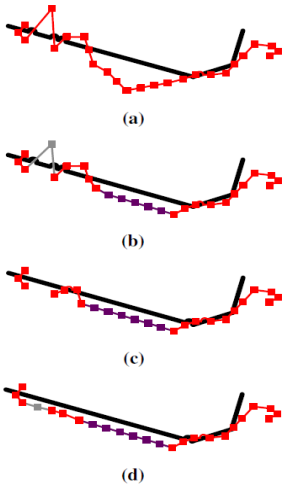
3.7 WTrack: HMM-based walk pattern recognition and indoor pedestrian tracking using phone inertial sensors

O trabalho proposto por Niu, Li, *et al.* (2014) é o WTrack, um sistema de rastreamento de pedestres aplicável a diversos ambientes fechados e capaz de lidar com ruídos de sensores, incluindo o balanço natural no caminhar, e distúrbios geomagnéticos sem nenhum tipo de infraestrutura adicional. O sistema usa Cadeias de Markov (HMM) na modelagem de padrões de características de caminhada de pedestres para lidar com os acumulados erros causados pelo movimento natural do corpo em movimento. Além disso, é feito o uso de técnicas de remoção de anomalias, eliminando leituras de localização que não a trajetória não bate com o padrão de caminhada ou onde a velocidade da leitura é diferente da de um pedestre (0,5 a 2 m/s). Após a remoção destes pontos de anomalia nas trilhas de localização, é necessária uma calibração para preencher as lacunas dos pontos que foram removidos. Estas lacunas são preenchidas usando pontos que completem a trajetória baseados nos traces anteriores e posteriores a lacuna e assumindo uma velocidade de caminhada normal. Todo o processo pode ser observado na

Figura 11, (a) demonstra a trilha obtida através da leitura de sensores, (b) a análise de reconhecimento de padrões de caminhada, onde o segmento roxo demonstra as partes que condizem com um padrão já estabelecido pela análise de HMM. Em (c), podemos observar a remoção de uma anomalia da leitura e, finalmente em (d) o processo de calibragem que preenche a lacuna gerada pela anomalia.

Além disso, os autores demonstram preocupação com a eficiência energética do sistema e propõem um ajuste adaptativo nas taxas de amostragem dos sensores, baseando-se nos estados de caminhada dos pedestres (curso estável e instável). Quando um pedestre não muda a direção de sua caminhada é possível economizar energia aumentando o tempo entre a leitura dos sensores.

Figura 11 – Processos no rastreamento do WTrack



Fonte: Niu, Li, *et al.* (2014)

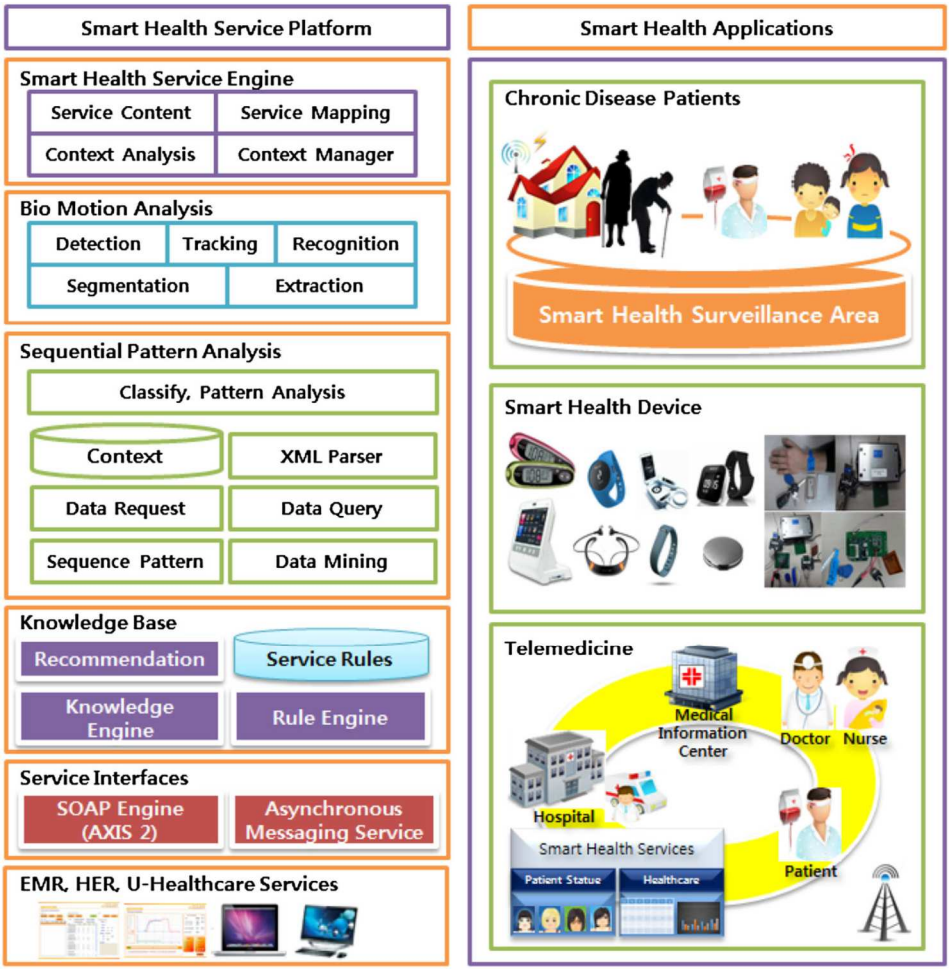
Os experimentos para a validação do WTrack foram realizados na biblioteca da Universidade de Wuhan. Os dispositivos utilizados para os testes foram 3 *smartphones* diferentes (i9300, HTC T328 W e Lenovo K900) e os dados utilizados foram extraídos dos sensores de aceleração e geomagnéticos. Para os testes, mais de 50 pessoas diferentes caminharam uma distancia total de 40km, segurando um dos dispositivos em sua mão para leitura contínua de posição, aceleração e direção.

Os resultados obtidos demonstram que o WTrack conseguiu reduzir o consumo de energia em 52%, quando comparado com o modelo de amostragem constante. Quanto a precisão do rastreamento, os resultados mostram 2m de precisão na posição, com 92,5% de probabilidade.

3.8 Sequential pattern profiling based bio-detection for smart health service

Jung e Chung (2015) propõem a utilização de técnicas de mineração de dados e reconhecimento de padrões, em uma plataforma de serviços de saúde, para detecção de correlações não facilmente reconhecidas em *life-logs*, contendo atividades de pacientes distribuídas no fluxo do tempo. Futuramente os perfis coletados poderiam ser armazenados e poderiam ser utilizados em qualquer lugar pela plataforma inteligente de serviços de saúde, ilustrada na Figura 12.

Figura 12 – Plataforma inteligente de serviços de saúde



Fonte: Jung e Chung (2015)

Primeiramente Jung e Chung (2015) descrevem como ocorre o processo de reconhecimento de imagens para sistemas inteligentes de saúde. Os dados de entrada para a plataforma são imagens coletadas por câmeras de vigilância CCD. Há uma fase de calibração onde as imagens colegadas são comparadas a imagens padrão das câmeras. As imagens são tratadas para a redução de ruído, quando presente, utilizando ajustes nos parâmetros de intensidade de luz ou distâncias dos parâmetros do ambiente. As coordenadas das imagens são então convertidas em uma matriz de área tridimensional.

Além das imagens de câmeras CCD também foi utilizada uma série de sensores e *tags* RFID para identificação de informação contextual de localização do paciente. Segundo os autores, a informação de localização pode ser obtida através do uso de sensores binários, porém por este tipo de sensor ser caro, sua replicação em um ambiente é inviável. Deste modo é difícil de reconhecer a localização precisa utilizando sensores binários e *tags* RFID. Uma maneira mais efetiva de utilizar esta tecnologia seria instalar um único sensor no paciente, e posicionar diversas *tags* RFID pelo ambiente, porém a precisão também não seria garantida, uma vez que com o aumento da densidade de *tags* RFID haveria mais interferência.

Mineração de dados é utilizada para descoberta de informações uteis que não são facilmente detectadas em dados médicos. Na área coberta pelo sistema de monitoramento de saúde inteligente, quando um objeto com uma *tag* se move, a localização corrente obtida é transmitida para um servidor de localização. Utilizando as localizações coletadas no servidor,

o caminho do objeto pode ser restaurado. O algoritmo *AprioriAll* adiciona a variabilidade de tempo as regularidades relevantes entre localizações. O algoritmo considera todas as relações antes e depois das localizações. Utilizando o *AprioriAll* um padrão de sequencia pode ser encontrado dentro de uma rota baseada na localização de uma *tag* reconhecida pelo sistema de monitoramento de saúde inteligente em uma sequencia de tempo.

A detecção de perfis de padrões sequenciais inclui padrões uteis de comportamento de um usuário na plataforma de monitoramento. Através da mineração frequente de informações de localização, o perfil é atualizado em situações normais, diminuindo o risco de uma detecção incorreta de emergência. Através da composição de perfis para situações normais, os autores afirmam ser possível conduzir detecção de emergências de uma maneira eficiente.

Para validação de desempenho o teste utilizado por (JUNG e CHUNG, 2015) foi comparar os resultados de precisão de detecção da plataforma com e sem a utilização dos perfis de padrões sequenciais propostos. Baseado na precisão dos dados gerados foi realizado um teste de hipótese para validação da hipótese de que: há diferença de desempenho de significância estatística na plataforma, utilizando perfis de padrões sequenciais. Para a validação da hipótese foi aplicado o teste t de Student, com nível de significância de 0.05. A hipótese proposta foi aceita e o modelo proposto foi mais preciso do que o original.

3.9 Comparativo

Nesta seção é feita uma comparação dos trabalhos relacionados em função de suas características, funcionalidades e relação com o modelo proposto. Foram realizados dois tipos de comparações.

Primeiramente cada um dos trabalhos relacionados foi avaliado quanto às características dos dados, classificando o domínio de atuação do trabalho, e a natureza dos dados analisados dentro das categorias de dados contextuais propostas por Dey, Abowd e Salber (2001). Os critérios de comparação para esta primeira avaliação são os seguintes:

- Entidade: Identifica quais os tipos de entidades que o trabalho considera quando trabalha com informações contextuais.
- Localização: Identifica quais tipos de dados de localização são considerados pelo trabalho. Neste caso, foram classificados como *Outdoor* (dados de localização externa, contendo coordenadas de latitude e longitude) ou *Indoor* (dados de localização em ambientes fechados, que podem conter informações como: dados de sensores de presença ou *tags* RFID).
- Situação: Identifica se o trabalho considera, ou não, dados contextuais que possam ser classificados como de situação. Situação se refere a características inerentes à entidade vinculada ao contexto. Para lugares, podem ser informações como a temperatura, umidade, luminosidade ou nível de ruído. Para pessoas ou grupos pode significar o estado emocional, seus sinais vitais.
- Atividade: Identifica se o trabalho considera, ou não, dados contextuais que possam ser classificados como de atividade. Atividade se refere à atividade que o usuário está exercendo, ou na qual está envolvido quando a informação contextual foi capturada. Um exemplo pode ser a informação de que um usuário está na cozinha, cozinhando. Ou ainda na sala, utilizando o computador.

- Domínio: Identifica qual o domínio onde o trabalho foi aplicado.

Por fim, os trabalhos foram comparados de modo a avaliar o funcionamento geral de cada um dos modelos e as funcionalidades oferecidas por eles para desenvolvedores de software. Os critérios de comparação para esta segunda avaliação são os seguintes:

- Técnicas de descoberta de padrões: Identifica quais algoritmos, modelos ou técnicas de descoberta de padrões foram utilizados em cada trabalho para realizar a busca por padrões em dados contextuais;
- Acompanha evolução dos padrões: Identifica se o trabalho realiza o acompanhamento dos padrões descobertos durante o tempo;
- Pré-Processamento: Avalia se o trabalho oferece recursos de pré-processamento dos dados a serem minerados;
- Pós-Processamento: Avalia se o trabalho oferece recursos de pós-processamento dos resultados após a mineração;
- Configurável: Avalia se o trabalho oferece parâmetros para configuração de busca por padrões utilizando outros algoritmos ou estratégias de mineração.

A Tabela 1 contém o comparativo entre os trabalhos avaliados.

Tabela 1 – Comparativo dos trabalhos relacionados

Característica	Lee, Paik, *et al.*, 2007	Huynh, Fritz *et al.*, 2008	Fatima, Fahim, *et al.*, 2013	Yin, Tian, *et al.*, 2014	Chikhaoui, Wang, *et al.*, 2014	Niu, Li, *et al.*, 2014	Jung e Chung, 2015
Entidade	Usuário de dispositivo móvel	Usuário	Usuário de smart home	Usuário de smart space	Usuário	Usuário	Usuários de serviços
Localização	Outdoor	Indoor	Indoor	Indoor	Indoor & Outdoor	Indoor	Indoor
Situação	NÃO	SIM	SIM	SIM	SIM	NÃO	SIM
Atividade	SIM	SIM	SIM	SIM	SIM	NÃO	SIM
Domínio	Serviços	Genérico	Serviços	Saúde	Genérico	Localiz.	Saúde
Técnica de descoberta de padrões	T-MAP	LDA	SPAM CRF	cSPADE FS	PST	HMM	AprioriAll
Acompanha evolução dos padrões	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO
Pré-proces.	SIM	SIM	SIM	SIM	NÃO	SIM	SIM
Pós-proces.	SIM	SIM	NÃO	NÃO	NÃO	NÃO	SIM
Configurável	NÃO	NÃO	SIM	NÃO	NÃO	NÃO	NÃO

Fonte: Elaborado pelo autor

Yin, Tian, *et al.* (2014) propõem um *framework* para reconhecimento automático de atividades humanas, com foco em sistemas de assistência a saúde. Jung e Chung (2015) utilizam descoberta de padrões de atividades para análise de estilo de vida, também com foco na saúde dos usuários.

O trabalho de Chikhaoui, Wang, *et al.,* (2014) tem como objetivo a identificação de perfis de usuários analisando relações de causalidade entre os padrões de comportamento dos usuários. Exemplos de domínios propostos para este modelo são: saúde, segurança, finanças e mídias sociais.

O objetivo do trabalho de Lee, Paik, *et al.,* (2007) é modelar corretamente padrões de comportamento dos usuários para melhorar a recomendação personalizada de serviços móveis. Já o trabalho de Fatima, Fahim *et al.*, (2013) se baseia em análise do comportamento de habitantes, que varia de reconhecimento atividade a previsão de ações, a fim de apoia-los em ambientes de casas inteligentes.

Huynh, Fritz e Schiele (2008) têm como objetivo em seu trabalho realizar o reconhecimento de rotinas complexas, com grande número de atividades que são executadas diariamente pelos usuários.

O único trabalho que apenas avalia dados contextuais de localização é o de Niu, Li (2014), que utiliza descoberta de padrões para melhorar a precisão de sistemas de rastreamento em ambientes fechados. É importante salientar que esta estratégia, que leva em consideração apenas a informação de contexto mais simples, é a mais comum em aplicações cientes a contexto, porém é deficiente quando aplicada em domínios diferentes.

O único trabalho que considera dados de localização como *Indoor* e *Outdoor* é o de Chikhaoui, Wang, *et al.*, (2014), enquanto os demais apenas suportam um dos tipos. É importante salientar que isso também pode se tornar um empecilho para aplicação dos trabalhos que apenas consideram um dos tipos de dado de localização em diferentes domínios. O modelo CHSPAM possui como estratégia para a representação de dados de localização o vetor associativo, possibilitando a utilização de informações *Indoor* e *Outdoor*.

A totalidade dos trabalhos avaliados possui como objetivo avaliar padrões relacionados a pessoas, usuários de aplicações específicas, sempre visando à melhoria dos serviços destas aplicações. Neste critério, destaca-se que nenhum trabalho trata entidades genéricas, o que pode impossibilitar sua utilização para outras aplicações onde as entidades são diferentes de pessoas (por exemplo, automóveis, sistemas e recursos computacionais). Deste modo, apesar de alguns dos trabalhos se encaixarem na classificação de domínio genérico, as opções de aplicação dos mesmos são reduzidas pela restrição no fato de representarem as entidades como pessoas. O CHSPAM usa uma abordagem genérica para representação de entidades e contextos, aumentando a possibilidade de aplicações do modelo, quando comparado aos trabalhos relacionados.

Quanto as funcionalidades oferecidas, a maioria dos trabalhos avaliados oferece recursos de pré-processamento, com exceção ao trabalho de Chikhaoui, Wang, *et al.,* (2014). O único trabalho que oferece opção de configuração de outras estratégias para a descoberta de padrões é o de Fatima, Fahim *et al.*, (2013), o restante dos trabalhos possuem estratégia definida de busca, sem a possibilidade de alteração. Além disso, muitos deles apresentam estratégias específicas para o tipo de domínio onde estão aplicados, o que também dificulta a aplicação destes trabalhos em outros tipos de domínios. O CHSPAM possibilita a configuração de diferentes estratégias de busca e oferece recursos de pré e pós-processamento.

Nenhum dos trabalhos avaliados realiza o acompanhamento dos padrões descobertos com o decorrer do tempo. Esta característica é uma contribuição do CHSPAM, que possibilita a descoberta de padrões sequenciais e acompanha sua evolução ao longo do tempo.

3.10 Considerações finais sobre o capítulo

Neste capítulo foi realizado um estudo comparativo dos trabalhos relacionados. Inicialmente foram realizadas buscas em bases de dados de periódicos a procura de pesquisas que pudessem ser comparadas ao modelo proposto nesta dissertação. Percebeu-se que os trabalhos que abordam a questão da descoberta de padrões em sequências de dados contextuais em geral concentram-se em domínios e métricas específicas.

É uma questão interessante de pesquisa a especificação de um modelo genérico e configurável, que permita combinar as funcionalidades de pós e pré-processamento, além de proporcionar diferentes técnicas para busca de padrões de modo independente, já que nenhum dos trabalhos possui tais características.

Por fim, o acompanhamento dos padrões descobertos em função do tempo é uma característica que não está presente em nenhum dos trabalhos analisados.

Tendo em vista estas questões, é proposta a especificação de um modelo para descoberta e acompanhamento de padrões em históricos de contextos, descrito no próximo capítulo.

4 MODELO CHSPAM

Este capítulo apresenta CHSPAM (*Context History Pattern Monitoring*), um modelo para descoberta e acompanhamento de padrões em históricos de contextos. O modelo CHSPAM foi proposto com base nos aspectos identificados a partir do estudo sobre tecnologias aplicadas à computação ubíqua, mineração de dados e busca de padrões em históricos de contextos.

A primeira seção apresenta a visão geral do modelo, a seção 4.2 apresenta os requisitos que o modelo atende, a seção 4.3 descreve a arquitetura do CHSPAM e cada um dos seus componentes. Por fim, as seções 4.4 e 4.5 apresentam o processo de busca e a configuração do modelo.

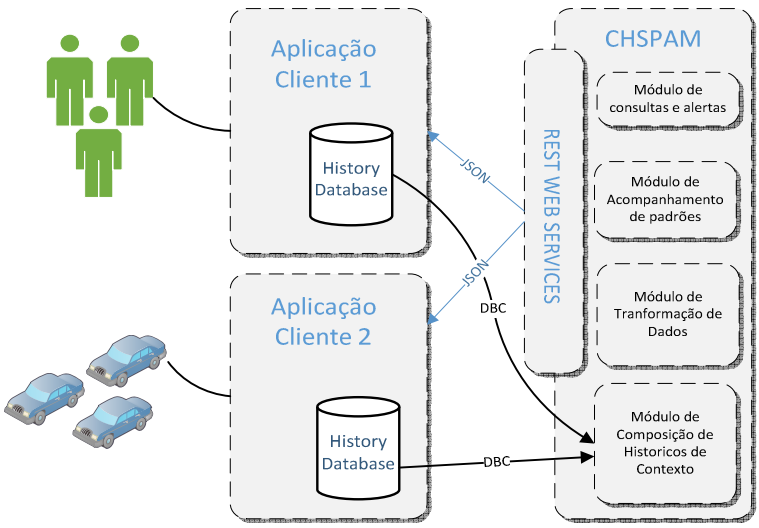
4.1 Visão Geral

O objetivo deste trabalho é a especificação de um modelo multi-domínio para descoberta e acompanhamento de padrões em históricos de contextos. O modelo provê uma interface para configuração de busca e acompanhamento de padrões.

A Figura 13 ilustra a visão geral do modelo. Na figura é possível observar dois sistemas externos e suas entidades, um dos sistemas possui seus usuários como entidades e o outro, automóveis. Dados contendo históricos de contextos das entidades estão armazenados na base de dados de cada um dos sistemas. O modelo CHSPAM realiza acesso às bases de dados destes sistemas para extração dos históricos, através de conexões diretas ao banco. Posteriormente o modelo realiza a descoberta de padrões nestes dados de históricos.

Os serviços do CHSPAM podem ser consumidos por outras aplicações para que possa ocorrer o acompanhamento de padrões das entidades que desejam ser avaliadas. A aplicação pode fazer consultas por padrões descobertos e armazenados pelo CHSPAM, que ficam disponíveis via *web services*, com dados disponibilizados no formato JSON.

Figura 13 - Visão geral do CHSPAM



Fonte: Elaborado pelo autor

A interface de configuração provida pelo modelo possibilita a configuração de periodicidade em que a busca deve ocorrer e a estratégia de busca. O resultado da execução de buscas é um histórico dos padrões descobertos, que contém informações relevantes desde o seu surgimento até o momento em que deixar de ser um padrão.

O modelo ainda provê a aplicação de filtros e transformações nos dados antes de seu processamento e, tais filtros podem ser utilizados para refinar e melhorar ainda mais a busca de acordo com a necessidade de cada aplicação.

4.2 Requisitos

Para que o modelo atenda aos objetivos expostos na visão geral, deve-se especificar e desenvolver as funcionalidades listadas a seguir:

- Busca de Padrões:
    a. Realizar a descoberta de padrões em históricos de contextos;
    b. Realizar o acompanhamento dos padrões descobertos;
    c. Prover meios de consultas para os padrões armazenados;
- Filtros e Transformações:
    a. Permitir a aplicação de filtros nos dados de entrada e saída, para aplicação de critérios de seleção;
    b. Permitir a realização de transformações nos dados de entrada e saída, para realização de procedimentos de pré e pós-processamento;
- Acesso a Dados:
    a. Permitir a realização de consultas em bases de sequencias de contextos ou arquivos;
    b. Permitir a configuração de periodicidade de consultas, para realizar o acompanhamento das bases de sequencias de contextos;

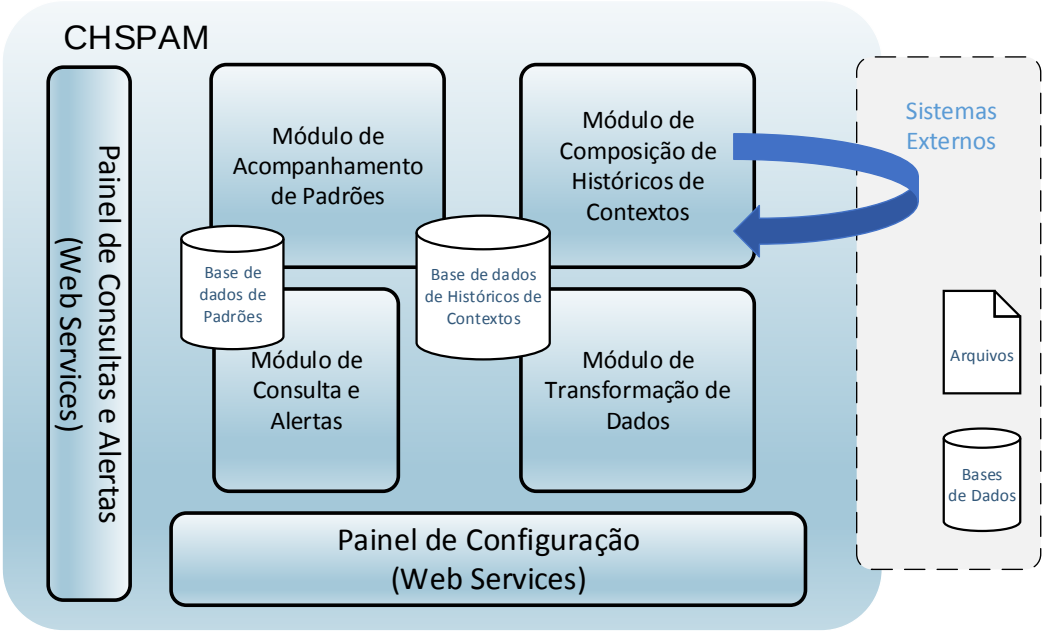
4.3 Arquitetura

A Figura 14 ilustra a arquitetura do CHSPAM. O modelo possui dois painéis, onde é possível gerenciar a configuração de parâmetros operacionais, e quatro módulos, responsáveis pela composição das sequências de contextos, execução das tarefas de transformação de dados, busca por padrões e consultas. As responsabilidades de cada um dos módulos do modelo são as seguintes:

- Módulo de composição de Históricos de Contextos:
    a. Responsável pelo acesso aos dados da aplicação cliente;
    b. Responsável pela conversão de dados da aplicação para o modelo de contexto utilizado pelo CHSPAM;
    c. Responsável pelo armazenamento dos dados de contexto em um banco de dados específico para que o módulo de extração de padrões realize as análises necessárias.

- Módulo de transformação de dados:
    a. Responsável pela execução de tarefas de pré e pós-processamento;
    b. Permite a aplicação de filtros nos dados de contextos;
    c. Permite as conversões de valores nos dados de contextos;
- Módulo de extração de padrões:
    a. Responsável pela busca de padrões nos dados de contexto armazenados e transformados;
    b. Realiza o armazenamento dos padrões reconhecidos em uma base de dados de padrões;
- Módulo de consulta e alertas:
    a. Responsável pela resposta a consultas por padrões conhecidos;
    b. Realiza a leitura e acompanhamento dos padrões reconhecidos;
    c. Possibilita o cadastro e consulta de regras de alertas;
    d. Realiza o envio de alertas quando determinadas regras são satisfeitas.

Figura 14 – Arquitetura do CHSPAM



Fonte: Elaborado pelo autor

As subseções a seguir descrevem os módulos que compõem o modelo CHSPAM, ilustrando seu funcionamento e estruturas utilizadas.

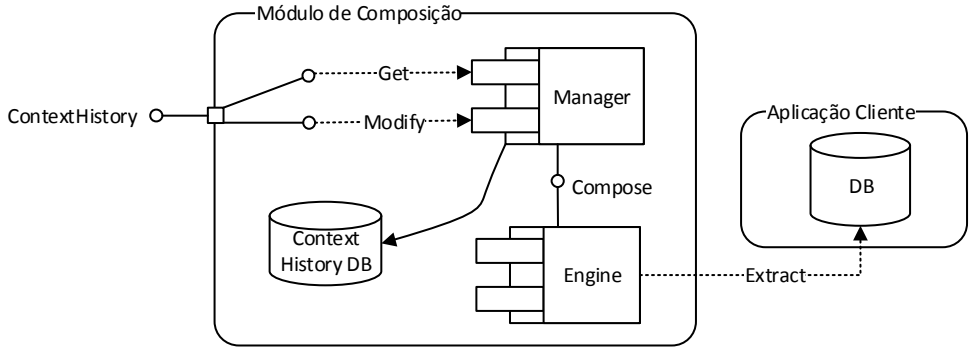
4.3.1 Módulo de Composição de Históricos de Contextos

Este módulo é responsável pelo armazenamento, gerenciamento e disponibilização dos históricos de contexto. O módulo realiza consultas em bases de sequencias de contextos já

existentes para composição de históricos de contextos que possam ser avaliados para extração de padrões sequenciais. As consultas são realizadas periodicamente pelo módulo para que possa ser realizado o acompanhamento da evolução dos históricos de contextos. Tais contextos podem estar armazenados em bancos de dados ou arquivos.

A partir da extração dos históricos de tais meios externos, o módulo armazena-os em uma base de dados específica para evitar a realização de análise e busca de padrões diretamente de bancos de dados de outras aplicações, podendo onerar as mesmas. A Figura 15 apresenta a arquitetura do módulo. Internamente o módulo é formado por um gerenciador e um motor de extração. O gerenciador (*manager*) é responsável por resolver solicitações feitas por clientes, resultando em operações de busca e atualização dos históricos. O motor de extração (*engine*) é responsável por consultar os dados das aplicações clientes para compor o histórico de contexto das entidades.

Figura 15 – Arquitetura do módulo de composição de históricos



Fonte: Elaborado pelo autor

A representação dos históricos de contextos utilizada pelo modelo é baseada na definição de Dey, Abowd e Salber (2001), onde contexto é composto, por dados que descrevem a situação, identidade e localização espaço-temporal. Para representação de contextos foram avaliados alguns modelos de representação comumente utilizados (PERERA, ZASLAVSKY, *et al.*, 2014; STRANG e LINNHOFF-POPIEN, 2004), entre eles se destacam modelos baseados em *markup*, modelos orientados a objeto, modelos gráficos, modelos lógicos e modelos baseados em ontologias. Apesar de modelos lógicos ou baseados em ontologias apresentarem melhor suporte semântico, eles são mais difíceis de serem representados e armazenados, e o seu processamento e a recuperação de informação é complexa e usa muitos recursos. Deste modo, a abordagem utilizada foi o modelo orientado a objeto. A Figura 16 exibe o diagrama de classes para o domínio de histórico de contextos. Cada contexto pertence a uma entidade, que possui um identificador único. Cada contexto possui a informação de tempo em que ocorreu, bem como informações de situação e localização.

Dados de localização e situação podem apresentar diferentes tipos e formatos, além disso, cada aplicação pode possuir estratégias de armazenamento especificas para tais dados. Algumas aplicações podem utilizar modelos de localização baseados em latitude e longitude, enquanto outras, que utilizam localização *indoor* podem armazenar etiquetas dos locais visitados pela entidade.

Para que a estrutura dos dados utilizada no modelo não dificulte ou impossibilite a sua utilização pelas aplicações é adotada uma estratégia de armazenamento baseada na estrutura de dados chamada de vetor associativo. Um vetor associativo é composto por um conjunto não ordenado contendo um par, chave e valor, no qual cada chave possui um valor associado. Esta é uma das estratégias mais utilizadas para modelagem de contextos (PERERA, ZASLAVSKY,

et al., 2014). Utilizando este tipo de estratégia, o modelo é capaz de realizar o armazenamento das informações de localização da seguinte maneira:

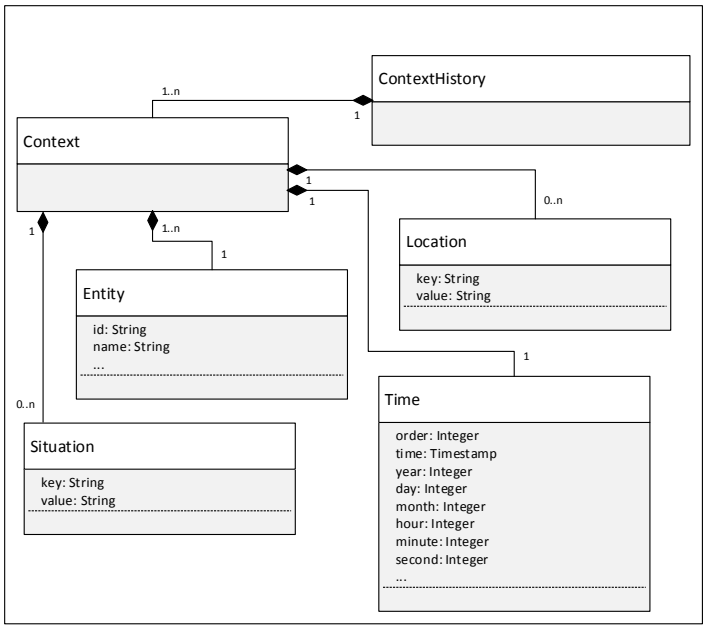
<key: CITY, $value$: $PORTO\ ALEGRE$>

<key: LATITUDE, $value$: -30.0516979>

<key: LATITUDE, $value$: -51.1819468>

Um histórico de contextos é composto por uma lista contendo todos os contextos de uma determinada entidade, ordenados cronologicamente. A Tabela 2 descreve os conceitos expostos no diagrama, situando o modelo apresentado de acordo com as definições de Dey, Abowd e Salber (2001).

Figura 16 – Diagrama de classes de Histórico de Contexto



Fonte: Elaborado pelo autor

É possível notar que apenas os atributos necessários para composição de cada uma das entidades estão ilustrados neste diagrama. Cada uma das entidades poderá possuir mais atributos, de acordo com as necessidades de aplicação. Isto será realizado através do uso de bancos de dados não relacionais, livres de *schemas*, também chamados de NOSQL ou *schemaless* (CATTELL, 2011).

Tabela 2 - Definições de conceitos do modelo

Conceito por Dey, Abowd e Salber (2001)	Descrição
Context	Contém a descrição da situação da entidade em um momento e local específico
Context History	Contém a lista de registros contextuais passados da entidade, cronologicamente ordenados
Entity	Entidade proprietária do histórico de contextos
Location	Armazena a lista de variáveis que identificam a posição geográfica da entidade no momento do registro, descrevendo a localização que esta posição representa
Status	Armazena a lista de variáveis disponíveis que descreve a situação da entidade no momento e no local deste registro
Time	Armazena informações relacionadas ao tempo do registro do contexto.

Fonte: Elaborado pelo autor

4.3.2 Módulo de Transformação de Dados

O módulo de transformação de dados é o responsável pela execução de filtros e transformações nos dados de entrada e saída suportadas pelo CHSPAM. Tais funções podem ser utilizadas para aplicação de critérios de seleção e realização de procedimentos adicionais. Além disso, este tipo de abordagem permite ao usuário personalizar a busca de padrões para seu problema.

Por exemplo, supondo uma análise a ser realizada sobre o histórico de dados de contextos de pacientes com doenças cardíacas. Certamente um dos dados de contexto que devem ser armazenados pelo sistema é o de frequência cardíaca. Geralmente dados provenientes de sensores possuem grande quantidade de leituras armazenadas de forma sequencial.

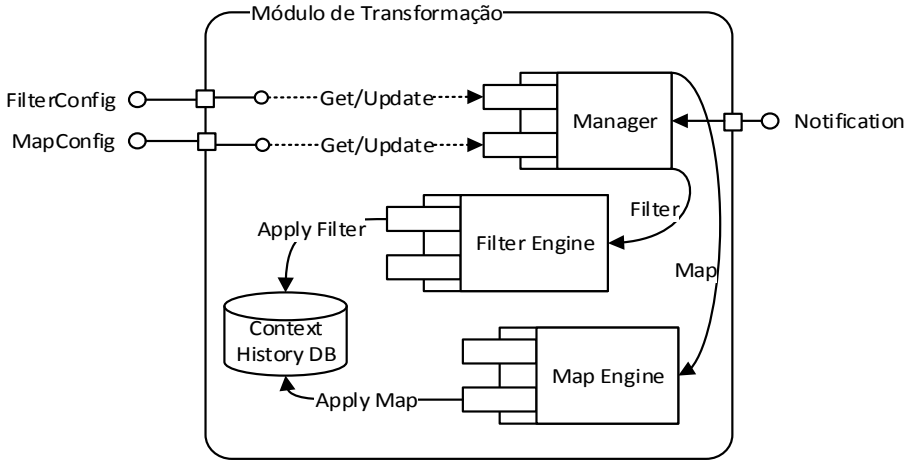
Seria mais interessante aplicar, na fase de pré-processamento, um filtro para remover grandes números de registros com valores repetidos em sequencia. Outro tipo de pré-processamento que pode ser aplicado no mesmo cenário é uma transformação de dados, que avalie a frequência cardíaca em batimentos por minuto e realize a conversão para uma constante que represente a situação do paciente naquele determinado instante, por exemplo, frequência normal, elevada ou baixa. Deste modo a informação de padrões minerados poderá ser mais rápida e os padrões extraídos, como resultado, poderão ser mais significativos.

Este mapeamento pode ser conseguido através do emprego de Ontologias nas funções de mapeamento. Uma ontologia é uma especificação explícita de uma conceitualização (GRUBER, 1995). O termo foi tomado emprestado da filosofia, porém com outro significado. Em sistemas de representação do conhecimento, uma ontologia pode ser descrita como a

especificação formal de conceitos sobre um determinado domínio, e os relacionamentos entre estes conceitos.

A arquitetura do módulo de transformação de dados é apresentada na Figura 17. Internamente o módulo é formado por um gerenciador (*manager*) e dois motores de execução, um responsável pela aplicação de filtros (*filter engine)*, e outro pelas transformações (*map engine*). O gerenciador é responsável por receber e armazenar as configurações atualizadas de filtros e transformações. Além disso, o gerenciador também recebe notificações dos demais módulos do sistema para realizar a execução dos filtros e transformações configurados.

Figura 17 – Arquitetura do módulo de transformação



Fonte: Elaborado pelo autor

O módulo de transformação de dados suporta dois tipos de opções para realização de transformações e filtros:

- Mapa: Função de transformação que será aplicada para cada um dos contextos que compõem um histórico de contextos. O resultado de sua execução será um novo histórico, composto por cada um dos contextos, modificados pela função Mapa. Um exemplo de uma função Mapa, implementado utilizando a linguagem de programação Java pode ser visto na Figura 18. No exemplo, a função está sendo utilizada para transformar dados de sensores lógicos em ações rotuladas.

Figura 18 - Exemplo da função Mapa

```
public Context apply(Context c) {
    Context mapedContext = new Context();
    mapedContext.setEntity(c.getEntity);
    mapedContext.setTime(c.getTime());
    mapedContext.setLocation(c.getLocation());

    if ( c.getSituation("SENSOR_1")) {
       mapedContext.setSituation(new HashMap<String,String>("ACTION", "SHOWER"));
    }
    if ( c.getSituation("SENSOR_2")) {
       mapedContext.setSituation(new HashMap<String,String>("ACTION", "SLEEP"));
    }
    if ( c.getSituation("SENSOR_3")) {
       mapedContext.setSituation(new HashMap<String,String>("ACTION", "LAUNDRY"));
    }
    return mapedContext;
}
```

Fonte: Elaborado pelo autor

- Filtro: Função que testa cada um dos contextos que compõem um histórico de contextos e retorna um valor booleano. O resultado de sua execução é um novo histórico, composto por apenas os contextos que satisfazem o teste da função de Filtro, retornando o valor verdadeiro. Um exemplo de uma função Filtro, também utilizando Java pode ser visto na Figura 19. No exemplo, a função está sendo utilizada para filtrar dados contextuais, para que na análise só sejam considerados contextos com informação de frequência cardíaca maiores que zero.

Figura 19 - Exemplo da função Filtro

```
public boolean test(Context c) {
    Integer freq = Integer.parseInt(c.getSituation("FREQ_CARDIACA"));
    return freq >= 0;
}
```

Fonte: Elaborado pelo autor

4.3.3 Módulo de Acompanhamento Padrões

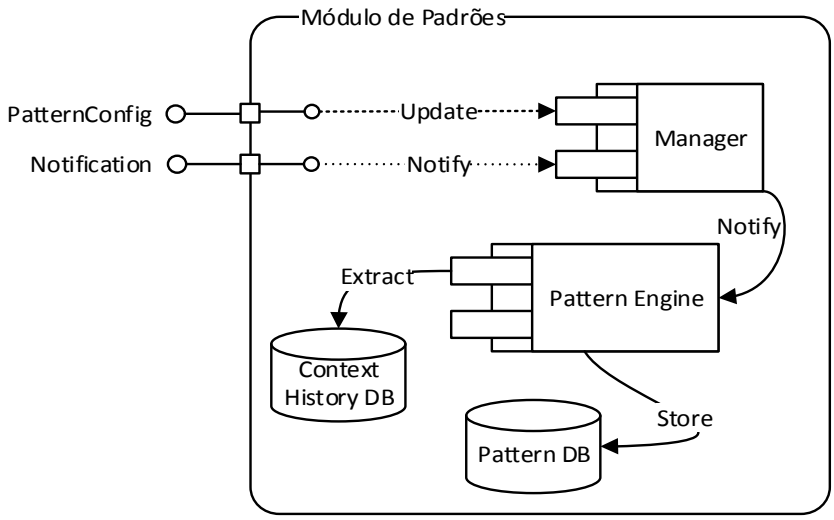
O módulo de acompanhamento padrões é responsável pela realização da descoberta de padrões, aplicando mineração de dados a base de dados de históricos de contexto, composta e pré-processada pelos módulos mencionados nas subseções 4.3.2 e 4.3.1.

Quando identificado, um padrão será armazenado em uma base de dados de padrões, para possibilitar a realização de consultas e o acompanhamento da evolução dos padrões. A definição adotada para padrão, neste caso é a de Gupta e Han (2012), onde padrão é uma subsequência de dados sequenciais, que possui uma métrica de suporte. Tal métrica se baseia no numero de vezes que a subsequência está contida nas sequencias de uma base.

A interface de configuração provida pelo modelo possibilita a configuração de periodicidade em que a busca deve ocorrer, a métrica de suporte e a estratégia de busca. A métrica de suporte busca é uma variável de suporte mínimo, dada por um percentual. Além disso, é possível utilizar múltiplas estratégias ou algoritmos de busca. Cada uma das estratégias configuradas será executada e terá seus resultados apresentados separadamente.

A Figura 20 apresenta a arquitetura do módulo. Internamente o módulo é formado por um gerenciador e um motor de descoberta de padrões. O gerenciador é responsável por receber e armazenar as configurações atualizadas para descoberta de padrões. Além disso, o gerenciador também recebe notificações dos demais módulos do sistema para dar inicio a execução da descoberta de padrões de acordo com as opções configuradas. O motor de descoberta (*engine*) é responsável pela mineração da base de dados de históricos de contextos e pelo armazenamento dos padrões identificados na base de dados de padrões.

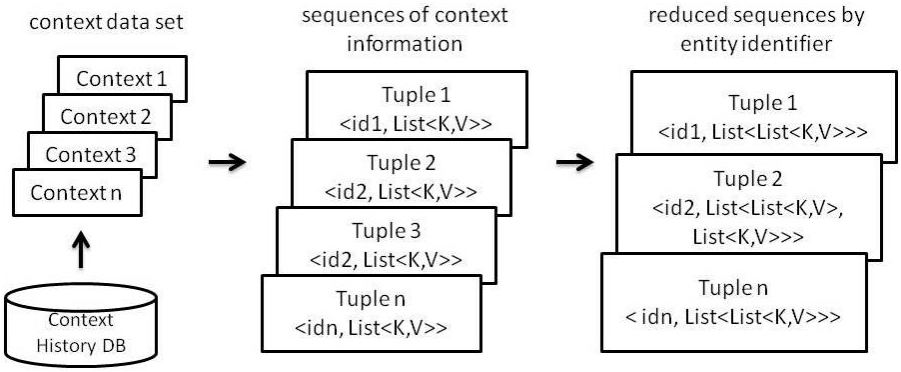
Figura 20 – Arquitetura do módulo acompanhamento de padrões



Fonte: Elaborado pelo autor

O acompanhamento dos padrões é realizado em três passos. O primeiro passo, ilustrado na Figura 21, é preparar os contextos para serem minerados. Primeiro, os históricos de contexto armazenados são convertidos em um conjunto de vetores. O banco de dados de histórico de contexto completo é usado como fonte de dados, cada contexto armazenado é mapeado para uma tupla, composta pelo identificador de entidade e um vetor de informações contextuais, armazenadas em pares de chave/valor. Uma função de redução é aplicada, agregando o vetor de informação contextual de cada contexto em uma nova sequência para cada identificador de entidade.

Figura 21 – Preparação para mineração dos contextos



Fonte: Elaborado pelo autor

O segundo passo é o processo de descoberta de padrões. É aqui que a tarefa de mineração de dados ocorre, o conjunto de dados resultante do primeiro passo é usado como entrada para a descoberta de padrões sequenciais. O algoritmo de descoberta configurado é executado. O modelo pode ser adaptado para o uso de outros algoritmos de extração de padrões sequenciais, uma vez que o resultado do processamento deste tipo de algoritmo é uma estrutura de dados que armazena as sequências mais frequentes (padrões) e o número de vezes que elas são observadas (frequência).

O último passo é aplicar o algoritmo apresentado na Figura 22. A estrutura de sequências frequentes fornecida como resposta pela extração de padrões é utilizada como parâmetro de

entrada. Para cada um dos padrões encontrados na extração faz-se uma verificação para identificar um padrão já armazenado com a mesma sequência no banco de dados de padrões. Se este é o caso, comparamos a frequência do padrão com a frequência observada para identificar eventos relacionados ao padrão – crescimento (*GROWTH*) ou redução (*REDUCTION*) na frequência do padrão. Se o padrão não existe no banco de dados então há um evento de descoberta de padrão (*DISCOVERY*). Finalmente, o histórico do padrão é atualizado.

O resultado da última extração de padrões é comparado com a atual, para identificar o evento de extinção (*EXTINCTION*) de um padrão. Por fim, o resultado da extração atual é armazenado para ser utilizado no processo seguinte.

Figura 22 - Algoritmo de monitoramento de padrões

```
for i ← 0 to freqSequences.length
do { patternToSearch ← freqSequences[i].sequence
     patternFound ← findPatternInDB(pattern)
     if patternFound
       then { if patternFound.frequency < freqSequences[i].frequency
                then event ← "GROWTH"
                else if patternFound.frequency = freqSequences[i].frequency
                then event ← "NOCHANGE"
                else event ← "REDUCTION"
       else event ← "DISCOVERY"
     UpdatePatternHistory(freqSequences[i], event)
for i ← 0 to lastFreqSequences.length
do { patternToSearch ← lastFreqSequences[i].sequence
     patternFound ← freqSequences.find(patternToSearch)
     if not patternFound
       then UpdatePatternHistory(freqSequences[i], "EXTINCTION")
lastFreqSequences ← freqSequences
```

Fonte: Elaborado pelo autor

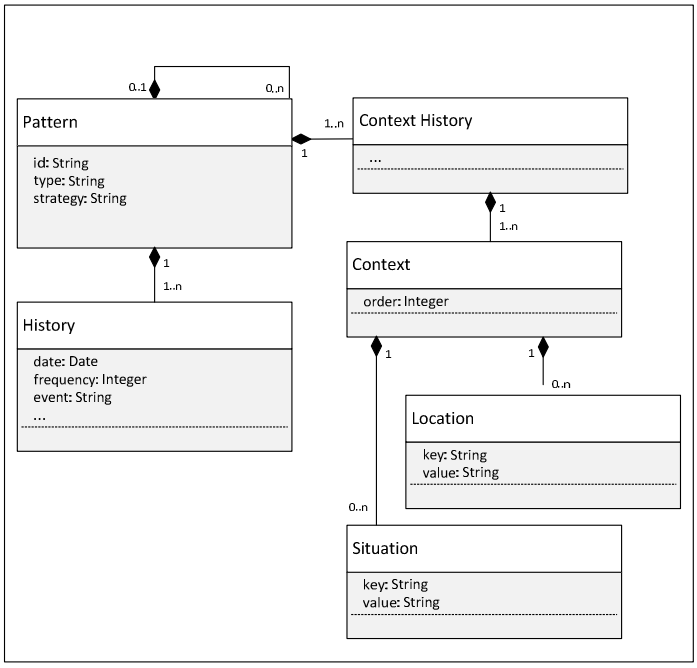
A partir desta análise, os dados de padrões são armazenados para compor o histórico dos padrões. Tal histórico contém:

- o *TimeStamp* do momento em que aquele subconjunto de contextos se tornou um padrão válido;
- a representação do próprio subconjunto;
- a taxa de crescimento de população que possui tal padrão.

A Figura 23 exibe o diagrama de classes para o domínio da entidade Padrão. Cada padrão possui um identificador único, informações sobre o tipo de padrão e estratégia utilizada para sua extração, bem como o subconjunto de contextos identificados. Além disso, para que possa ser realizado o acompanhamento do padrão, o valor total da população que possui determinado padrão e a data em que ele foi avaliado deve ser armazenado, deste modo um padrão possui um histórico contendo estas informações, juntamente com o tipo de evento identificado no momento que o histórico foi atualizado. Um padrão também pode, em alguns

casos, estar contido, ou ser um subconjunto de outro padrão, desta forma, há uma relação de composição de padrão para padrão.

Figura 23 – Diagrama de classes de Padrão



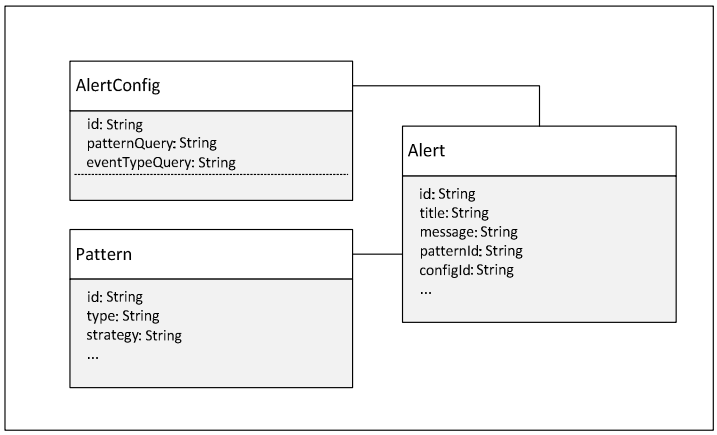
Fonte: Elaborado pelo autor

4.3.4 Módulo de Consulta e Alertas

O módulo de consulta e alertas é responsável pelo armazenamento e gerenciamento dos padrões descobertos. O módulo de padrões apenas realiza as inserções no banco de dados de padrões, enquanto o módulo de consultas é responsável por resolver solicitações de serviços provenientes de sistemas externos através de sua interface *web services*. Além de fornecer uma interface de consulta, este módulo também realiza a geração de alertas, que também podem ser consumidos pelas aplicações clientes. Os alertas são gerados a partir de configurações que são cadastradas utilizando *web services*.

Uma configuração de alerta contém um identificador único e duas propriedades que são utilizadas para seleção dos padrões que deverão gerar um alerta. Quando um evento relacionado a um padrão é identificado, tal evento é comparado com as configurações de alerta cadastradas, se o evento satisfaz uma das configurações, um alerta é gerado. A Figura 24 ilustra o diagrama de classes de alertas.

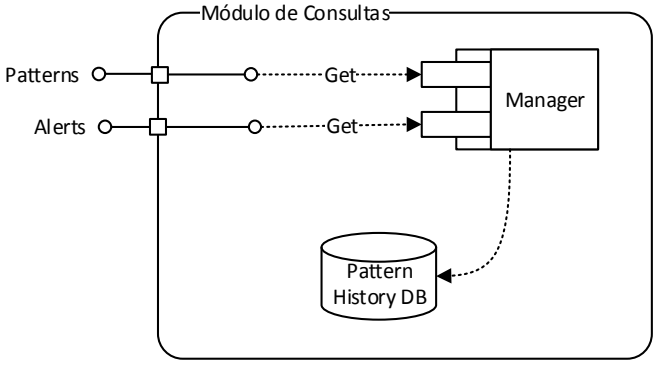
Figura 24 – Diagrama de classes de alertas



Fonte: Elaborado pelo autor

A Figura 25 apresenta a arquitetura do módulo. Internamente o módulo é formado por um gerenciador (*manager*), que realiza consultas na base de dados de padrões e disponibiliza padrões e alertas através de uma interface *web services*.

Figura 25 – Arquitetura do módulo de consultas



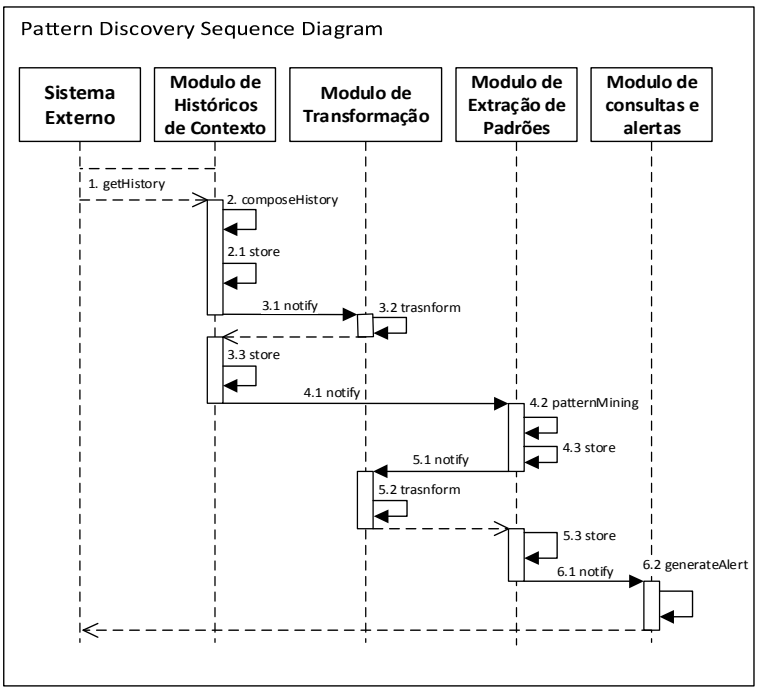
Fonte: Elaborado pelo autor

4.4 Processo de busca

A

Figura 26 apresenta o diagrama de sequencia do processo completo de uma busca por padrões do CHSPAM. O processo é iniciado pelo módulo de composição de históricos de contextos, que fará consultas à base de dados do sistema externo (1) e realizará a composição dos contextos e históricos (2), armazenando tais informações em uma base de dados específica (2.1). O módulo de transformação será notificado (3.1) e executará as transformações nos dados de históricos armazenados (3.2). Os dados tratados serão armazenados (3.3) e o módulo de extração de padrões será notificado (4.1) e iniciará a busca por padrões nos dados (4.2). Os padrões descobertos serão armazenados (4.3) e o módulo de transformação será notificado novamente para realizar as tarefas de pós-processamento (5.2). Os dados pós-processados serão armazenados (5.3) e o módulo de consultas e alertas será notificado e, fará a criação de alertas contendo os padrões identificados (6.2).

Figura 26 - Diagrama de sequência do processo de busca por padrões



Fonte: elaborado pelo autor

4.5 Configuração

A configuração do modelo pode ser realizada através de uma interface web, contendo a configuração de busca de padrões e configuração de alertas. Além disso, serviços REST (FIELDING, 2000) poderão ser utilizados para alterar as mesmas configurações. As mensagens de configuração são dadas através do formato JSON[3], que é uma formatação leve de troca de dados de fácil entendimento (CROCKFORD, 2006).

A Figura 27 apresenta um exemplo de arquivo de configuração de busca, contendo a especificação da estratégia de busca utilizando o algoritmo *PrefixSpan*, com suporte mínimo de 50% e tamanho máximo para a o histórico de contexto contendo 10 entradas. O arquivo de configuração também possibilita a configuração de filtros de pré e pós-processamento. Além disso, são configurados o intervalo de tempo para execução da análise em milissegundos e a diferença mínima das entidades armazenadas no banco para a análise.

[3] http://www.json.org/

Figura 27 - Configuração de busca

```
{
    "name": "Config 1",
    "application": "application_1",
    "patternConfiguration": {
        "strategy": "prefixSpam",
        "minSupport": 0.5,
        "maxPatternLength": 10
    },
    "triggerConfiguration": {
        "timeInterval": 60000000,
        "minRowDifference": 300
    },
    "transformationConfiguration": {
        "preFilter": "/filters/InputFilter.java",
        "postFilter": "/filters/OutputFIlter.java",
        "preMap": "/maps/InputMap.java",
        "postMap": "/maps/OutputMap.java"
    }
}
```

Fonte: elaborado pelo autor

4.6 Considerações sobre o capítulo

Neste capítulo foi apresentada a especificação do modelo CHSPAM. Inicialmente foi apresentada a visão geral do modelo. Em seguida foram listados os requisitos, procurando apontar as funcionalidades que deveriam ser atendidas. A arquitetura do modelo foi apresentada e cada um dos seus componentes foi descrito. Por fim, a sequência do processo de busca e a configuração foram descritas.

5 ASPECTOS DE IMPLEMENTAÇÃO E AVALIAÇÃO

Este capítulo apresenta aspectos de implementação e avaliação do CHSPAM. Para realizar a avaliação foi desenvolvido um protótipo contemplando os principais módulos e funcionalidades do modelo.

O desenvolvimento do protótipo é discutido na seção 5.1, a avaliação na seção 5.2, por fim a seção 5.3 relata as considerações sobre o capítulo.

5.1 Desenvolvimento do protótipo

O desenvolvimento do protótipo foi dividido em duas etapas: análise e implementação. Na etapa de análise foi gerada a documentação de âmbito técnico para dar suporte e andamento à etapa de desenvolvimento. Para projeto e análise empregou-se a linguagem UML, por ser reconhecida e utilizada internacionalmente para a criação de diagramas para modelagem de sistemas (FOWLER, 2004).

O desenvolvimento ocorreu na etapa de implementação e utilizou como base a linguagem Java e bibliotecas que proporcionam suporte para o desenvolvimento utilizando o paradigma de programação funcional, que trata a computação como uma avaliação de funções matemáticas evitando estados ou dados mutáveis (HUDAK, 1989).

As linguagens funcionais dão ênfase no processo de identificar blocos e partes repetidas de códigos (como a leitura de arquivo ou cálculos em uma estrutura recursiva) e constroem funções sem efeito colateral que encapsulam a funcionalidade dentro de uma simples definição, isso faz com que se aumente a facilidade de manutenção e a confiabilidade.

Este tipo de abordagem tornou a manipulação de dados de históricos de contextos mais fácil de ser programada, utilizando funções para realização de filtragem e mapeamento foi possível realizar as transformações necessárias nos dados durante as tarefas de pré e pós-processamento. Além disso, a abordagem funcional suporta a realização de processamento de forma distribuída e escalável (WHITE, 2012).

A versão específica da linguagem de programação Java escolhida foi a Java SE8[4]. A partir da distribuição Java Standard Edition 8, a linguagem suporta expressões lambda, que provêm uma abstração para funções anônimas. Desta forma passou a permitir a execução operações no paradigma funcional em *Streams* de elementos (SCHILDT e COWARD, 2011).

Para a realização de processamento de busca por padrões, foram utilizadas bibliotecas disponíveis na plataforma Apache Spark[5]. O Apache Spark é uma plataforma popular, de código aberto, que proporciona processamento de dados em larga escala (MENG, BRADLEY, *et al.*, 2015). A plataforma é compatível com diversas linguagens de programação, entre elas estão Java, Scala e Python. Esta ferramenta foi escolhida, pois proporciona um ambiente de processamento distribuído que permite realizar a mineração de padrões de forma esacalável. O Spark realiza balanceamento de carga automático entre os nodos disponíveis para processamento.

[4] Diponível em http://www.oracle.com/technetwork/java/javase/overview

[5] Diponível em http://spark.apache.org/

Cada um dos módulos apresentados no capítulo 4 for dividido em classes para o desenvolvimento de cada funcionalidade. As funcionalidades que foram implementadas formam a leitura e composição dos históricos de contextos a partir de arquivos, filtros e transformações, descoberta de padrões, acompanhamento de padrões, configurações e geração de alertas. Deste modo, foram implementados todos os módulos e classes, apenas com restrições.

Cada um dos módulos foi desenvolvido com as seguintes restrições:

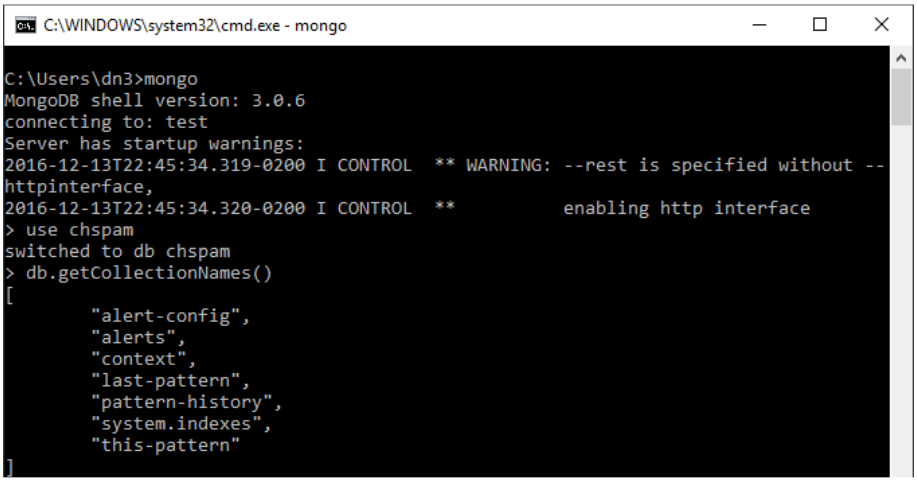
- Módulo de composição de Históricos de Contextos: Realiza a importação, composição e armazenamento de históricos de contextos através da leitura de arquivos ou consultas a banco de dados;
- Módulo de transformação de dados: Foi implementado para realização de transformações e filtros de dados utilizando funções de mapeamento e filtros;
- Módulo de extração de padrões: Realiza a descoberta e acompanhamento de padrões através das bibliotecas de mineração de dados disponibilizadas pelo *framework* Apache Spark;
- Módulo de consulta e alertas: *web services* foram desenvolvidos para consulta de padrões, consulta de acompanhamento de padrões, configuração para geração de alertas e consulta de alertas gerados.

5.1.1 Armazenamento

Quanto ao armazenamento dos dados de históricos de contextos e padrões, bancos relacionais possuem limitações, já que são pouco flexíveis quanto aos tipos e formatos de dados a serem armazenados, uma vez que sua estrutura de dados está pré-definida pela definição das tabelas (POKORNY, 2013).

Por se tratar de armazenamento de dados de contexto, que possuem características distintas para cada tipo de aplicação ou entidade, a abordagem tomada foi a utilização da base de dados MongoDB[6]. O MongoDB é um banco de dados orientado a documentos, escalável, de alto desempenho e livre de esquemas, o que simplifica o armazenamento de dados de contextos.

Figura 28 – Interface de linha de comando do MongoDB



[6] Diponível em https://www.mongodb.org/

Fonte: elaborado pelo autor

Na Figura 28 é possível observar as coleções configuradas na base de dados do CHSPAM através da interface de linha de comando do MongoDB, utilizando o comando *db.getCollectionNames()*, que retorna a lista completa de coleções existentes no banco de dados selecionado. As coleções *alerts* e *alert-config* armazenam dados de alertas gerados e a configuração para geração de alertas que são gerenciados e consumidos pelos módulos de Consultas e Alertas e pelo módulo de Acompanhamento de Padrões.

Já *context*, é a coleção que armazena os documentos de históricos de contextos importados das aplicações clientes pelo módulo de Composição de Históricos de Contextos. Os módulos de Transformações de Dados e Acompanhamento de Padrões consomem os documentos armazenados na coleção *context* para realizar os processamentos. As coleções *pattern-history*, *this-pattern* e *last-pattern* são utilizadas no armazenamento e acompanhamento de padrões: *pattern-history* armazena o histórico dos padrões, enquanto *this-pattern* armazena os padrões encontrados na execução atual do modelo e *last-pattern* salva os padrões encontrados na última execução.

Através da comparação de cada um dos padrões encontrados na execução atual (coleção *this-pattern*) com os encontrados na última execução do modelo (coleção *last-pattern*) é possível realizar o acompanhamento da evolução dos padrões.

5.1.2 Serviços oferecidos

Os serviços são disponibilizados pelo CHSPAM por meio de URIs com protocolo REST. Os *web services* desenvolvidos utilizam a estrutura JAXRS que é a implementação de REST *web services* em Java, proposta pela especificação JSR-311 (HADLEY e SANDOZ, 2009). Tais serviços foram disponibilizados através do *container* de *servlets* Apache Tomcat[7], em sua versão 8.5. O Tomcat é uma aplicação *open source*, desenvolvido pela Fundação Apache, o qual permite a execução de aplicações para web. Sua principal característica técnica é estar centrada na linguagem de programação Java, mais especificamente nas tecnologias de *Servlets* e de *Java Server Pages*.

Para atender as requisições dos *web services* foram projetados oito serviços básicos. A Tabela 3 lista os serviços disponibilizados pelo protótipo. A primeira coluna apresenta as URIs de acesso para cada serviço e os verbos HTTP utilizados em cada chamada, a segunda coluna lista os parâmetros e a terceira coluna descreve a tarefa realizada por cada serviço.

Os quatro primeiros serviços são utilizados para configuração de geração de alertas do CHSPAM. Eles provêm uma interface básica para realizar a criação, edição, consulta e deleção de configurações de alertas. O serviço que realiza a criação de configuração de alerta recebe como parâmetro um arquivo JSON, contendo os detalhes da configuração a ser salva. Já os serviços de consulta e deleção recebem por parâmetro o Id da configuração a ser consultada ou excluída.

Os demais serviços provêm acesso para realização de consultas das principais informações oferecidas pelo modelo: alertas e padrões. Dois serviços foram desenvolvidos para a consulta de cada uma destas informações. Um serviço que não requer nenhum parâmetro e lista todos os itens salvos, e outro que recebe o Id de cada item e retorna apenas os detalhes daquele item.

[7] Diponível em http://tomcat.apache.org/

Tabela 3 – Serviços do protótipo CHSPAM

URI / HTTP Verb	Parâmetro	**Descrição**
/rest/alert-configs/ DELETE	Id	Deleta configuração de alerta para o Id informado
/rest/alert-configs/ GET	Id	Retorna configuração de alerta para o Id informado
/rest/alert-configs/ GET		Retorna a lista completa de configuração de alertas
/rest/alert-configs/ POST	JSON *Payload*	Salva uma nova configuração de alerta
/rest/alerts/ GET		Retorna a lista completa de alertas gerados
/rest/alerts/ GET	Id	Retorna alerta gerado para o Id informado
/rest/patterns/ GET		Retorna a lista completa de padrões encontrados com seus respectivos históricos
/rest/patterns/ GET	Id	Retorna padrão encontrado com seu respectivo histórico para o Id informado

Fonte: elaborado pelo autor

A Figura 29 apresenta um exemplo de retorno do serviço de detalhes de um padrão, no formato JSON. Neste exemplo é possível identificar o padrão reconhecido, composto por uma lista de contextos (*pattern)*. O histórico do padrão é representado por uma lista de documentos, que contém um *timestamp* do momento em que o padrão foi computado (*discoveredIn*), a frequência apresentada pelo padrão (*frequency*) e o tipo de evento (*type*).

Figura 29 – Retorno do serviço de consulta de padrão


```
▶ _id: {counter: 10765446, date: 1482285837000, machineIdentifier: 7792122, processIde
▼ history: [{discoveredIn: 1482285837867, frequency: 4, type: "DISCOVERY"},…]
  ▼ 0: {discoveredIn: 1482285837867, frequency: 4, type: "DISCOVERY"}
      discoveredIn: 1482285837867
      frequency: 4
      type: "DISCOVERY"
  ▼ 1: {discoveredIn: 1482285859922, frequency: 8, type: "GROWTH"}
      discoveredIn: 1482285859922
      frequency: 8
      type: "GROWTH"
  ▼ 2: {discoveredIn: 1482285876414, frequency: 11, type: "GROWTH"}
      discoveredIn: 1482285876414
      frequency: 11
      type: "GROWTH"
▼ pattern: [["location:[PLACE=HOME]", "situation:[ACTION=SLEEPING]"], ["situation:[ACT
  ▶ 0: ["location:[PLACE=HOME]", "situation:[ACTION=SLEEPING]"]
  ▶ 1: ["situation:[ACTION=EATING]"]
  ▶ 2: ["location:[PLACE=HOME]"]
  ▶ 3: ["situation:[ACTION=SLEEPING]"]
```


Fonte: elaborado pelo autor

5.2 Aspectos de avaliação

Para avaliação foi utilizado o protótipo do CHSPAM. Os dados utilizados nos testes foram obtidos através de simulações e dados reais em três experimentos distintos. No primeiro, foram utilizados conjuntos de dados artificiais de contextos gerados especialmente para avaliar as principais funcionalidades e características do modelo. Tais conjuntos correspondem a testes controlados para validar cenários distintos de cada funcionalidade oferecida pelo CHSPAM.

O segundo experimento foi utilizado para comparar a predição de contextos baseada em padrões sequenciais comuns e padrões sequenciais com acompanhamento. Por fim, o último experimento apresentou a utilização do CHSPAM como componente de um sistema de recomendação de objetos de aprendizado baseado em padrões sequenciais com acompanhamento.

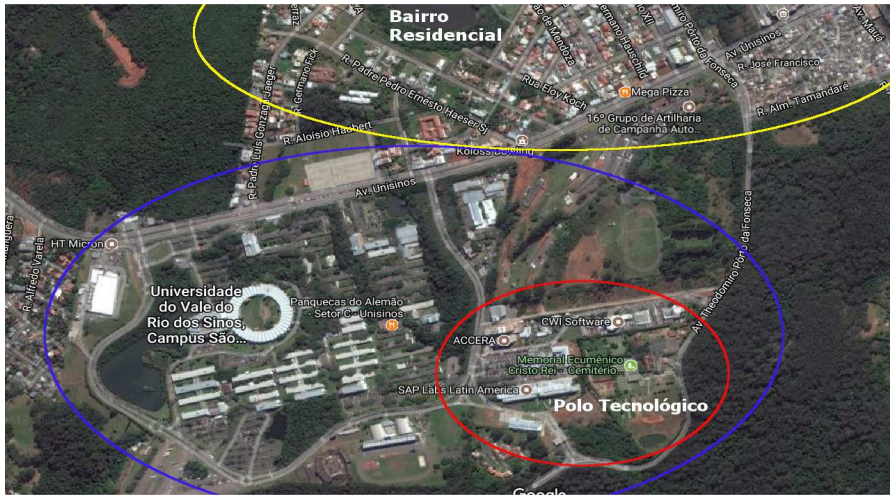
Os três experimentos são descritos nas seções a seguir.

5.2.1 Testes de Funcionalidade

O conjunto de dados utilizado para esta avaliação simulou a base de uma aplicação cliente, sendo alimentado incrementalmente, simulando a evolução de uma base de dados real. Desta maneira a capacidade do modelo realizar o acompanhamento dos padrões descobertos ao longo do tempo pôde ser avaliada.

Para geração dos históricos de contextos utilizando dados artificiais foi criado um conjunto de dados composto por informações de entidades e contextos organizadas sequencialmente. Buscou-se simular um cenário de uma aplicação que acompanha e armazena dados de contextos visitados por pessoas no seu dia a dia, realizando tarefas e passando por diferentes locais. A Figura 30 ilustra o ambiente no qual a simulação foi baseada, a área próxima a Universidade do Vale do Rio dos Sinos. Dentro do *campus* da universidade há restaurantes onde alunos e professores almoçam e também um polo com empresas de tecnologia. Próximo à universidade também há um bairro residencial.

Figura 30 – Ambiente da simulação



Fonte: elaborado pelo autor

Os dados contextuais utilizados para composição do histórico de cada usuário foram baseados em três perfis distintos de pessoas:

- Estudante: João acorda e toma seu café da manhã em casa. Ele passa toda a manhã estudando para a prova que terá durante a noite. João almoça em casa e vai para a universidade no início da tarde para fazer um trabalho junto com outros colegas. No fim da tarde João resolve fazer um lanche em um dos restaurantes dentro do campus da universidade. Depois de comer, João faz a prova e então vai para casa dormir.
- Professor da universidade: Fábio acorda atrasado e resolve ir direto para a universidade. No caminho para sala onde ele dará sua primeira aula ele para em uma lanchonete e toma seu café da manhã. Após a primeira aula, Fábio almoça em um restaurante dentro da universidade. Fábio ministra a segunda aula durante a tarde e retorna para casa. Em casa, Fábio janta com sua família antes de dormir.
- Trabalhador do polo tecnológico: Leandro acorda e toma seu café da manhã em casa. Ele vai até a empresa onde trabalha, dentro do *campus* da universidade. Leandro trabalha durante todo o dia, parando apenas para almoçar com seus colegas em um restaurante próximo a empresa. No fim da tarde Leandro vai para casa onde janta sozinho antes de dormir.

Um exemplo do histórico de contextos descrito para o perfil de estudante é apresentado na Tabela 4 e contém as informações de localização e situação do usuário, ordenadas em uma sequencia cronológica.

Tabela 4 – Histórico de contexto para um perfil de estudante

Sequência	Localização	Situação
1	HOME	SLEEPING
2	HOME	EATING
3	HOME	STUDYING
4	HOME	EATING
5	UNIVERSITY	STUDYING
6	RESTAURANT	EATING
7	UNIVERSITY	TEST
8	HOME	SLEEPING

Fonte: elaborado pelo autor

A geração do cenário avaliado seguiu a seguinte metodologia:

1) O *dataset* inicial foi gerado contendo registros de cinco pessoas e cerca de cinco informações contextuais para cada uma delas, seguindo um dos perfis mencionados anteriormente. Alguns dos dados de contexto utilizados para cada perfil foram repetidos de modo que na primeira execução do modelo padrões já possam ser descobertos.
2) Um processamento completo do modelo foi executado. O processamento é composto por importação dos históricos de contexto, aplicação de filtros, descoberta de padrões e geração de alertas;

3) O *dataset* foi incrementado, adicionando-se cinco novas pessoas. Novamente alguns dos dados contextuais utilizados foram repetidos, desta vez para avaliar a capacidade do modelo em identificar o crescimento da frequência com que os padrões são observados;
4) Um novo processamento completo do CHSPAM foi executado;
5) Novamente o *dataset* foi incrementado, adicionando-se mais cinco pessoas;
6) Um novo processamento completo foi executado;
7) O *dataset* foi incrementado pela ultima vez, adicionando 20 pessoas, com apenas uma informação contextual cada. Neste caso, a mesma informação contextual foi repetida para as 20 entidades adicionadas com o objetivo de avaliar a capacidade do modelo em identificar a extinção de um padrão.
8) Uma última rodada de processamento foi executada.

Os aspectos avaliados através dos testes realizados com dados artificiais tiveram como objetivo a validação das principais funcionalidades oferecidas pelo protótipo do CHSPAM. Os resultados obtidos para cada funcionalidade são descritos a seguir.

Nos testes realizados com o *dataset* contendo dados artificiais o protótipo se mostrou capaz de importar e armazenar os dados de históricos de contextos. A tarefa de importação foi executada na fase inicial de cada processamento realizado pelo protótipo, os dados foram armazenados em uma base de dados NoSQL, em uma coleção de documentos no formato JSON. A abordagem de centralização dos dados de históricos de contextos nesta base de dados evita a realização de busca de padrões diretamente de bancos de dados de outras aplicações, podendo onerar as mesmas. O modelo de classe de Histórico de Contexto apresentado na seção 4.3.1 se mostrou eficaz para importação e armazenamento das informações contextuais das entidades.

Quanto à funcionalidade de descoberta de padrões, o protótipo demonstrou a capacidade de realizar tal tarefa, aplicando mineração de dados na base de dados de históricos de contexto. Os padrões identificados também foram armazenados em uma base de dados NoSQL, em uma coleção de documentos no formato JSON. Para este cenário a configuração utilizada foi composta pela estratégia de busca utilizando o algoritmo PrefixSpan e suporte mínimo de 0.7, desta forma apenas sequencias de contextos que se repetem em mais de 70% dos históricos das entidades foram consideradas padrões.

A Tabela 5 apresenta um exemplo de padrão descoberto pelo protótipo neste teste. Este padrão é composto por uma sequencia de contextos conforme o seguinte cenário: Uma pessoa possui contexto de localização identificando estar em casa. Durante o dia ela apresenta dois contextos de situação identificando refeições realizadas e finalmente um ultimo contexto que une localização e situação, identificando o ato de dormir em casa. Neste caso o padrão foi descoberto durante a segunda rodada de processamento e apresentou uma frequência de valor 9. O valor de suporte é obtido através da divisão da frequência pelo número total de históricos encontrados na base, neste caso havia 10 históricos na base ($\frac{9}{10} = 0.9$).

Tabela 5 – Exemplo de padrão descoberto pelo protótipo CHSPAM

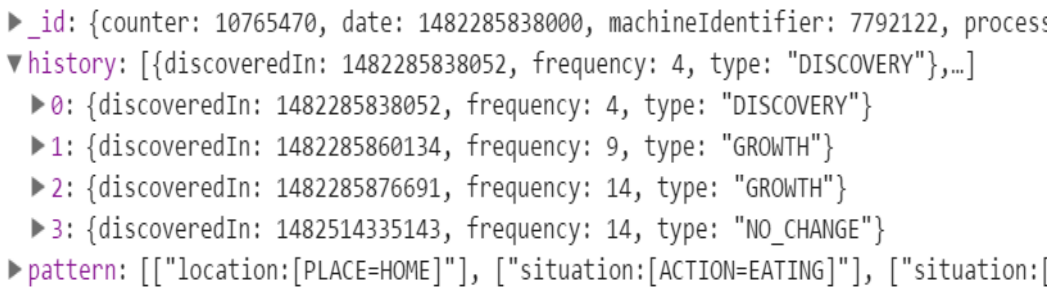
Sequência	Localização	Situação
1	HOME	-
2	-	EATING
3	-	EATING
4	HOME	SLEEPING

Frequência	9
Suporte	0.9

Fonte: elaborado pelo autor

Também foi possível verificar a capacidade do protótipo em realizar o acompanhamento dos padrões descobertos. No cenário utilizado para testes, foram executadas quatro rodadas de processamento completo do CHSPAM enquanto o *dataset* foi incrementado para simular a evolução da base de dados de uma aplicação cliente. A Figura 31 apresenta o resultado de retorno do serviço de detalhes de um padrão. Pode-se observar o resultado de cada um dos quatro processamentos executados, listados em ordem cronológica, contendo o *timestamp* (*discoveredIn*) da execução, juntamente com a frequência de observação do padrão (*frequency*) e o tipo de evento (*type*). Neste caso o padrão foi descoberto na primeira execução (*DISCOVERY*), na segunda execução houve um crescimento no número de entidades que se encaixavam naquele padrão (*GROWTH*). Na terceira execução novamente houve um crescimento e, por fim na quarta execução o número de entidades seguindo este padrão continuou o mesmo (*NO_CHANGE*).

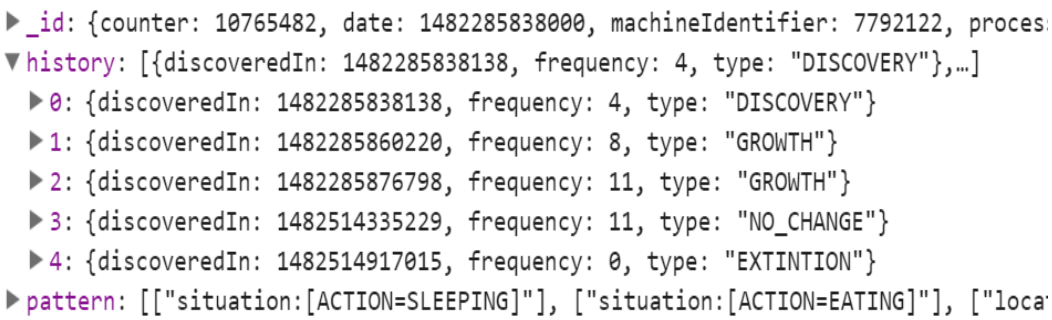
Figura 31 – Exemplo de reconhecimento de crescimento de um padrão



Fonte: elaborado pelo autor

Além de reconhecer o crescimento dos padrões descobertos anteriormente, o protótipo também se mostrou capaz de reconhecer o evento de sua extinção. A Figura 32 apresenta o resultado de retorno do serviço de detalhes de um padrão onde pode ser observado um evento de extinção (*EXTINCTION*). A obtenção deste tipo de resultado apenas foi possível devido à estratégia de armazenamento apresentada na seção 5.1.1, onde foram utilizadas duas coleções da base de dados *NoSQL* para armazenar resultados dos padrões descobertos no processamento anterior e compará-los com o processamento atual. Além de extinção e crescimento o CHSPAM também suporta o reconhecimento de eventos de redução (*REDUCTION*) no número de entidades com aquele padrão. Porém este tipo de evento só ocorre quando os históricos de contextos originais já processados são modificados na base de dados da aplicação cliente, ou quando entidades são removidas da mesma.

Figura 32 – Exemplo de reconhecimento de extinção de um padrão



Fonte: elaborado pelo autor

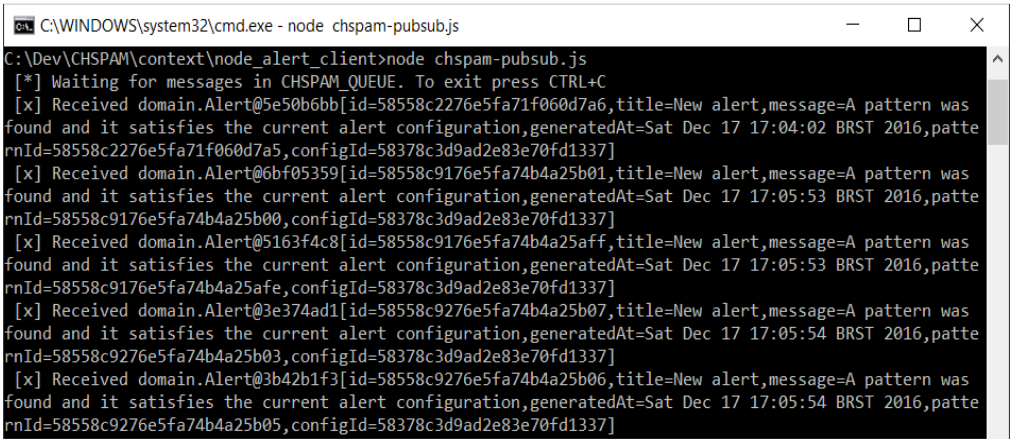
Para validar a capacidade do CHSPAM em gerar alertas foi desenvolvido uma aplicação utilizando NodeJS[8]. NodeJS é um *framework* que proporciona ferramentas para desenvolvimento de programas concorrentes, com alta performance baseado no modelo I/O assíncrono utilizando a linguagem JavaScript[9] (TILKOV e VINOSKI, 2010).

Como *broker* para envio de mensagens foi utilizado o RabbitMQ Server. O RabbitMQ é um *middleware* orientado a mensagens que implementa o padrão *Advanced Message Queuing Protocol*[10] (AMQP) e possui bibliotecas de interface disponíveis em diversas linguagens de programação (VIDELA e WILLIAMS, 2012).

A aplicação cria um canal de conexão com o servidor RabbitMQ, assina uma fila do protocolo AMQP, neste caso a fila utilizada foi "CHSPAM_QUEUE", e aguarda por mensagens enviadas. O protótipo do CHSPAM foi configurado para realizar o envio de alertas para esta fila. Uma configuração de envio de alertas foi criada para considerar alertas contendo padrões com qualquer tipo de contexto e apenas eventos de crescimento ou extinção.

A Figura 33 mostra a aplicação de teste sendo executada e os alertas sendo recebidos. Cada alerta contém título, uma mensagem e os identificadores da configuração de alerta que os gerou e do padrão relacionado, deste modo uma aplicação cliente poderá consumir os dados do padrão relacionado através do serviço de informação de padrões, que utiliza o id do padrão como parâmetro de entrada.

Figura 33 – Alertas recebidos via RabbitMQ



Fonte: elaborado pelo autor

5.2.2 Predição baseada em padrões sequenciais com acompanhamento

O primeiro experimento realizado com dados reais teve como objetivo avaliar o uso de padrões sequenciais com acompanhamento para tarefa de predição de contextos. Neste experimento foi utilizado o *dataset* de atividades diárias obtido no repositório do Centro de

[8] Diponível em http://nodejs.org

[9] Diponível em http://www.javascript.com

[10] Diponível em http://www.amqp.org

Aprendizado de Máquina e Sistemas Inteligentes da Universidade da Califórnia Irvine[11]. Este conjunto de dados é composto por informação relativa às atividades realizadas por dois usuários diariamente em suas próprias casas.

Os dados estão separados em duas instâncias, uma por usuário, somando 35 dias de dados totalmente rotulados com mais de 2700 dados contextuais. Cada instância do conjunto de dados é formada por três arquivos de texto, contendo a descrição dos sensores utilizados, eventos capturados pelos mesmos e as atividades diárias previamente rotuladas. Os eventos de sensor foram registrados usando uma rede de sensores sem fio e os dados foram rotulados manualmente. (ORD, DE TOLEDO e SANCHIS, 2013).

Tabela 6 – Lista de sensores utilizados para compor o *dataset*

Sensor	Localização	Usuário A	Usuário B
Shower	Bathroom	X	X
Basin	Bathroom	X	X
Door	Bathroom / Bedroom / Kitchen		X
Flush	Bathroom	X	X
Fridge	Kitchen	X	X
Cupboard	Kitchen	X	X
Cabinet	Kitchen	X	
Microwave	Kitchen	X	X
Toaster	Kitchen	X	
Cooktop	Kitchen	X	
Seat	Living Room	X	X
Main Door	Living Room	X	X
Bed	Bedroom	X	X

Fonte: elaborado pelo autor

A Tabela 6 mostra os sensores utilizados, sua localização e para quais usuários eles foram instalados. É possível observar que a instalação dos sensores não foi idêntica em cada uma das casas. Por exemplo, não foram instalados sensores nas portas dos ambientes da casa do Usuário A, nem no fogão, torradeira ou armário do Usuário B.

Para os testes com o CHSPAM apenas os arquivos contento os dados de capturas de sensores foram utilizados. Foi necessário implementar, no módulo de composição de históricos, uma classe chamada *FileSource* responsável por realizar o mapeamento dos arquivos, composto por *logs* de sensores, para o modelo de dados do CHSPAM, composto por históricos de contextos. Tal classe pode ser observada na Figura 34. Cada dia de um dos usuários foi tratado como sendo uma Entidade e a lista de sensores ativados durante o dia como sendo a sequência de contextos desta entidade (*Entity*). Os objetos onde os sensores foram instalados foram considerados como sendo Situações (*Situation*) de cada Contexto e o ambiente da casa onde o sensor foi instalado foi considerado a Localização (*Location*) de cada Contexto.

[11] Disponível em: http://archive.ics.uci.edu/ml/datasets

Figura 34 – FileSource implementado para o teste do CHSPAM

```
public class FileSource {
        final static Logger Logger = Logger.getLogger(FileSource.class);
        static HistoryComposer hc = new HistoryComposer();

   public static void importHistory(File fileToRead) throws Exception{

        Scanner scr = new Scanner(fileToRead);

        while (scr.hasNext()) {
                String line = scr.next();
                /* Comma separated file was used as input */
                List<String> splitedLine = Arrays.asList(StringUtils.split(line, ";"));
                /* First column of file was composed by USER + DAY*/
                Entity entity = new Entity(splitedLine.get(0));
                /* Second column of file was timestamp */
                Date date = DateUtils.parseDate(splitedLine.get(1), "dd-MM-yyyy HH:mm:ss");

                Context c = new Context();
                c.setEntity(entity);
                c.setTime(date);

                /* 4th column of file was the object where the sensor was installed */
                Map<String, String> actionCtxMap = new HashMap<String, String>();
                actionCtxMap.put("ACTION", splitedLine.get(3));
                c.setSituation(actionCtxMap);

                /* 6th column of file was sensor instalation place inside user house */
                Map<String, String> placeCtxMap = new HashMap<String, String>();
                placeCtxMap.put("PLACE", splitedLine.get(5));
                c.setLocation(placeCtxMap);

                /* Store context into context history database */
                hc.saveContext(c);

        }
        scr.close();
   }
}
```

Fonte: elaborado pelo autor

Um exemplo de histórico de contexto gerado a partir deste *dataset* está representado na Tabela 7. Pode-se observar cada uma das atividades coletadas pelos sensores, classificadas como dado contextual de localização e situação, ordenadas cronologicamente. Este exemplo ilustra parte do dia de um dos usuários, ele acorda, vai ao banheiro e toma banho. Em seguida toma café da manhã na cozinha. Passa algum tempo na sala, utiliza o banheiro novamente e sai de casa.

Tabela 7 – Histórico de contextos das atividades diárias de um usuário

Sequência	Localização	Situação
1	Bed	Bedroom
2	Cabinet	Bathroom
3	Basin	Bathroom
4	Toilet	Bathroom
5	Shower	Bathroom
6	Fridge	Kitchen
7	Cupboard	Kitchen
8	Toaster	Kitchen
9	Fridge	Kitchen
10	Cupboard	Kitchen
11	Cooktop	Kitchen
12	Microwave	Kitchen
13	Basin	Bathroom
14	Seat	Living
15	Basin	Bathroom

16	Toilet	Bathroom
17	Maindoor	Entrance

Fonte: elaborado pelo autor

O objetivo dos testes foi comparar a predição de contextos utilizando dois tipos de padrões sequenciais como base:

- Padrões sequenciais comuns: Análise simples que busca todas as subsequências que mais se repetem e que satisfazem uma métrica de suporte mínimo. Esta análise é executada apenas uma vez com todos os dados de históricos de contexto. Este tipo de padrão apresenta apenas duas informações, a subsequência e o valor de frequência obtido. Um exemplo é um padrão formado pela sequência $[A, B, C]$ com frequência f. Esse tipo de resultado pode ser obtido através da execução simples de um algoritmo de busca de padrões sequenciais.
- Padrões sequenciais com acompanhamento: Análise é executada mais do que uma vez, enquanto mais dados de históricos de contexto são adicionados. Este tipo de padrão apresenta a subsequência, e um histórico de evolução do padrão. Um exemplo é um padrão formado pela sequência $[A, B, C]$, com histórico de frequências $[f_1, f_2, f_3]$. Esse tipo de resultado pode ser obtido através de múltiplas execuções de um algoritmo de busca de padrões sequenciais em conjunto com o algoritmo apresentado na Figura 22.

Para realizar o experimento, a abordagem tomada para a avaliação foi dividir o *dataset* em dois conjuntos:

- Conjunto de treinamento: Contém dados de sensores de 23 dos 35 dias e foi utilizado para a descoberta de padrões;
- Conjunto de teste: Contém os 6 últimos dia do *dataset* de cada um dos usuários, totalizando 12 dias, e foi utilizado para testar os valores contra a predição obtida a partir dos padrões descobertos utilizando o conjunto de treinamento.

O processo para realizar a predição utilizando padrões sequenciais pode ser dividido em duas fases. A primeira é a descoberta dos padrões contidos no conjunto de dados de treinamento. A segunda fase consiste em prever um contexto que segue uma sequência fornecida pelo usuário com base nos padrões descobertos na primeira fase.

A tarefa de descoberta de padrões foi realizada utilizando o CHSPAM. A configuração do modelo foi a mesma para as duas análises (padrões sequenciais comuns e padrões sequenciais com acompanhamento):

- **Pré-processamento**: Analisando os dados contextuais dos dois usuários antes da execução do modelo, foi possível identificar inconsistências nas atividades registradas, que foram geradas pelas diferenças na instalação dos sensores nas casas de cada usuário. Estas diferenças poderiam gerar ruídos no processo de descoberta de padrões. Um exemplo deste problema são os dados contextuais de situação obtidos pelos sensores instalados nas portas dos ambientes da casa do Usuário B. Para lidar com este tipo de problema, na etapa de pré-processamento foi utilizado um filtro para remover contextos contendo leituras dos sensores das portas.
- **Estratégia de Busca**: A estratégia de busca utilizada foi o algoritmo *PrefixSpan*, apresentado na subseção 2.5.5. Este algoritmo realiza a busca das subsequências

mais frequentes, preservando a sua ordem de ocorrência. A métrica de suporte mínimo configurada foi de 0.70.

- **Pós-processamento:** Como neste teste o objetivo foi utilizar os padrões de comportamento dos usuários para predição, um filtro de pós-processamento também foi aplicado, para remover padrões com menos de quatro informações contextuais sequenciais. Deste modo apenas os padrões mais complexos, formados por cinco ou mais sequências de contextos foram armazenados.

A única diferença entre as duas análises ocorreu na análise de padrões sequenciais com acompanhamento, onde o *dataset* foi dividido em cinco partes, para simular a evolução da base de dados de uma aplicação. Os dados utilizados para a análise foram incrementados e o processo de busca apresentado na seção 4.4 foi executado cinco vezes.

Para análise de padrões sequenciais comuns, foram descobertos 129 padrões. Como resultado da análise de padrões sequenciais com acompanhamento, 931 padrões foram descobertos e acompanhados. A .

Tabela 8 apresenta três destes padrões com as informações contextuais organizadas sequencialmente.

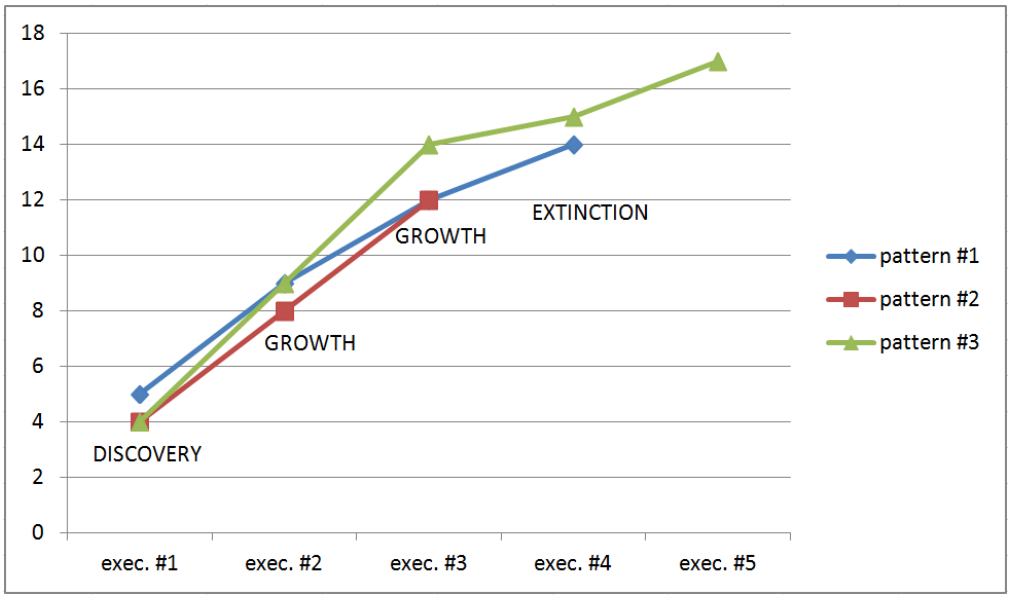
Tabela 8 – Exemplos de padrões descobertos

Padrão #1		
Sequência	Situação	Localização
1	Bed	Bedroom
2	Toilet	Bathroom
3	Microwave	Kitchen
4	Seat	Living
5	Fridge	Kitchen
6	Basin	Bathroom
Padrão #2		
Sequência	Situação	Localização
1	Bed	Bedroom
2	Fridge	Kitchen
3	Microwave	Kitchen
4	Toilet	Bathroom
5	Basin	Bathroom
6	Seat	Living
Padrão #3		
Sequência	Situação	Localização
1	Bed	Bedroom
2	Toilet	Bathroom
3	Basin	Bathroom
4	Cupboard	Kitchen
5	Fridge	Kitchen

Fonte: elaborado pelo autor

A Figura 35 apresenta a evolução dos padrões apresentados anteriormente, destacando os eventos que ocorreram com o padrão 2. Este padrão foi descoberto (*DISCOVERY*) na primeira execução do modelo (exec. #1), e apresentou frequência de valor 4. Na segunda execução (exec. #2), a frequência apresentada aumentou para 8 (*GROWTH*) e em seguida para 12 na terceira execução (exec. #3). Por fim, na quarta execução (exec. #4), a frequência observada pelo padrão 2 não foi maior do que a métrica mínima de suporte configurada (0.70), deste modo ele não foi mais identificado como um padrão, e foi considerado extinto (*EXTINCTION*).

Figura 35 - Acompanhamento da evolução dos padrões



Fonte: elaborado pelo autor

Os padrões 1 e 2 foram descobertos nas execuções iniciais do modelo, mas não foram mais considerados padrões devido aos novos históricos de contextos adicionados ao *dataset*. Dentre os padrões apresentados, o padrão 3 foi o único que também foi descoberto com a análise de padrões sequenciais comuns, uma vez que foi o único que se manteve como padrão válido até a última execução. Os padrões descobertos podem ser considerados padrões de comportamento dos usuários encontrados durante os 23 dias contidos no conjunto de dados de treinamento.

Para realizar a predição utilizando os padrões resultantes das duas análises e comparar seus resultados, foi desenvolvida uma ferramenta chamada *PPredictor*, utilizando o *framework* NodeJS. O *PPredictor* realiza predição de contextos baseada em padrões sequenciais, sendo esta predição é realizada em dois passos. O primeiro passo é verificar todos os padrões para identificar aqueles que combinam com a sequência de contextos fornecida pelo usuário. Neste caso um padrão combina com a sequência, quando os contextos antecedentes da sequência estão presentes no padrão. O segundo passo é gerar a predição, baseado em um dos padrões que combinam com a sequência, para selecionar este padrão diversos critérios podem ser utilizados. Neste caso, o padrão selecionado é o que obtém a maior pontuação, onde pontuação para um padrão é definida por $Score(p) = length(p) \times freq(p) \times type(p)$, onde $lenght(p)$ é o número de contextos contidos em p que estão presentes na sequência, $freq(p)$ é a frequência de p e $type(p)$ é o fator multiplicador baseado no tipo de evento de histórico de p. O fator multiplicador utilizado é 1,1 para históricos do tipo *DISCOVERY* e *GROWTH*, e 1 para os outros históricos (*NO_CHANGE* e *EXTINCTION*). Deste modo padrões recentemente

descobertos ou que apresentam crescimento de frequência tem um aumento de pontuação de 10% quando comparados com padrões extintos ou que não apresentaram aumento. Por fim, para selecionar o contexto utilizado para predição, o padrão com maior pontuação é comparado à sequência fornecida. O último contexto do padrão que se repete na sequencia é identificado e o contexto seguinte é selecionado.

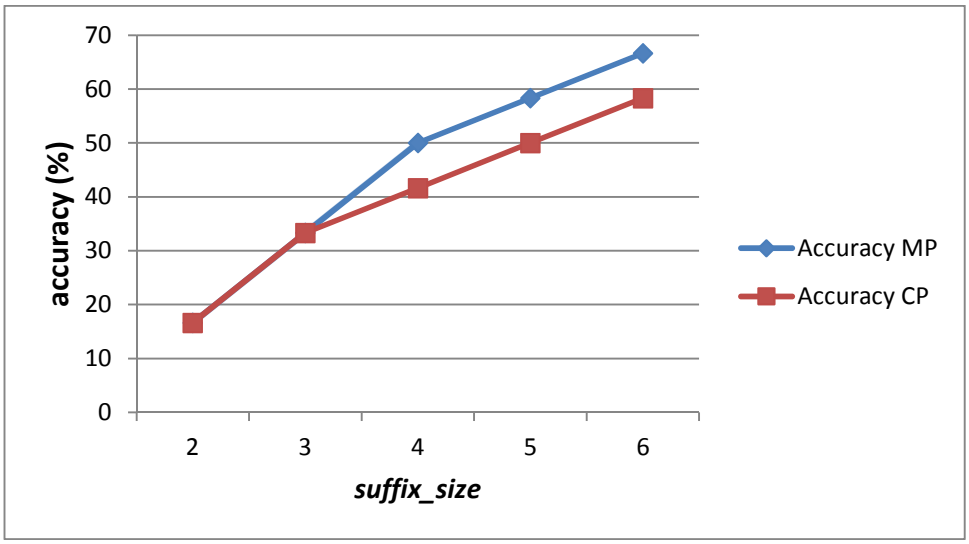
Para execução do *PPredictor* três parâmetros devem ser informados. Primeiramente o tipo de padrão que será avaliado, deste modo, a aplicação vai testar as predições utilizando padrões sequenciais comuns (CP – *Common Patterns*) ou padrões sequenciais com acompanhamento (MP – *Monitored Patterns*). O segundo parâmetro é o tamanho do prefixo (*prefix_size)*, que informa quantos contextos de cada sequência de teste devem ser utilizados como base para a predição. Já o terceiro é o tamanho do sufixo (*suffix_size*) que informa quantos contextos de cada sequencia de teste devem ser avaliados para verificar a predição. Deste modo, considerando um histórico de contextos $HC = [A, B, C, D, E, F, G]$. Dado $prefix\_size = 4$ e $suffix\_size = 3$, o problema de predição consiste em prever um item de $[E, F, G]$ com base na sequência $[A, B, C, D]$. Neste caso uma predição correta é E, ou F, ou G.

O *PPredictor* utiliza um arquivo contendo o conjunto de dados de teste como entrada do processamento e converte o seu conteúdo na representação de histórico de contexto apresentada na seção 4.3.1. Em seguida a ferramenta compara a sequência de contextos, de cada histórico com os padrões gerados pelo CHSPAM e calcula o valor da pontuação para cada padrão. A partir do padrão com maior pontuação um contexto é selecionado para predição e é testado contra os contextos da sequencia original. Como resultado o *PPredictor* apresenta a medida de precisão das predições. A precisão é definida pelo número de predições corretas, dividido pelo número total de históricos no conjunto de testes.

A avaliação foi realizada utilizando três cenários. Primeiramente o valor do parâmetro $prefix\_size$ utilizado foi 3, para avaliar a precisão utilizando um número reduzido de contextos como base para predição. No segundo cenário o parâmetro $prefix\_size$ foi aumentado para 6, e por fim, o último cenário utilizou-se o valor 12. Em cada um dos cenários, o valor de $suffix\_size$ foi variado de 2 a 6, para avaliar a diferença na precisão entre os dois tipos de padrão utilizados.

A Figura 36 apresenta o resultado obtido no primeiro cenário, que buscou avaliar a precisão utilizando apenas 3 contextos como base para a predição. É possível observar um ganho de 8% na precisão utilizando padrões monitorados em três dos cinco testes deste cenário.

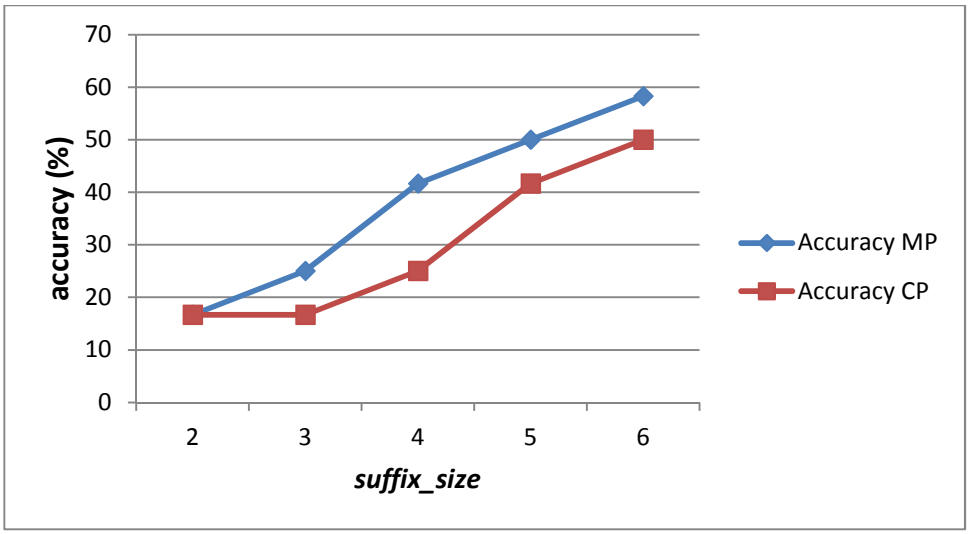
Figura 36 - Precisão das predições utilizando o parâmetro prefix_size=3



Fonte: elaborado pelo autor

A Figura 37 ilustra os resultados obtidos no segundo cenário, onde a precisão foi avaliada utilizando 6 contextos como base para predição. Este é o cenário onde ocorre a maior diferença entre a precisão utilizando os dois tipos de padrão. No teste utilizando 4 como valor de $suffix\_size$ foi possível observar um ganho de 17% de precisão na predição baseada em padrões monitorados.

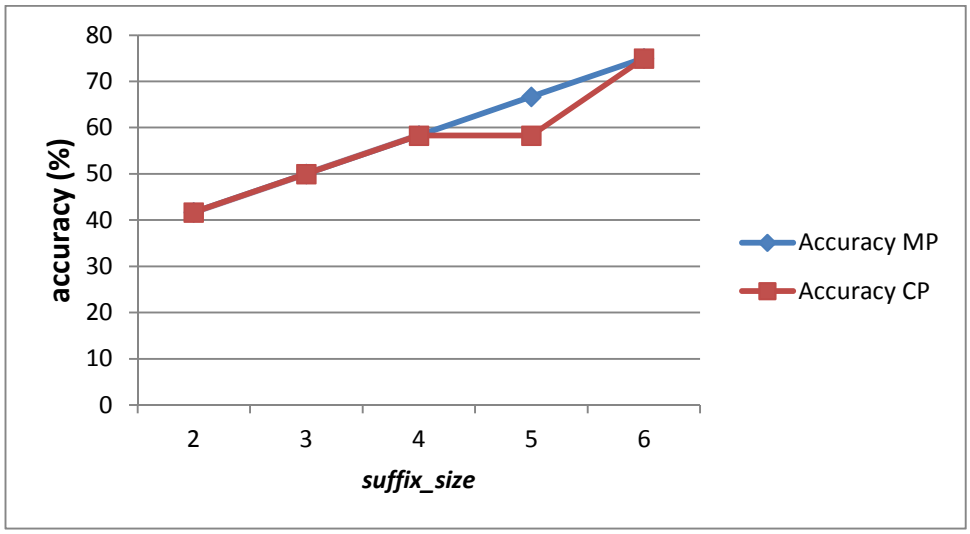
Figura 37 - Precisão das predições utilizando o parâmetro prefix_size=6



Fonte: elaborado pelo autor

O cenário onde houve menos diferença entre as duas estratégias de predição foi o avaliou a precisão da predição baseada em 12 contextos. A Figura 38 ilustra os resultados obtidos neste cenário. A única diferença observada foi um ganho de 8% em um dos testes, utilizando 5 como valor de $suffix\_size$, no restante dos testes a precisão foi a mesma para as duas estratégias.

Figura 38 - Precisão das predições utilizando o parâmetro prefix_size=12



Fonte: elaborado pelo autor

Através deste experimento foi possível concluir que o CHSPAM é capaz de realizar a análise de padrões sequenciais com acompanhamento, e que tais padrões podem ser utilizados como base para aplicações de predição. Além disso a predição baseada em padrões com acompanhamento demonstrou ganhos na precisão. Comparando a quantidade de padrões identificados em cada análise é possível também concluir que a abordagem utilizando acompanhamento resulta em um número maior de padrões, onde foi possível identificar 931 padrões contra os 129 identificados na análise mais simples. Esta diferença se mostrou impactante quando os padrões são utilizados como base para tarefa de predição.

De maneira geral, a predição utilizando padrões sequenciais com acompanhamento demonstrou maior precisão do que a utilizando padrões comuns, principalmente quando a predição se baseia em números reduzidos de contextos (cenários 1 e 2). Os resultados demonstram que a utilização de padrões sequenciais com monitoramento pode aumentar a precisão da predição em até 17% quando comparado ao uso de padrões sequenciais comuns com a mesma configuração de suporte mínimo.

5.2.3 Recomendação baseada em padrões sequenciais com acompanhamento

O segundo experimento utilizando dados reais teve como objetivo avaliar a recomendação baseada em padrões sequenciais com acompanhamento. Os dados para realização do experimento foram obtidos a partir dos *logs* de acesso ao sistema Moodle[12] do curso de Sistemas de Informação das Faculdades Integradas de Taquara/RS[13], referentes ao período de Agosto/2012 a Agosto/2013. Foram considerados os registros da tabela de *log* referentes aos acessos de alunos aos materiais disponibilizados pelos professores. Estes materiais podem ser considerados exemplos de Objetos de Aprendizagem (POLSANI, 2006). Deste modo, foram obtidas 11.039 sessões de usuários em 33.778 registros de acessos a Objetos de Aprendizagem.

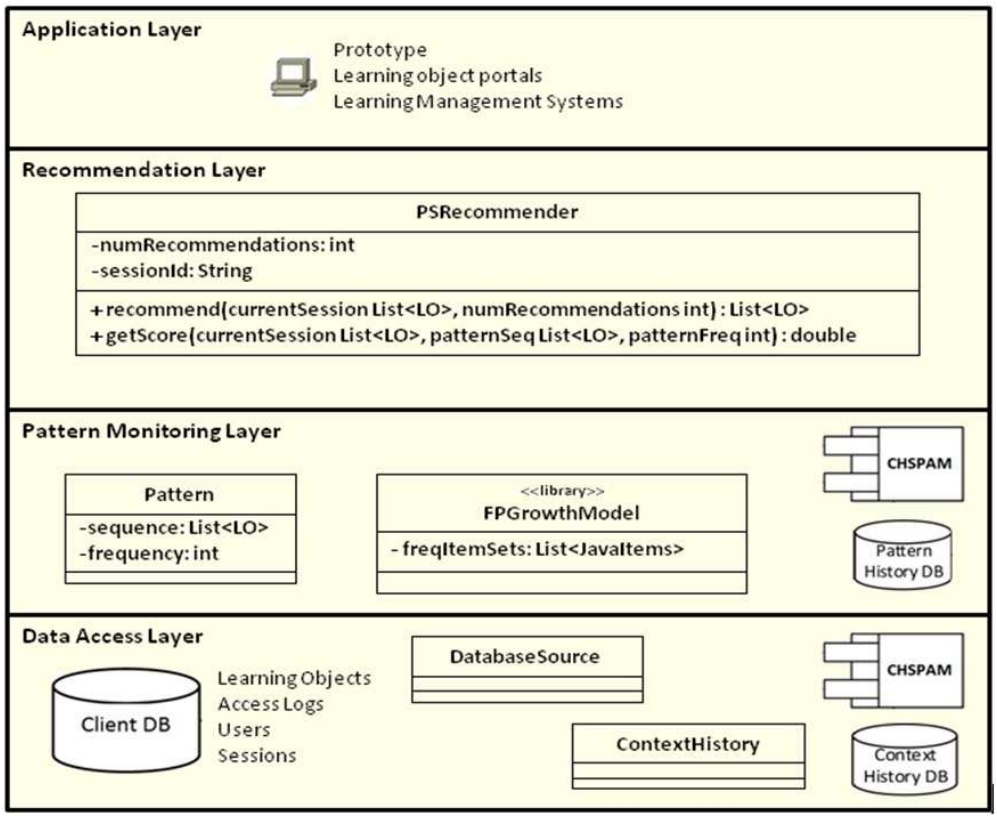
[12] Disponível em http://moodle.org

[13] Disponível em http://fit.faccat.br

O objetivo é realizar a recomendação de Objetos de Aprendizagem (OA) levando em consideração os objetos acessados durante uma sessão. Considera-se “sessão” as atividades executadas por um usuário em um ambiente de consulta de objetos de aprendizagem após o seu *login*. A aplicação proposta recebe a sequência de OAs consultados durante a sessão atual de um usuário e recomenda novos OAs baseando-se em padrões sequenciais monitorados, extraídos de históricos de acesso a OAs.

A Figura 39 mostra a arquitetura utilizada no experimento. A primeira camada corresponde à aplicação que usa o CHSPAM como, por exemplo, um módulo no portal web de um repositório. A camada de recomendação é responsável por identificar, a partir dos padrões sequenciais, OAs que ainda não tenham sido consultados na sessão atual. A terceira camada é a camada de acompanhamento de padrões, que utiliza o CHSPAM para realizar o processo de descoberta e acompanhamento dos padrões sequenciais. A quarta e última camada, chamada de camada de acesso aos dados, é responsável por acessar o histórico de sessões no repositório e mapear os dados obtidos para o modelo de dados utilizado pelo CHSPAM.

Figura 39 - Arquitetura utilizada no experimento



Fonte: elaborado pelo autor

O protótipo do CHSPAM foi utilizado como um componente da camada de acompanhamento de padrões com a tarefa de identificar e acompanhar padrões sequenciais entre as sessões armazenadas nos históricos de consulta do repositório de OAs. Para este fim, foi necessário implementar, na camada de acesso aos dados, uma classe chamada *DatabaseSource*, responsável por realizar o mapeamento dos históricos de acesso a OAs, para o modelo de dados do CHSPAM, composto por históricos de contextos. Cada sessão foi tratada como sendo uma Entidade (*Entity*) e a lista de OAs consultados em cada sessão como sendo a Sequência de Contextos desta entidade. O atributo identificador de cada OA consultado foi

convertido em Situação (*Situation*) de cada contexto. A implementação da classe *DatabaseSource* é exibida na Figura 40.

Figura 40 - Classe *DatabaseSource* implementada para o teste do CHSPAM

```
public class DatabaseSource {
   final static Logger logger = Logger.getLogger(FileSource.class);
   static HistoryComposer hc = new HistoryComposer();

   public static void importHistory() {
      try {
         Connection conn = null;
         Statement stmt = null;
         try {
            Class.forName("org.postgresql.Driver");
            conn = DriverManager.getConnection("jdbc:postgresql://localhost:5432/moodle”);
            conn.setAutoCommit(false);

            stmt = conn.createStatement();
            String query = "select sessionid, resource_id, name, datahora, ip "
            + "from mdl_log join mdl_resource on (mdl_resource.id = mdl_log.resource_id)"
            + "order by sessionid, datahora;";

            ResultSet rs = stmt.executeQuery(query);
            while (rs.next()) {
            Entity entity = new Entity(rs.getString("sessionid"));
            Map<String, String> entityProps = new HashMap<String, String>();
            entityProps.put("IP", rs.getString("ip”);
            entity.setProperties(entityProps);
            Date date = new Date(rs.getDate(datahora));
            Context c = new Context();
            c.setEntity(entity);
            c.setTime(date);

            Map<String, String> ctxMap = new HashMap<String, String>();
            ctxMap.put("LO", rs.getString("resource_id"));
            c.setSituation(ctxMap);

            hc.saveContext(c);
         }
         rs.close();
         stmt.close();
         conn.close();
      } catch (Exception e) {
         logger.error(e.getClass().getName() + ": " + e.getMessage());
         System.exit(0);
      }
   }
}
```

Fonte: elaborado pelo autor

Para a descoberta de padrões sequenciais com acompanhamento, a configuração utilizada foi a seguinte:

- **Pré e Pós-Processamento**: Não foram utilizados filtros ou mapeamento nas etapas de pré e pós-processamento.
- **Estratégia de Busca**: A estratégia de busca utilizada foi o algoritmo *FP-Growth*, apresentado na subseção 2.5.6. Este algoritmo extrai conjuntos de sequências frequentes sem geração de candidatos. Um ponto importante para esta escolha foi o fato do *FP-Growth* não permitir repetição de um mesmo item em cada sequência. Deste modo sessões com acesso repetido a um mesmo OA são reduzidas, diminuindo a complexidade da análise de padrões. A métrica de suporte mínimo configurada foi de 0.001.

Para que fosse possível realizar o acompanhamento dos padrões, a base de dados foi dividida em 12 partes, baseadas em cada mês de acesso dos *logs*. A cada execução do CHSPAM, um mês de dados de *logs* de acesso foi adicionado à base para simular a evolução da base de dados da aplicação. Como resultado da análise de padrões sequenciais com acompanhamento, foram identificados 598 padrões de históricos de acesso a OAs.

Para realizar a recomendação baseada nestes padrões, o primeiro passo é verificar todos eles para identificar quais combinam com a sequência de contextos presente na sessão do

usuário atual. Neste caso um padrão combina com a sequência, quando um ou mais contextos da sequência estão presentes no padrão.

O segundo passo é calcular um valor de pontuação para cada padrão. Neste caso pontuação para um padrão é definida por $Score(p) = length(p) \times freq(p) \times type(p)$, onde $lenght(p)$ é o número de contextos contidos em p que estão presentes na sequência, $freq(p)$ é a frequência de p e $type(p)$ é o fator multiplicador baseado no tipo de evento de histórico de p. Em seguida os padrões são ordenados em ordem crescente de pontuação e são utilizados para obtenção dos OAs a serem recomendados.

O terceiro passo é percorrer a lista de padrões ordenada por pontuação, verificando os OA não presentes na sequência de OAs da sessão corrente para então selecioná-los para serem recomendados ao usuário. Este processo é repetido até que o número máximo de recomendações seja atingido ou até o final da lista de padrões.

Para realizar a tarefa de recomendação usando os padrões resultantes da análise foi desenvolvida uma aplicação chamada *PSRecomender* (*Pattern Session Recommender*). O *PSRecomender* recebe como parâmetros o número de recomendações desejadas e um número identificador de uma sessão que é carregado do banco de dados, simulando a sessão corrente de um usuário. Em seguida o processo de recomendação é ativado, realizando uma busca nos padrões sequenciais acompanhados pelo CHSPAM e recomendando os OAs presentes nos padrões com maior pontuação.

A Figura 41 mostra o resultado da recomendação para a sessão de número [760201], vinculada ao usuário de número [1673]. Esta sessão possui um registro de acesso, referente ao OA identificado pelo código [1886]. O componente de recomendação analisou os 598 padrões e recomendou o OA de código [1887], baseado no padrão de identificador [14732800], que recebeu a maior pontuação quando comparado à sessão original. É possível observar, analisando o título dos OAs deste cenário que há relação direta entre os conteúdos dos mesmos. O *PSRecomender* foi capaz de realizar a recomendação de um OA similar ao acessado anteriormente pelo usuário baseando-se unicamente nos padrões sequenciais com acompanhamento.

Figura 41 – Resultado do processo de recomendação para sessão 760201

```
Original session:
   UID:..... 760201
   UserId:.... 1673
   IP:........ 192.168.3.143
   Date:...... Fri Aug 10 2012 01:22:31 GMT-0300 (E. South America Standard Time)
 Session Sequence
  1. [1886] Orientação a Objetos com Java
 Start Recommendation Process:
 Retrieving Patterns... 598 patterns found.
 Recommended L.O.:
  1. [1887] Começando a programar em Java | 65 | source=pattern[14732880]
```

Fonte: elaborado pelo autor

A Figura 42 mostra o resultado da recomendação para a sessão de número [951294], vinculada ao usuário de número [208]. Da mesma forma, o *PSRecomender* recomendou os OAs de código [345] e [380], baseado nos padrões [14611288] e [10436956] que obtiveram maior pontuação para a sessão. A sequência de acesso a OAs destes padrões pode ser visualizada na Figura 43. A primeira recomendação, “[345] introdução à computação gráfica” baseou-se no

fato do usuário ter acessado o OA “[343] exercícios” e esta sequência ter sido reconhecida no padrão [14611288]. Já a segunda recomendação, “[380] modelagem - material didático” se deu devido ao usuário ter acessado três, dos quatro OAs identificados no padrão [14611288], assim o OA que ainda não tinha sido acessado foi recomendado.

Novamente é possível observar a similaridade entre os conteúdos dos OAs da sessão do usuário e o que foi recomendado pela aplicação.

Figura 42 – Resultado do processo de recomendação para sessão 951294

```
Original session:
   UID:....... 951294
   UserId:.... 1110
   IP:........ 200.132.2.240
   Date:...... Tue Apr 23 2013 22:17:44 GMT-0300 (E. South America Standard Time)
 Session Sequence
  1. [343] exercícios
  2. [343] exercícios
  3. [3034] Exercícios sobre Modelagem
  4. [2374] material didático - Transformações 2D
  5. [3035] Exercícios sobre Transformações 2D
 Start Recommendation Process:
 Retrieving Patterns... 598 patterns found.
 Recommended L.O.:
  1. [345] introdução à computação gráfica | 16 | source=pattern[14611288]
  2. [380] modelagem - material didático | 16 | source=pattern[10436956]
```

Fonte: elaborado pelo autor

Figura 43 – sequencias de acesso a OAs dos padrões 14611288 e 10436956

```
Pattern ID: 10436956
Pattern Sequence:
 1. [380] modelagem - material didático | 16
 2. [2374] material didático - Transformações 2D | 16
 3. [3034] Exercícios sobre Modelagem | 16
 4. [3035] Exercícios sobre Transformações 2D | 16

Pattern ID: 14611288
Pattern Sequence:
 1. [343] exercícios | 16
 2. [345] introdução à computação gráfica | 16
```

Fonte: elaborado pelo autor

Para avaliar as recomendações realizadas pela aplicação 25 sessões de diferentes usuários foram selecionadas para serem utilizadas como cenários de teste. Estas sessões possuem históricos de acessos com tamanho variando de 1 a 71 OAs. O *PSRecommender* foi utilizado para gerar recomendações para cada sessão. Em seguida uma busca na tabela de *logs* de acesso a OAs foi realizada para verificar se o usuário de cada sessão consultou o OA recomendado posteriormente. A Tabela 9 apresenta cada uma das sessões selecionadas, o número de OAs acessados pelo usuário e a confirmação de consulta do OA recomendado baseado nesta sessão.

Das 25 recomendações realizadas, 21 foram consideradas úteis, com base no acesso realizado pelo usuário ao OA recomendado. Deste modo, em 84% dos cenários de teste avaliados as recomendações foram consideradas úteis.

Tabela 9- Utilidade das recomendações

ID da sessão	Número de OAs acessados	Consultou algum OA recomendado
948377	1	SIM
915452	1	SIM
820163	2	SIM
1000250	2	SIM
778442	2	SIM
878945	3	NÃO
800501	3	SIM
837129	3	SIM
814022	3	SIM
832103	4	SIM
776279	4	SIM
951294	5	SIM
789038	6	SIM
782746	7	SIM
857420	8	SIM
904550	10	NÃO
953224	12	SIM
1037933	12	SIM
1030568	16	SIM
857683	19	SIM
776140	29	SIM
869173	30	SIM
961141	31	NÃO
1026743	55	SIM
877346	71	NÃO

Fonte: elaborado pelo autor

A Figura 44 apresenta o resultado da recomendação para a sessão de número [776279], vinculada ao aluno [216] e registrada no dia 27 de agosto de 2012. O resultado da recomendação foi o OA "[2744] Gestão de projeto - A Equipe (ppt)". Analisando o *log* de acessos do usuário desta sessão, observado na Figura 45, é possível identificar consultas ao OA recomendado no dia 26 de novembro de 2012. Deste modo, é possível inferir que o OA recomendado está relacionado aos interesses do aluno.

Figura 44 - Resultado do processo de recomendação para a sessão 776279

```
Original session:
   UID:....... 776279
   UserId:.... 216
   IP:........ 187.5.245.126
   Date:...... Mon Aug 27 2012 14:05:19 GMT-0300 (E. South America Standard Time)
 Session Sequence
1. [2788] Modelo de Plano de Software
2. [2789] Exemplo de Plano de Software
3. [2746] Gestão de projeto - Métricas (ppt)
4. [2747] Gestão de projeto - Estimativas
Start Recommendation Process:
Retrieving Patterns... 598 patterns found.
Recommended L.O.:
 1. [2744] Gestão de projeto - A Equipe (ppt) | 77 | source=pattern[14732589]
```

Fonte: elaborado pelo autor

Figura 45 - *Log* de acessos do usuário 216 ao OA 2744

sessionid bigint	resource... bigint	datahora timestamp without time z...	userid bigint	ip characte...	name character varying
869306	2744	2012-11-26 22:36:22	216	200.132....	Gestão de projeto - A Equipe (ppt)
869306	2744	2012-11-26 22:37:16	216	200.132....	Gestão de projeto - A Equipe (ppt)

Fonte: elaborado pelo autor

Através deste experimento, foi possível concluir que os padrões sequenciais acompanhados obtidos através do CHSPAM podem ser utilizados como base para aplicações de recomendação. O protótipo desenvolvido foi capaz de identificar objetos relacionados aos interesses do aluno, mesmo sem ciência de vínculos entre os objetos consultados durante a sessão atual com os objetos recomendados. O protótipo também se mostrou capaz de tratar as estruturas de dados específicas do domínio da aplicação, neste caso Objetos de Aprendizagem, através do modelo multi-domínio proposto para a representação de Históricos de Contextos.

5.3 Considerações sobre o capítulo

Neste capítulo foram apresentados os detalhes de implementação do protótipo do CHSPAM, as tecnologias envolvidas e as estratégias utilizadas para o armazenamento dos históricos de contexto e dos padrões identificados. Os serviços disponibilizados pelo modelo também foram descritos, bem como seu funcionamento e exemplos de retorno.

Por fim, foram apresentados os resultados obtidos na utilização do protótipo do modelo em três cenários distintos. O primeiro cenário foi utilizado para validar as funcionalidades e serviços oferecidos pelo CHSPAM e foi baseado em dados sintéticos. O segundo cenário foi utilizado para comparar a predição de contextos baseada em padrões sequenciais comuns e padrões sequenciais com acompanhamento. Por fim, o terceiro cenário apresentou a utilização do CHSPAM como componente de um sistema de recomendação de objetos de aprendizado baseado em padrões sequenciais com acompanhados.

6 CONSIDERAÇÕES FINAIS

Este trabalho de dissertação abordou o problema da descoberta de padrões em históricos de contextos. Foi proposta a especificação de um modelo multi-domínio, chamado CHSPAM (*Context History Pattern Monitoring*), que possibilita o acompanhamento de padrões em históricos de contextos.

O modelo provê uma interface para configuração para busca e acompanhamento de padrões sequenciais, além de serviços para consulta de padrões e sua evolução ao longo do tempo. Tais serviços podem ser consumidos por aplicações clientes que desejam utilizar este tipo de análise.

Para a avaliação do CHSPAM foi desenvolvido um protótipo com as funcionalidades básicas necessárias dentro da proposta do modelo. Através do protótipo o modelo pôde ser avaliado em três cenários distintos, um utilizando dados sintéticos e outros dois utilizando dados de históricos de aplicações reais.

Este capítulo está dividido em três seções. A primeira descreve as conclusões gerais da pesquisa realizada. A segunda descreve as contribuições do modelo em comparação com os trabalhos relacionados. Na terceira seção são descritos os trabalhos futuros.

6.1 Conclusões

O modelo CHSPAM mostrou-se capaz de realizar a descoberta e acompanhamento de padrões em dados históricos de aplicações reais e obtendo resultados positivos conforme avaliações apresentadas no Capítulo 5.

O primeiro experimento foi utilizado para avaliar as funcionalidades e serviços oferecidos pelo CHSPAM e foi baseado em dados sintéticos. No segundo experimento, o modelo foi utilizado para comparar a predição de contextos baseada em padrões sequenciais comuns e padrões sequenciais com acompanhamento. Neste caso o modelo foi capaz de realizar o acompanhamento de padrões a partir de informações de sensores espalhados pelas casas de dois usuários, utilizando o algoritmo *PrefixSpan.* Em seguida, os padrões identificados foram utilizados como base para a predição de contextos. A comparação realizada entre os dois tipos de padrões utilizados na predição demonstrou aumento na precisão das predições quando utilizando padrões sequenciais com acompanhamento.

Em um terceiro experimento, o CHSPAM foi utilizado como componente de uma aplicação de recomendação de objetos de aprendizagem (OA) que propõe o uso de padrões sequenciais de históricos de consultas em repositórios, para identificação de OAs que possam estar relacionados aos interesses do aluno. No experimento realizado, utilizando-se uma base de dados real mantida pelo sistema de EAD de uma universidade, o protótipo desenvolvido para o experimento foi capaz de identificar objetos relacionados aos interesses do aluno, mesmo sem ciência de vínculos entre os objetos consultados durante a sessão atual com os objetos recomendados. Neste cenário, o CHSPAM foi capaz de realizar o acompanhamento de padrões sequenciais a partir dos históricos de consultas a OAs, usando o algoritmo *FP-Growth.*

Pode-se concluir ao final deste trabalho, que os objetivos foram alcançados de acordo com o que foi proposto. O modelo foi utilizado em duas aplicações distintas, para avaliar a sua

capacidade de ser aplicado em múltiplos domínios. Os benefícios do monitoramento dos padrões sequenciais também foram confirmados.

6.2 Contribuições

A partir dos trabalhos relacionados que foram pesquisados para a realização deste trabalho, foram identificados tópicos e métodos bastante explorados no tema de descoberta de padrões em dados de contexto. Este trabalho possui o foco em descoberta e acompanhamento de padrões sequenciais em históricos de contextos, e provê serviços para que aplicações consumam tais informações, o que o difere dos demais trabalhos relacionados.

Os trabalhos relacionados avaliados utilizam a descoberta de padrões para um fim específico, sendo ela parte de um sistema complexo. Esta estratégia não permite que outros sistemas acessem as informações de padrões descobertos ou até que usem estes trabalhos para realizar as tarefas de descoberta de padrões para outra aplicação.

Outro ponto importante é que os trabalhos relacionados, em sua totalidade, focam em aplicações onde as entidades são pessoas, o que pode limitar suas áreas de aplicação, enquanto o modelo CHSPAM propõe uma abordagem genérica para o modelo de contexto e entidade.

A Tabela 10 compara as características do CHSPAM com os trabalhos relacionados estudados no capítulo 3. As principais contribuições deste trabalho são:

- A criação de modelo multi-domínio que permita realizar a descoberta de padrões em bases de históricos de contextos para qualquer tipo de aplicação, através da utilização de uma abordagem genérica para representação de contextos e entidades;
- A realização do acompanhamento da evolução dos padrões de históricos de contextos descobertos no decorrer do tempo;
- O fornecimento das informações de padrões descobertos para aplicações clientes realizado através de *web services*.

Além disso, o modelo proposto também oferece ferramentas para realização de procedimentos de pré e pós-processamento, visando prover ferramentas para configuração de ajustes personalizados para cada aplicação.

Por fim, cabe ressaltar que a abordagem de descoberta, acompanhamento e serviços fornecidos pelo CHSPAM torna-o uma poderosa ferramenta que pode ser utilizada em diversas aplicações como ferramenta de pesquisa para desenvolvimento e testes.

Tabela 10 - Comparativo dos trabalhos relacionados

Característica	Lee, Paik, *et al.*, 2007	Huynh, Fritz *et al.*, 2008	Fatima, Fahim, *et al.*, 2013	Yin, Tian, *et al.*, 2014	Chikhaoui, Wang, *et al.*, 2014	Niu, Li, *et al.*, 2014	Jung e Chung, 2015	CHSPAM
Entidade	Usuário de dispositivo móvel	Usuário	Usuário de smart home	Usuário de smart space	Usuário	Usuário	Usuários de serviços	Genérico
Localização	Outdoor	Indoor	Indoor	Indoor	Indoor & Outdoor	Indoor	Indoor	Indoor & Outdoor
Situação	NÃO	SIM	SIM	SIM	SIM	NÃO	SIM	SIM
Atividade	SIM	SIM	SIM	SIM	SIM	NÃO	SIM	SIM
Domínio	Serviços	Genérico	Serviços	Saúde	Genérico	Localiz.	Saúde	Genérico
Técnica de descoberta de padrões	T-MAP	LDA	SPAM CRF	cSPADE FS	PST	HMM	AprioriAll	Configurável
Acompanha evolução dos padrões	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO	SIM
Pré-proces.	SIM	SIM	SIM	SIM	NÃO	SIM	SIM	SIM
Pós-proces.	SIM	SIM	NÃO	NÃO	NÃO	NÃO	SIM	SIM
Configurável	NÃO	NÃO	SIM	NÃO	NÃO	NÃO	NÃO	SIM

Fonte: elaborado pelo autor

6.3 Trabalhos Futuros

Como trabalhos futuros, é possível sugerir inicialmente mais testes envolvendo diferentes aplicações de análise em históricos de contextos. Uma possível avaliação seria utilizar os padrões gerados pelo CHSPAM como base para recomendação ou predição e comparar os resultados obtidos com recomendação ou predição baseada em análise de similaridade de históricos de contextos.

Outro trabalho futuro seria implementar o suporte de outras estratégias de descoberta de padrões sequenciais, como por exemplo Regras de Associação (AGRAWAL, IMIELI e SWAMI, 1993). Disponibilizar suporte à configuração de mais de um processo de extração de padrões simultaneamente também adicionaria mais valor as funcionalidades oferecidas pelo CHSPAM, tal suporte está incluso na especificação do modelo, porém não foi implementado no protótipo.

Outra extensão que o modelo pode sofrer é em relação à configuração de alertas. Atualmente a configuração de alertas é limitada ao tipo de evento relacionado ao padrão ou aos contextos que compõem o padrão. Seria interessante fornecer uma forma mais fácil de configuração, utilizando como base os históricos de contextos já identificados.

Mais uma questão para melhoria diz respeito ao tratamento de dados de localização externa utilizando latitude e longitude. É bastante difícil obter padrões utilizando este tipo de informação, uma vez que para que este tipo de padrão exista, muitas entidades devem percorrer exatamente o mesmo percurso. Para tratar este tipo de problema seria interessante realizar a implementação de um mapeamento pré-definido especificamente para conversão destes dados

em informações de localização anotadas, utilizando uma matriz de regiões ou até mesmo endereços ou CEP.

REFERÊNCIAS

ABOWD, G. D. **What next, ubicomp?:** celebrating an intellectual disappearing act. Proceedings of the 2012 ACM Conference on Ubiquitous Computing. [S.l.]: [s.n.]. 2012. p. 31--40.

ABOWD, G. D.; MYNATT, E. D. Charting past, present, and future research in ubiquitous computing. **ACM Transactions on Computer-Human Interaction (TOCHI)**, v. 7, n. 1, p. 29--58, 2000.

AGRAWAL, R.; IMIELI; SWAMI, A. **Mining association rules between sets of items in large databases**. Acm sigmod record. [S.l.]: [s.n.]. 1993. p. 207--216.

BARBERO, C.; DAL ZOVO, P.; GOBBI, B. **A flexible context aware reasoning approach for iot applications**. Mobile Data Management (MDM), 2011 12th IEEE International Conference on. [S.l.]: [s.n.]. 2011. p. 266--275.

BARBOSA, J. L. V. et al. TrailTrade: A model for trail-aware commerce support. **Computers in Industry**, v. 80, p. 43--53, 2016.

BOHM, C.; KREBS, F. The k-nearest neighbour join: Turbo charging the KDD process. **Knowledge and Information Systems**, v. 6, n. 6, p. 728--749, 2004.

BRDICZKA, O.; CROWLEY, J. L.; REIGNIER, P. Learning situation models in a smart home. **IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)**, v. 39, n. 1, p. 56--63, 2009.

BROWN, P. J.; BOVEY, J. D.; CHEN, X. Context-aware applications: from the laboratory to the marketplace. **Personal Communications, IEEE**, v. 4, n. 5, p. 58--64, 1997.

BYUN, H. E.; CHEVERST, K. **Exploiting user models and context-awareness to support personal daily activities**. Workshop in UM2001 on User Modeling for Context-Aware Applications. [S.l.]: [s.n.]. 2001.

CACERES, R.; FRIDAY, A. Ubicomp systems at 20: Progress, opportunities, and challenges. **IEEE Pervasive Computing**, n. 1, p. 14--21, 2011.

CATTELL, R. Scalable SQL and NoSQL data stores. **ACM SIGMOD Record**, v. 39, n. 4, p. 12--27, 2011.

CHAVARRIAGA, R. et al. The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition. **Pattern Recognition Letters**, v. 34, n. 15, p. 2033--2042, 2013.

CHENG, H.; YAN, X.; HAN, J. **IncSpan:** incremental mining of sequential patterns in large database. Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. [S.l.]: [s.n.]. 2004. p. 527--532.

CHIKHAOUI, B. et al. Pattern-based causal relationships discovery from event sequences for modeling behavioral user profile in ubiquitous environments. **Information Sciences**, v. 285, p. 204--222, 2014.

CLARKE, S.; DRIVER, C. Context-aware trails [mobile computing]. **Computer**, v. 37, n. 8, p. 97--99, 2004.

COOK, D. et al. **Collecting and disseminating smart home sensor data in the CASAS project**. Proceedings of the CHI Workshop on Developing Shared Home Behavior Datasets to Advance HCI and Ubiquitous Computing Research. [S.l.]: [s.n.]. p. 1--7.

COOK, D. J. et al. CASAS: A smart home in a box. **Computer**, v. 46, n. 7, 2013.

CROCKFORD, D. The application/json media type for javascript object notation (json), 2006.

DA ROSA, J. H.; BARBOSA, J. L.; RIBEIRO, G. D. ORACON: An adaptive model for context prediction. **Expert Systems with Applications**, v. 45, p. 56--70, 2016.

DEY, A. K. Understanding and using context. **Personal and ubiquitous computing**, v. 5, n. 1, p. 4--7, 2001.

DEY, A. K.; ABOWD, G. D.; SALBER, D. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. **Human-computer interaction**, v. 16, n. 2, p. 97--166, 2001.

DEY, A. K.; ABOWD, G. D.; WOOD, A. CyberDesk: A framework for providing self-integrating context-aware services. **Knowledge-Based Systems**, v. 11, n. 1, p. 3--13, 1998.

DOUKAS, C. et al. **Patient fall detection using support vector machines**. IFIP International Conference on Artificial Intelligence Applications and Innovations. [S.l.]: [s.n.]. 2007. p. 147--156.

FATIMA, I. et al. A unified framework for activity recognition-based behavior analysis and action prediction in smart homes. **Sensors**, v. 13, n. 2, p. 2682--2699, 2013.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. **AI magazine**, v. 17, n. 3, p. 37, 1996.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. The KDD process for extracting useful knowledge from volumes of data. **Communications of the ACM**, v. 39, n. 11, p. 27--34, 1996.

FIELDING, R. T. **Architectural styles and the design of network-based software architectures**. [S.l.]. 2000.

FOWLER, M. **UML distilled:** a brief guide to the standard object modeling language. [S.l.]: [s.n.], 2004.

GOMES CARDOSO, I. et al. Vulcanus 2.0: A Recommender System for Accessibility. **CLEI Electronic Journal**, v. 19, n. 1, p. 6--6, 2016.

GRUBER, T. R. Toward principles for the design of ontologies used for knowledge sharing? **International journal of human-computer studies**, v. 43, n. 5-6, p. 907--928, 1995.

GUPTA, M.; HAN, J. Applications of pattern discovery using sequential data mining. **Pattern Discovery Using Sequence Data Mining: Applications and Studies**, p. 1--23, 2012.

GUPTA, M.; HAN, J. Approaches for pattern discovery using sequential data mining. **Pattern Discovery Using Sequence Data Mining: Applications and Studies**, p. 137--154, 2012.

HADLEY, M.; SANDOZ, P. JAX-RS Java API for RESTful Web Services. **Java Specification Request (JSR)**, v. 311, 2009.

HALLER, P.; ODERSKY, M. Scala actors: Unifying thread-based and event-based programming. **Theoretical Computer Science**, v. 410, n. 2, p. 202--220, 2009.

HAN, J. et al. **FreeSpan:** frequent pattern-projected sequential pattern mining. Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. [S.l.]: [s.n.]. 2000. p. 355--359.

HAN, J. et al. **Prefixspan:** Mining sequential patterns efficiently by prefix-projected pattern growth. proceedings of the 17th international conference on data engineering. [S.l.]: [s.n.]. 2001. p. 215--224.

HAN, J.; KAMBER, M.; PEI, J. **Data mining:** concepts and techniques. [S.l.]: [s.n.], 2011.

HAN, J.; PEI, J.; YIN, Y. **Mining frequent patterns without candidate generation**. ACM Sigmod Record. [S.l.]: [s.n.]. 2000. p. 1--12.

HEARST, M. A. et al. Support vector machines. **IEEE Intelligent Systems and their Applications**, v. 13, n. 4, p. 18--28, 1998.

HIPP, J.; GUNTZER, U.; NAKHAEIZADEH, G. Algorithms for association rule mining—a general survey and comparison. **ACM sigkdd explorations newsletter**, v. 2, n. 1, p. 58--64, 2000.

HONG, J. et al. Context-aware system for proactive personalized service based on context history. **Expert Systems with Applications**, v. 36, n. 4, p. 7448--7457, 2009.

HUANG, S.-H. et al. **A decision tree approach to conducting dynamic assessment in a context-aware ubiquitous learning environment**. Wireless, Mobile, and Ubiquitous Technology in Education, 2008. WMUTE 2008. Fifth IEEE International Conference on. [S.l.]: [s.n.]. 2008. p. 89--94.

HUDAK, P. Conception, evolution, and application of functional programming languages. **ACM Computing Surveys (CSUR)**, v. 21, n. 3, p. 359--411, 1989.

HUYNH, T.; FRITZ, M.; SCHIELE, B. **Discovery of activity patterns using topic models**. Proceedings of the 10th international conference on Ubiquitous computing. [S.l.]: [s.n.]. 2008. p. 10--19.

ISHII, H.; ULLMER, B. **Tangible bits:** towards seamless interfaces between people, bits and atoms. Proceedings of the ACM SIGCHI Conference on Human factors in computing systems. [S.l.]: [s.n.]. 1997. p. 234--241.

JUNG, H.; CHUNG, K. Sequential pattern profiling based bio-detection for smart health service. **Cluster Computing**, v. 18, n. 1, p. 209--219, 2015.

KIM, E.; HELAL, S.; COOK, D. Human activity recognition and pattern discovery. **Pervasive Computing, IEEE**, v. 9, n. 1, p. 48--53, 2010.

KNAPPMEYER, M. et al. Survey of context provisioning middleware. **Communications Surveys \& Tutorials, IEEE**, v. 15, n. 3, p. 1492--1519, 2013.

KO, K.-E.; SIM, K.-B. **Development of context aware system based on Bayesian network driven context reasoning method and ontology context modeling**. Control, Automation and Systems, 2008. ICCAS 2008. International Conference on. [S.l.]: [s.n.]. 2008. p. 2309--2313.

KOHONEN, T. The self-organizing map. **Neurocomputing**, v. 21, n. 1, p. 1--6, 1998.

KONSTANTINOU, N. et al. **Priamos:** a middleware architecture for real-time semantic annotation of context features. Intelligent Environments, 2007. IE 07. 3rd IET International Conference on. [S.l.]: [s.n.]. 2007. p. 96--103.

LEE, S.-C. et al. Efficient mining of user behaviors by temporal mobile access patterns. **Int'l J. Computer Science Security**, v. 7, n. 2, p. 285--291, 2007.

LI, T.-R. et al. Sequential pattern mining. [S.l.]: [s.n.], 2005. p. 103--122.

LIN, T.-N.; LIN, P.-C. **Performance comparison of indoor positioning techniques based on location fingerprinting in wireless networks**. Wireless Networks, Communications and Mobile Computing, 2005 International Conference on. [S.l.]: [s.n.]. 2005. p. 1569--1574.

MARTIN, M.; NURMI, P. **A generic large scale simulator for ubiquitous computing**. 2006 Third Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services. [S.l.]: [s.n.]. 2006. p. 1--3.

MARTÍNEZ-TORRES, M. D. R. et al. The moderating role of prior experience in technological acceptance models for ubiquitous computing services in urban environments. **Technological Forecasting and Social Change**, v. 91, p. 146--160, 2015.

MASSEGLIA, F.; TEISSEIRE, M.; PONCELET, P. Sequential Pattern Mining. [S.l.]: [s.n.], 2005. p. 1028--1032.

MAYRHOFER, R. Context prediction based on context histories: Expected benefits, issues and current state-of-the-art. **COGNITIVE SCIENCE RESEARCH PAPER-UNIVERSITY OF SUSSEX CSRP**, v. 577, p. 31, 2005.

MENG, X. et al. Mllib: Machine learning in apache spark. **arXiv preprint arXiv:1505.06807**, 2015.

MILLER, D. A logic programming language with lambda-abstraction, function variables, and simple unification. **Journal of logic and computation**, v. 1, n. 4, p. 497--536, 1991.

MOONEY, C. H.; RODDICK, J. F. Sequential pattern mining--approaches and algorithms. **ACM Computing Surveys (CSUR)**, v. 45, n. 2, p. 19, 2013.

MOTEGAONKAR, V. S.; VAIDYA, M. V. A Survey on Sequential Pattern Mining Algorithms. **International Journal of Computer Science and Information Technologies (IJCSIT)**, v. 5, n. 2, p. 2486--2492, 2014.

NIU, X. et al. WTrack: HMM-based walk pattern recognition and indoor pedestrian tracking using phone inertial sensors. **Personal and Ubiquitous Computing**, v. 18, n. 8, p. 1901--1915, 2014.

NURMI, P.; MARTIN, M.; FLANAGAN, J. A. **Enabling proactiveness through context prediction**. Proceedings of the Workshop on Context Awareness for Proactive Systems, Helsinki. [S.l.]: [s.n.]. 2005.

ODERSKY, M.; SPOON, L.; VENNERS, B. **Programming in scala**. [S.l.]: [s.n.], 2008.

ORD; DE TOLEDO, P.; SANCHIS, A. Activity recognition using hybrid generative/discriminative models on home environments using binary sensors. **Sensors**, v. 13, n. 5, p. 5460--5477, 2013.

PAIK, J.; DONG, R.; KIM, M. XML-Encoded Context Data Mining for Ubiquitous Middleware Platforms.

PARK, H.-S.; OH, K.; CHO, S.-B. Bayesian network-based high-level context recognition for mobile context sharing in cyber-physical system. **International Journal of Distributed Sensor Networks**, 2011.

PEI, J. et al. **Prefixspan:** Mining sequential patterns efficiently by prefix-projected pattern growth. icccn. [S.l.]: [s.n.]. 2001. p. 0215.

PERERA, C. et al. Context aware computing for the internet of things: A survey. **IEEE Communications Surveys \& Tutorials**, v. 16, n. 1, p. 414--454, 2014.

POKORNY, J. NoSQL databases: a step to database scalability in web environment. **International Journal of Web Information Systems**, v. 9, n. 1, p. 69--82, 2013.

POLSANI, P. R. Use and abuse of reusable learning objects. **Journal of Digital information**, v. 3, n. 4, 2006.

RABINER, L.; JUANG, B. An introduction to hidden Markov models. **ieee assp magazine**, v. 3, n. 1, p. 4--16, 1986.

RACHURI, K. K. et al. **EmotionSense:** a mobile phones based adaptive platform for experimental social psychology research. Proceedings of the 12th ACM international conference on Ubiquitous computing. [S.l.]: [s.n.]. 2010. p. 281--290.

RANDELL, C.; MULLER, H. **Context awareness by analysing accelerometer data**. Wearable Computers, The Fourth International Symposium on. [S.l.]: [s.n.]. 2000. p. 175--176.

RIBONI, D.; BETTINI, C. **Context-aware activity recognition through a combination of ontological and statistical reasoning**. International Conference on Ubiquitous Intelligence and Computing. [S.l.]: [s.n.]. 2009. p. 39--53.

ROGGEN, D. et al. **OPPORTUNITY:** Towards opportunistic activity and context recognition systems. World of Wireless, Mobile and Multimedia Networks & Workshops, 2009. WoWMoM 2009. IEEE International Symposium on a. [S.l.]: [s.n.]. 2009. p. 1--6.

ROSA, J. H. et al. A multi-temporal context-aware system for competences management. **International Journal of Artificial Intelligence in Education**, v. 25, n. 4, p. 455--492, 2015.

SALIM, F.; HAQUE, U. Urban computing in the wild: A survey on large scale participation and citizen engagement with ubiquitous computing, cyber physical systems, and Internet of Things. **International Journal of Human-Computer Studies**, v. 81, p. 31--48, 2015.

SATYANARAYANAN, M. Pervasive computing: Vision and challenges. **Personal Communications, IEEE**, v. 8, n. 4, p. 10--17, 2001.

SCHILDT, H.; COWARD, D. **Java:** the complete reference. [S.l.]: [s.n.], 2011.

SCHILIT, B. N.; THEIMER, M. M. Disseminating active map information to mobile hosts. **Network, IEEE**, v. 8, n. 5, p. 22--32, 1994.

SCHILIT, B.; ADAMS, N.; WANT, R. **Context-aware computing applications**. Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on. [S.l.]: [s.n.]. 1994. p. 85--90.

SILVA, J. M. et al. Content distribution in trail-aware environments. **Journal of the Brazilian Computer Society**, v. 16, n. 3, p. 163--176, 2010.

SRIKANT, R.; AGRAWAL, R. **Mining sequential patterns:** Generalizations and performance improvements. [S.l.]: [s.n.], 1996.

STRANG, T.; LINNHOFF-POPIEN, C. **A context modeling survey**. Workshop on Advanced Context Modelling. Nottingham/England: [s.n.]. 2004.

TILKOV, S.; VINOSKI, S. Node. js: Using JavaScript to build high-performance network programs. **IEEE Internet Computing**, v. 14, n. 6, p. 80, 2010.

VAN LAERHOVEN, K. **Combining the self-organizing map and k-means clustering for on-line classification of sensor data**. International Conference on Artificial Neural Networks. [S.l.]: [s.n.]. 2001. p. 464--469.

VAPNIK, V. N.; VAPNIK, V. **Statistical learning theory**. [S.l.]: [s.n.], v. 1, 1998.

VIDELA, A.; WILLIAMS, J. J. **RabbitMQ in action**. [S.l.]: [s.n.], 2012.

WAGNER, A.; BARBOSA, J. L. V.; BARBOSA, D. N. F. A model for profile management applied to ubiquitous learning environments. **Expert Systems with Applications**, v. 41, n. 4, p. 2023--2034, 2014.

WEISER, M. The computer for the 21st century. **Scientific american**, v. 265, n. 3, p. 94--104, 1991.

WEISER, M. Some computer science issues in ubiquitous computing. **Communications of the ACM**, v. 36, n. 7, p. 75--84, 1993.

WHITE, T. **Hadoop:** The definitive guide. [S.l.]: [s.n.], 2012.

WIEDEMANN, T. SIMCOP: Um Framework para Análise de Similaridade em Sequências de Contextos, 2014.

WOOD, A. D. et al. Context-aware wireless sensor networks for assisted living and residential monitoring. **Network, IEEE**, v. 22, n. 4, p. 26--33, 2008.

YAN, X.; HAN, J.; AFSHAR, R. **CloSpan:** Mining closed sequential patterns in large datasets. In SDM. [S.l.]: [s.n.]. 2003. p. 166--177.

YIN, J. et al. Human activity recognition based on multiple order temporal information. **Computers & Electrical Engineering**, v. 40, n. 5, p. 1538--1551, 2014.

YURUR, O. et al. Context-awareness for mobile sensing: a survey and future directions, 2014.

ZAKI, M. J. SPADE: An efficient algorithm for mining frequent sequences. **Machine learning**, v. 42, n. 1-2, p. 31--60, 2001.