Universidade Federal do Maranhão
Centro de Ciências Exatas e Tecnologia
Programa de Pós-graduação em Ciência da Computação

MARCELO HENRIQUE MONIER ALVES JÚNIOR

Confidere - Um modelo de confiança para Sistemas Sensíveis ao Contexto voltados ao Domínio da Saúde

São Luís
2015

MARCELO HENRIQUE MONIER ALVES JÚNIOR

Confidere - Um modelo de confiança para Sistemas Sensíveis ao Contexto voltados ao Domínio da Saúde

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da UFMA, como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Samyr Béliche. Vale, Dr

Doutor em Informática - UFMA (Orientador)

Co-orientador: Prof. Francisco José da Silva e Silva, Dr.

(Doutor em Informática - UFMA - (Coorientador))

São Luís

2015

Henrique Monier Alves Júnior, Marcelo

Confidere - Um modelo de confiança para Sistemas Sensíveis ao Contexto voltados ao Domínio da Saúde / Marcelo Henrique Monier Alves Júnior - São Luís, 2015.

73.p

Orientador: Prof. Samyr Béliche. Vale, Dr

Coorientador: Prof. Francisco José da Silva e Silva, Dr. Dissertação (Mestrado) - Universidade Federal do Maranhão, Programa de Pós-Graduação em Ciência da Computação, 2015.

1.Middleware 2.MobileHealthNet 3.Confiabilidade 4.Qualidade de Contexto 5.Confidere. I.Título.

CDU 519.687

MARCELO HENRIQUE MONIER ALVES JÚNIOR

Confidere - Um modelo de confiança para Sistemas Sensíveis ao Contexto voltados ao Domínio da Saúde

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da UFMA, como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciência da Computação.

Aprovado em 26/08/2015

BANCA EXAMINADORA

Prof. Samyr Béliche. Vale, Dr

Doutor em Informática - UFMA (Orientador)

Prof. Francisco José da Silva e Silva, Dr.

(Doutor em Informática - UFMA - (Coorientador))

Prof. Denivaldo Cicero Pavão Lopes, Dr.

(Membro da Banca Examinadora)

Prof. Eveline de Jesus Viana Sá, Dr.

(Membro da Banca Examinadora)

Ione Ribeiro Alves

(In memory)

Agradecimentos

Em primeiro lugar agradeço a Deus, pois é graças a ele mais essa conquista. Aos meus pais e a todos os membros da minha família que ajudaram direta e indiretamente para obtenção desse título. A Silvia e Ionete, minhas tias e verdadeiras mães que sempre estenderam a mão quando eu precisei e por todo carinho e fé que em mim depositaram. A meu primo Maurício Monier Filho que sempre esteve presente nessa caminhada.

A todos os membros do LSD, LAWS e GESEC. Em especial ao professor Francisco Silva pela sua total colaboração, paciência e compartilhamento dos seus conhecimentos durante essa luta e também pela grande amizade formada. Ao Berto de Tácio, uma grande amizade que nasceu no LSD, agradeço-te por todas as palavras de sabedoria e apoio. Ariel Teles pelas palavras fortes e necessárias para o meu crescimento enquanto pesquisador. Daniel Ribeiro e Bruno Moraes por serem verdadeiros amigos e companheiros nas horas mais difíceis dessa caminhada.

Ao professor Luciano Coutinho que também sempre esteve presente para sanar todas as dúvidas referentes à pesquisa. Ao meu orientador, professor Samyr Vale pela sua fundamental contribuição no desenvolvimento desse trabalho.

Elza Bernardes, minha grande companheira, e toda sua família por me apoiar nos momentos mais complicados. Todos foram de fundamental importância para a conclusão desse trabalho.

Ao Programa de Pós-Graduação em Ciências da Computação - UFMA e a Fundação de Amparo à Pesquisa e ao Desenvolvimento Científico e Tecnológico do Maranhão, que financiaram toda a pesquisa desenvolvida durante esses dois anos.

E por fim, a todos os amigos que aqui não foram citados. Essa conquista é por vocês e para vocês.

"Se enxerguei mais longe, foi porque estava sobre os ombros de gigantes."(Isaac Newton).

Resumo

Saúde Móvel ou *m-health* é a denominação dada à pratica de medicina e cuidados da saúde através de dispositivos móveis. Essa dissertação está sob o escopo do projeto *MobileHealthNet* que fornece um *middleware* sensível ao contexto focado na criação de aplicações da saúde em Redes Sociais Móveis (RSMs). RSMs são ambientes colaborativos com grande variedade de diferentes contextos. Uma das deficiências do projeto MobileHealthNet, é a pequena quantidade de Qualidade de Contexto (QoC) em seus componentes, em especial o Provedor de Contexto (CP), ocasionando inúmeros problemas, assim como a tomada de decisão equivocada de um médico devido aos dados coletados de um sensor com baixa precisão. Esse trabalho possui o foco na implementação de parâmetros de QoC na infraestrutura do *MobileHealthNet*. O bjetivo principal do trabalho é fornecer suporte ao *Trustworthiness* (Confiabilidade), sugerimos um modelo de confiança denominado de *Confidere* para detectar a confiança do componente CP. Por fim, é realizada uma avaliação do *Confidere* com foco na eficácia em detectar comportamentos maliciosos por parte dos CPs.

Palavras-chave: Middleware; Mobilehealthnet; Confiabilidade; Qualidade de Contexto; Confidere.

Abstract

Mobile health is the name given to the practice of medicine and health care through mobile device applications. This work is under the scope of the MobileHealthNet project which provides a context-aware middleware focused on creating health care applications in Mobile Social Networks (MSNs). MSNs are collaborative environments with large variety of different contexts. One of the drawbacks of MobileHealthNet is the absence of Quality of Context (QoC) between consumers and producers of data, which can cause problems, such as wrong decisions by physicians, due to data collected from a sensor with low accuracy. This paper focuses on implementing QoC parameters in the MobileHealthNet infrastructure. Our main goal is to provide support to Trustworthiness, suggest a trust model named as Confidere to detect the trust of the Context Provider (CP) component. Finally, a evaluation of Confidere in the context of occurrence of CP's malicious behavior in different contexts.

Keywords: Middleware; Mobilehealthnet; Trustworthiness; Quality of Context; Confidere.

Lista de Figuras

Conteúdo

1 Introdução

Com o aumento da utilização de dispositivos móveis, aumentou também a demanda por aplicações práticas desses dispositivos em diversas áreas, tais como na saúde, na educação e no sistema financeiro, por exemplo. Devido à facilidade de acesso à informação de qualquer lugar e em qualquer momento com o uso desses mecanismos, naturalmente cresceu bastante o interesse das pessoas por esses dispositivos. Outro fator relevante diz respeito à acessibilidade no mercado, tendo em vista o fato de que tais dispositivos estão cada vez mais baratos ao mesmo tempo em que aumenta o seu poder computacional, a exemplo da expansividade no armazenamento, de uma maior quantidade de sensores internos, de um maior poder de processamento de dados, diferentes interfaces de redes etc. Dessa forma, é possível que se executem aplicações cada vez mais robustas, como as aplicações ubíquas.

Um dos primeiros a trazer à tona o termo "*Computação Ubíqua*" foi Weiser (WEISER, 1991), definindo-o como um modelo de interação homem-máquina que visa melhorar o uso do computador, através da disponibilização de inúmeros dispositivos interagindo entre si de maneira transparente aos usuários. Aplicações ubíquas são adaptáveis ao ambiente do usuário, pois não necessitam da entrada explicita do mesmo para prover serviços, funções e informações.

Quem também aderiu a esse tipo de acesso foram as redes sociais online, pois, com a facilidade de conexão, ficaram mais fáceis o compartilhamento de informações e a conectividade entre os usuários. Como resultado, uma nova tendência operou-se no campo da sociabilidade entre os indivíduos, as chamadas Redes Sociais Móveis (RSM), que atraíram consideravelmente a atenção das comunidades acadêmicas e da indústria (KARAM; MOHAMED, 2012). Essas redes são consideradas uma subclasse de redes sociais, às quais os usuários móveis podem acessar, publicar ou compartilhar conteúdo gerado ou obtido através de sensores no dispositivo móvel para a interação com os seus contatos na rede social(TELES et al., 2013).

Teles et. al. (TELES; SILVA; BATISTA, 2013), definem RSM como uma combinação de três áreas de conhecimento: computação móvel, redes sociais e ciência

de contexto. Podemos observar essa definição na figura 1.1.

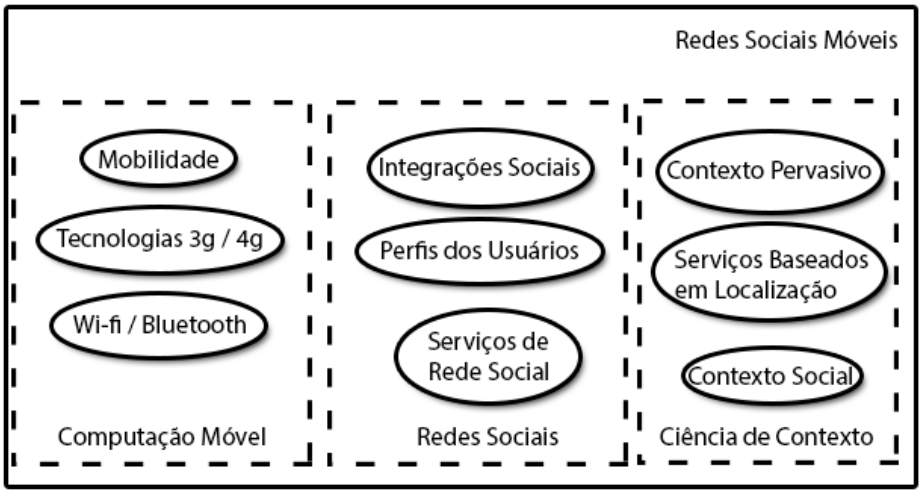

Figura 1.1: Definição de RSMs baseado em [Teles et al. 2013]

O suporte à mobilidade é obtido pelos dispositivos portáteis e pela conectividade sem fio. Assim, o usuário é capaz de permanecer sempre online. Redes Sociais possibilitam a criação de perfis e a interação entre eles, podendo representar indivíduos, sistemas ou organizações. A Ciência de Contexto é responsável por adaptar as aplicações e funcionalidades das redes sociais, permitindo, dessa forma, que se ofereçam recursos de acordo com informações de contexto.

Nesse cenário, surgiram vários tipos de aplicações para RSMs. Estas definem-se como publicadoras de informações de contexto às Redes Sociais, pois os dispositivos móveis são capazes de sensorear dados físicos do ambiente e, com isso, possibilitar que uma aplicação possa combinar dados de contexto para inferir uma situação do usuário. Diferentes áreas começaram a criar aplicações desse tipo, incluindo, como visto, as que são voltadas ao domínio da saúde.

Essas aplicações são conhecidas como *m-health* ou *Mobile Health* (Saúde Móvel). Tal denominação é dada à prática de medicina e cuidados da saúde através de dispositivos móveis (ISTEPANIAN ROBERT, 2006), que, nesse sentido, conferem uma nova perspectiva à relação profissional existente nesse campo, por uma simples razão: aplicações de *m-health* possibilitam aos profissionais de saúde o acompanhamento dos pacientes sem a necessidade de um espaço físico em comum. Em outras palavras, é possível que médicos, enfermeiros e outros profissionais procedam ao tratamento/acompanhamento de seus pacientes à distância, através da utilização desses dispositivos.

Esse novo enfoque à relação profissional implica considerável redução de custos, permitindo uma posição mais confortável aos envolvidos na relação, de maneira que é perfeitamente viável a hipótese de uma consulta médica na qual o profissional da saúde e o paciente estejam cada qual em sua casa, ou em localidades distintas. Um exemplo para esse tipo de situação são as aplicações de RSMs, pois podem prover o compartilhamento de informações, integração e colaboração social entre todos os envolvidos no processo de atendimento à saúde (BATISTA, 2013).

Nesse contexto, foi criado o projeto *Mobile Social Networks for Health Care in Offside Regions* (MobileHealthNet), que se encontra em desenvolvimento através da parceria entre o Laboratorio de Sistemas Distribuidos da Universidade Federal do Maranhao (UFMA) e *o Laboratory for Advanced Collaboration* da Pontifícia Universidade Católica do Rio de Janeiro (PUC - Rio). Esse projeto tem por objetivo geral avançar o estado da arte em sistemas de *middleware* para RSMs voltadas para a área da saúde. Este trabalho está inserido no projeto por, principalmente, utilizar recursos do *middleware* que estão em desenvolvimento. Ele será descrito em detalhes no capítulo 2.

1.1 Caracterização do problema

Os serviços de contexto disponibilizam uma infraestrutura de suporte para coleta, gerenciamento e distribuição das informações de contexto sobre uma série de temas, que podem estar relacionados ao usuário, a objetos ou até mesmo ao ambiente. Tais serviços adquirem informações de contexto de várias fontes - coletadas por terceiros -, as quais geralmente fornecem os dados contextuais.

Exemplificando: considere a "temperatura"no local atual do usuário móvel. Essa informação pode ser obtida diretamente do dispositivo móvel do usuário ou através de um serviço web que disponibiliza a temperatura local da região. Nesse sentido, um dos grandes problemas desse tipo de serviço é que suas informações de contexto não possuem um alto nível de confiabilidade, a exemplo de um provedor que coleta dados alterados (na hipótese de um sensor, que registra a temperatura do ambiente de uma sala, poder repassar dados alterados para a infraestrutura, em razão de se encontrar fisicamente próximo a uma grande fonte de calor).

Qualidade de Contexto (QoC) é qualquer informação que descreve a qualidade

da informação que é usada como informação de contexto. Assim, QoC refere-se à informação, não ao processo e nem ao componente de hardware que possivelmente fornece as informações (BUCHHOLZ; SCHIFFERS, 2003). A proposta desta dissertação é prover QoC em middlewares que trabalham com informações de contexto, especificamente o MobileHealthNet, já que a nossa proposta encontra-se em conformidade com as diretrizes do projeto MobileHealthNet.

Um dos inconvenientes do MobileHealthNet é que suas informações de contexto não possuem QoC, podendo gerar certos problemas, a exemplo de uma decisão tomada erroneamente por um médico, em razão de um dado coletado de um sensor com baixo nível de precisão. As aplicações geradas pelo *middleware* necessitam de informações dotadas de alto nível de confiança, pois que podem envolver circuntâncias, de certa forma, delicadas, como um sistema de acompanhamento de pacientes com problemas cardíacos.

Esses sistemas não podem receber dados sem QoC, pois se não existir um alto nível de precisão das informações, o médico poderá executar uma ação equivocada, atingindo diretamente o estado de saúde do paciente, podendo-se falar, inclusive, em risco de morte. Outra grande dificuldade que merece ser lembrada é a qualidade das redes móveis, pois existem várias limitações, como a largura da banda, intermitência do sinal, menor área de cobertura, dentre outros. No entanto, será aplicado um protocolo de comunicação bem robusto, chamado *Mobile Reliable UDP* (MRUDP) (DAVID et al., 2012b). Apesar do MRUDP ser confiável, ele não possui parâmetros de QoC.

Saber o nível de importância de cada informação de contexto (por exemplo, uma aplicação pode preferir um nível de precisão maior para o batimento cardíaco do paciente do que a precisão da localização do mesmo) é fundamental. É preciso que se tenha conhecimento sobre a real necessidade de cada aplicação e se as informações de contexto consumidas atendem aos níveis de qualidade solicitados. No MobileHealthNet, uma de suas camadas é o context service, responsável por todo o gerenciamento das informação de contexto na infraestrutura. Nesse caso, utilizaremos os parâmetros de QoC para que a infraestrutura possa gerar aplicações com critério de confiança significativo.

Portanto, para que haja um nível substancial na qualidade da informação, é necessário que o parâmetro de QoC, em específico o *Trustworthiness* definido na seção 2, seja implementado na infraestrutura do *middleware MobileHealthNet*. Um dos nossos principais desafios é especificar o nível de confiança que o provedor de contexto (CP), terá

em relação ao contexto do paciente.

Os dados fornecidos por esse componente implicam diretamente na tomada de decisão dos profissionais da saúde para com o paciente em acompanhamento, causando assim alteração no estado de saúde do mesmo. Todas as questões vistas nesta seção comprovam, por fim, a complexidade e a necessidade do *Trustworthiness* em sistemas voltados para o domínio da saúde.

1.2 Objetivos

O objetivo geral deste trabalho é o desenvolvimento do suporte para o parâmetro de QoC Confiabilidade (*Trustworthiness*) a ser incorporado ao *middleware MobileHealthNet*. Esta dissertacao tem como objetivos específicos:

- Investigar o estado da arte no provimento de Qualidade de Contexto para aplicações móveis voltadas ao domínio da saúde;
- Desenvolvimento de uma proposta de modelo de confiança que possa representar a confiabilidade dos componentes em sistemas sensíveis ao contexto;
- Implementar o modelo proposto no *middleware MHNCS*;
- Tratar aspectos de segurança para a garantia da troca de informações seguras dentro do *middleware MHNCS*;
- Testar e avaliar o desempenho do suporte de *Trsutworthiness* a ser desenvolvido.

1.3 Organização da Dissertação

Esta dissertação organiza-se da seguinte forma: o capítulo 2 detalha a revisão bibliográfica, abordndoa-se a arquitetura do projeto *MobileHealthNet* assim como os conceitos de Ciência de Contexto e Qualidade de Contexto e um estudo sobre confiança computacional. No capítulo 3, são expostos os trabalhos relacionados.Já o capítulo 4 trata do modelo de confiança proposto, da arquitetura da aplicação e de detalhes da implementação do *Confidere* no projeto *MobileHealthNet*. No capítulo 5 é apresentada a

análise dos resultados dos experimentos realizados e, finalmente, temos o capítulo 6, em que serão apresentadas as conclusões e os trabalhos futuros.

2 Fundamentação teórica

2.1 Ciência de Contexto

Dey (DEY, 2001) define contexto como sendo qualquer informação que pode ser usada para caracterizar a situação de uma entidade. Uma entidade pode ser uma pessoa, lugar ou objeto que seja considerado relevante na interação do usuário com a aplicação. Sistemas sensíveis ao contexto são habilitados para adaptar suas funcionalidades e comportamento de acordo com o contexto atual do usuário, sem sua explícita intervenção.

Bolchini (BOLCHINI et al., 2009) descreve as informações de contexto como um conjunto de variáveis que pode ser de interesse para uma entidade, podendo, também, influenciar em suas ações. Por meio desse tipo de informação, é possível o desenvolvimento de sistemas computacionais, os quais podem se reconfigurar ou adaptar a uma determinada situação, ou recomendar ações a partir de análises de informações coletadas do ambiente.

Para Chen (CHEN; KOTZ, 2002), as informações de contexto classificam-se em quatro tipos:

- **Contexto Físico**: informações sobre o mundo real obtidas por meio de sensores. Por exemplo, sensor de luminosidade, sensor de ruido, temperatura, luminosidade e localização;
- **Contexto Computacional**: informações sobre um sistema computacional, como os seus recursos e características. Por exemplo, nível de bateria, consumo de memória ou processamento e conexões de rede disponíveis;
- **Contexto do Usuário**: informações que caracterizam o usuário, tais como: estado emocional, localização e atividade atual;
- **Contexto de Tempo**: informações relacionadas ao tempo de uma atividade real ou virtual. Está relacionada à dimensão do tempo, como a hora do dia, dia da semana, mês, ano ou uma estação climática.

A utilização de informações de contexto permite que desenvolvedores de sistemas possam enriquecer a usabilidade de sua aplicação e, assim, o sistema sensível ao contexto pode reagir a determinadas situações sem a necessidade da interação com o usuário. Por exemplo, caso o sistema detecte que o nível de bateria está baixo, ele pode realizar ações para economizá-la, seja reduzindo a luminosidade do visor do dispositivo, seja desativando as interfaces de rede ou sensores que não estão em uso no momento.

O Contexto Pervasivo é aquele obtido a partir de sensores de hardware nos dispositivos móveis, os quais podem ser de vários tipos: luminosidade, visual (câmera), áudio, movimento ou acelerômetro, localização, toque, temperatura, físicos (bio-sensores), dentre outros (BALDAUF; DUSTDAR; ROSENBERG, 2007). Dados gerados a partir desses sensores podem ser usados individualmente ou de forma combinada para inferir a situação do usuário. Essa situação pode ser, por exemplo, a (in)disponibilidade do usuário para realizar alguma atividade, seu estado de saúde ou o lugar em que se encontra (restaurante, residência ou local de trabalho).

O termo Contexto Social é utilizado para caracterizar as possíveis formas de relacionamento e de interações entre pessoas, intermediadas ou não por alguma tecnologia de comunicação (WIBISONO; ZASLAVSKY; LING, 2008). A expressão envolve o ambiente social do usuário (uma festa ou uma reunião) e a relação que pode ser estabelecida com outros usuários. A noção de contexto social deve levar em consideração tanto as experiências no mundo real quanto no virtual. Informações de contexto social podem ser extraídas por meio de redes sociais, sensores ou formulários, sendo - tais informações - aspectos de contexto de alto nível relacionados com a dimensão social dos usuários, tais como o perfil do usuário, pessoas próximas, e sua atual situação social (ADAMS; PHUNG; VENKATESH, 2008).

Como visto, os conceitos de rede social se unem à computação móvel e à ciência de contexto e, a partir disso, Lubke et. al.,(LUBKE; SCHUSTER; SCHILL, 2011) desenvolveram o conceito de Contexto Social Pervasivo. Trata-se, basicamente, de um conjunto de informações decorrentes de interações diretas e indiretas entre pessoas que carregam dispositivos móveis equipados com sensores e que estejam conectadas através de uma mesma rede social. Os autores ainda classificaram e diferenciaram várias formas em que o contexto social pervasivo pode ser utilizado baseados nas *W5H Questions*, como visto abaixo:

- Quem - ***W**ho*: expressa quem são os participantes envolvidos no consumo e na produção das informações de contexto;

- O que - ***W**hat*: diz respeito a qual tipo de contexto é utilizado ou se é importante para a aplicação;

- Onde - ***W**here*: relacionado ao espaço físico onde os laços ou interações sociais são estabelecidos;

- Quando - ***W**hen*: caracterização das interações entre usuários e as informações de contexto que eles produziram em uma perspectiva temporal;

- Por que - ***W**hy*: expressa o porquê de uma informação de contexto ser usada, determinando a causa ou razão de sua utilização pela aplicação. Nesse caso, isso é bem relacionado ao objetivo da aplicação;

- Como - ***H**ow*: expressa como a informação de contexto (originada a partir do mundo real, mundo virtual ou de ambos) pode influenciar ou comprometer aplicações.

2.2 Representação de Contexto

Informações de contexto devem ser formatadas utilizando modelos de representação de informações contextuais. Devido às características de dinamismo e heterogeneidade em ambientes pervasivos, o modelo de representação de dados de contexto deve ser robusto o suficiente para representar diversos tipos de dados de contextuais. Sendo assim, diferentes aplicações sensíveis a contexto podem trocar informações utilizando um mesmo modelo de representação. Para Henricksen (HENRICKSEN; INDULSKA; RAKOTONIRAINY, 2002) e Held (HELD; BUCHHOLZ; SCHILL, 2002) os modelos de representação de dados de contexto devem possuir as seguintes características:

- Estruturada: permite filtrar ou extrair eficientemente as informações de contexto que são relevantes para uma aplicação. Esta característica permite reduzir a ambiguidade de dados;

- Intercambiável: a informação contextual pode ser trocada entre as diferentes aplicações, bem como os diferentes componentes de uma mesma aplicação;

- Composta/Decomposta: compor/decompor informações de contexto é útil para prover distribuição de forma eficiente de informações entre aplicações. Por exemplo, no caso de uma atualização da informação de contexto, esta pode ser enviada apenas a parte da informação que foi modificada, evitando que informações redundantes sejam enviadas;

- Extensível: permite que novos parâmetros sejam adicionados ao modelo de representação de contexto, visto que, isso é de extrema importância devido à dinamicidade que sistemas sensíveis a contexto necessitam. Para tal, é necessário que seja utilizado um modelo que permita a representação dinâmica de dados de contexto;

- Padronizada: como a informação pode vir de diferentes entidades, é fundamental que a informação seja representada de forma padronizada. O mesmo tipo de informação deve ser representada de maneira única, isto é, aplicações diferentes devem representar a mesma informação em um formato pré-estabelecido pelo modelo.

Em Strang et. al.(STRANG; LINNHOFF-POPIEN, 2004), os modelos contextuais são classificados por um esquema de estrutura de dados, que permitem a troca de informações contextuais no respectivo sistema. Existem casos que o mesmo modelo contextual pode ser classificado em mais de uma categoria.

2.3 Modelos de Representação

A representação de contexto está diretamente associada a mecanismos de descoberta e monitoração do contexto. É necessário disponibilizar uma representação das informações sobre seu tipo, atributos e características funcionais para que o recurso possa ser utilizado pelos clientes da infraestrutura. De acordo com Strang e Linnhoff-Popien em (STRANG; LINNHOFF-POPIEN, 2004), os modelos contextuais são classificados em:

2.3.1 Par Chave-Valor

A Figura 2.1 mostra um exemplo do modelo Par Chave-Valor. Pode-se perceber que é um modelo de estrutura simples. Os dados de contexto são representados

através da utilização de chaves como atributo de identificação e um valor associado a ela (SOARES, 2010).

CHAVE	VALOR
DATA	01/01/2014 - 31/12/2014
HORA	00:00 - 23:59
LOCALIZAÇÃO	-2° 30.692', -44° 18.268'

AÇÃO
Exibir menssagem "Bem vindo à sua casa!"

Figura 2.1: Exemplo de contexto representado por par chave-valor

Seu processo de recuperação das informação contextuais baseia-se em uma busca linear com a combinação exata de nomes(chave ou *key*). Devido a sua representação da informação ser simplória, esse modelo não é recomendado para aplicações com estruturas de dados complexas, pois não tem suporte a hierarquias entre os atributos (MIRAOUI et al., 2011).

2.3.2 Modelo de Esquema de Marcação

O modelo de marcação permite a representação das informações contextuais de modo hierárquico. Esse modelo representa as informações de modo textual possuindo regras rígidas de formatação, sendo que a principal estrutura de formatação são *tags* que permitem delimitar e agrupar conjuntos de dados.

Nesse modelo, o XML (*eXtensible Markup Language*) é usado como padrão para a modelagem de informações contextuais, pois facilita o compartilhamento de dados de contexto por diferentes aplicações (VIEIRA; TEDESCO; SALGADO, 2011). No trecho XML, observa-se o armazenamento do seguinte contexto: "O professor Francisco Silva está no LSD em uma sexta pela manhã".

Listing 2.1: Exemplo de documento XML

```
<context>
<primitive>
<whoS><who type = "teacher" value = "Francisco Silva" /> </whoS>
<whereS><where type = "room" value="LSD" /> </whereS>
<whenS><when type = "date" value = "Sexta AM" qualifier = "date" /> </whenS>
</primitive>
</context>
```

2.3.3 Modelo Orientado a Objetos

Uma das grandes vantagens desse modelo é uso do encapsulamento, herança e reusabilidade, pois a estruturação das informações contextuais é feita através de hierarquias de classes, como se pode observar na Figura 2.2. Segundo Pessoa (PESSOA, 2006), esse modelo não explora a descrição semântica das informações contextuais e, por isso, há uma dificuldade na implementação de algoritmos que realizem a inferência de novos dados de contextos a partir dele.

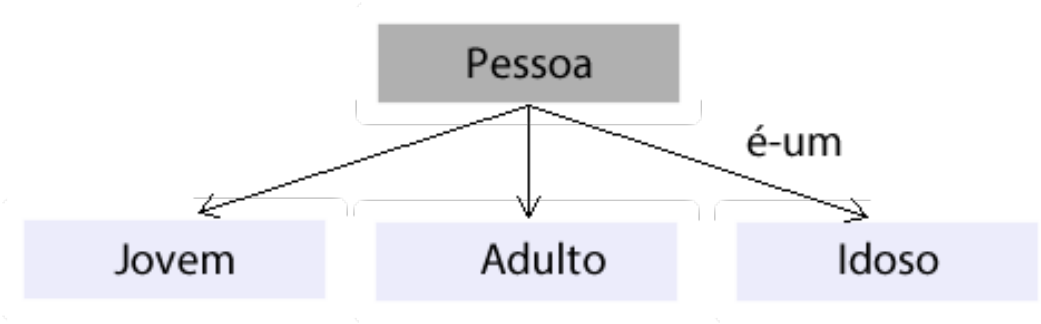

Figura 2.2: Exemplo de contexto representado por um modelo orientado a objetos (SOARES, 2010)

Em Vieira (VIEIRA; TEDESCO, 2006), afirma-se que neste modelo de representação concentram-se no nível de interfaces, enquanto os detalhes de processamento encontram-se encapsulados nos objetos. Outra característica interessante é poder trabalhar com composição distribuída. A forma de tratamento das informações contextuais (classes), as instâncias que são atualizadas ou até mesmo as novas instâncias que são atribuídas (objetos) no sistema, e distribuída entre os componentes do sistema.(?).

2.3.4 Modelo Baseado em Lógica

Nesse modelo a definição das condições de inferência de um fato, é por meio da lógica booleana sobre um conjunto de premissas. Normalmente, aplicações que utilizam esse modelo focam no mecanismo de inferência de fatos e também no fornecimento de formalismo para especificar as regras de inferência. No modelo baseado em Lógica, a informação contextual é adicionada, atualizada e excluída de um sistema baseado em expressões sobre um conjunto de fatos (FILHO, 2010). Sua desvantagem desse modelo é não oferecer praticidade e ser complexo para representar e processar grandes conjuntos de dados em tempo real.

Existem alguns tipos de abordagem para modelagem de contexto baseados em lógica. Uma das primeiras pesquisas e publicações foram feitas visando tratar informações contextuais como entidades matemáticas abstratas, com propriedades úteis para inteligência artificial. Outra abordagem desse modelo é feita em Chetan et. al. (CHETAN; RANGANATHAN; CAMPBELL, 2005), na qual a representação de contexto é feita através do uso de predicados de primeira classe.

2.3.5 Modelo baseado em Ontologias

A ontologia é usada para representar informações complexas por ter capacidade de expressar relações complexas entre informações, como por exemplo a validação dos dados que é normalmente expressa pela imposição de restrições da ontologia (BELLAVISTA et al., 2012). Uma das grande vantagens desse modelo é oferecer o reuso e compartilhamento de informações.

Ontologias permitem representar conhecimento sobre o mundo de forma processável por sistemas computacionais (VIEIRA; TEDESCO; SALGADO, 2009). Esse modelo é uma das formas mais robustas para mapear o conhecimento sobre dados de um determinado contexto.

Segundo Soares (SOARES, 2010), a ontologia permite a criação de aplicações complexas, no entanto, é necessário a estruturação de vocabulários e a associação da semântica entre informações de diferentes domínios. O trabalho (MIRAOUI et al., 2011) lista as duas principais vantagens desse modelo: i) melhora o compartilhamento de dados, eliminando as fontes de ambiguidades (o mesmo significado para um determinado conceito); e ii) possibilita criar raciocínio lógico facilmente, usando uma lógica descritiva relacionada (deduzir fatos implícitos de contexto, ou seja, modelar fatos de contexto através de experiências acumuladas pelo sistema).

Diversos modelos de contexto têm sido criados para atender às necessidades de determinadas aplicações, como as formas de representação de dados de contexto que atendam a necessidades de diversos domínios de informação. Na literatura, os autores buscam um modelo único e completo, devendo ser simples, flexível e bem expressivo, dando suporte a representação de qualquer informação contextual.

Em Vieira (VIEIRA; TEDESCO; SALGADO, 2009), afirma-se que cada técnica

usada para representar contexto apresenta vantagens e desvantagens. Pode-se concluir que não existe ainda um único modelo de representação que possa ser aplicado a todos os tipos de sistemas sensíveis ao contexto.

2.4 Qualidade de Contexto

Conforme afirmado na seção 2, o termo QoC foi definido primeiramente por Buchholz et. al. (BUCHHOLZ; SCHIFFERS, 2003). Nesse trabalho, foram apresentados *trust-worthiness, probability of correctness, precision, resolution e up-to-datedness* como importantes parâmetros de QoC. Além disso, o trabalho comparou o referido conceito com a Qualidade de Serviço (QoS), categoria esta que provê informação sobre o desempenho de um serviço. Equiparou-se, igualmente, a referida expressão - QoC - com a Qualidade de Dispositivo (QoD), a qual, por seu turno, qualifica a capacidade e propriedades técnicas de um dispositivo. Após essa relação, os autores ressaltam que essas três métricas, embora independentes, podem influenciar-se mutuamente.

Conceituação diversa de QoC é feita por Krause et. al. (KRAUSE; HOCHSTATTER, 2005), os quais afirmam ser o termo qualquer dado inerente que descreve a informação de contexto, a qual pode ser utilizada para determinar o valor de uma informação para uma aplicação específica. Identificam-se as fontes de parâmetros de QoC como as características do sensor, o valor declarado pela própria informação de contexto, verificação específica da situação e a granularidade do formato de representação.

Uma nova definição de QoC é feita por Manzoor et. al.(MANZOOR; TRUONG; DUSTDAR, 2011). Afirma o autor que o termo indica o grau de conformidade da coleta de contexto pelo sensor para a situação prevalente do ambiente e as exigências de um consumidor de contexto específico. Na figura 2.4 Manzoor (MANZOOR; TRUONG; DUSTDAR, 2011) cria um novo modelo de processamento de QoC. Tal modelo possui três diferentes camadas para o processamento das informações com QoC.

A camada mais baixa é a origem da QoC, formada pelos dados utilizados pela camada superior. Compõe-se de três sub-categorias, a saber: i) características do sensor - são informações sobre o sensor que podem afetar a qualidade da informação de contexto fornecido pelo dispositivo, como a acurácia, precisão, granularidade, período de tempo, estado do sensor e o alcance dele; ii) medição do contexto - mostra as informações

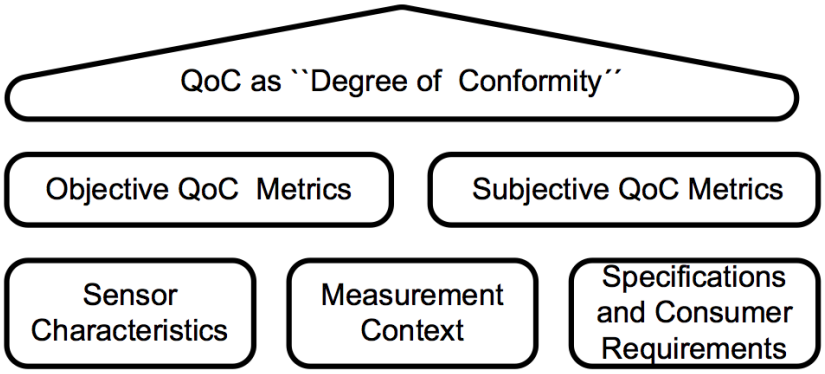

Figura 2.3: Modelo de processamento de QoC (MANZOOR; TRUONG; DUSTDAR, 2011)

relacionadas a uma medição específica, a exemplo do tempo de medição, do sensor de localização, das informações de localização de uma entidade, em suma, referentes aos atributos específicos do contexto de um objeto; e iii) especificações e exigências do consumidor - o consumidor faz um detalhamento de suas exigências sobre a qualidade das informações de contexto, como o tempo de validade da informação, atributos necessários, valor crítico e nível de acesso.

A camada superior é representada por dois tipos de métricas: as métricas de QoC objetivas e as de QoC subjetivas. As primeiras demonstram a qualidade de contexto como uma quantidade independente e seu cálculo envolve características do sensor e medição do contexto; as subjetivas, a qualidade do contexto para utilização de um consumidor específico para uma determinada finalidade.

Nas subseções a seguir detalharemos os principais parâmetros de qualidade de contexto que podem ser encontrados na literatura.

2.4.1 Accuracy

Manzoor (MANZOOR, 2010) afirma que *accuracy* é o grau de exatidão do contexto ou a capacidade do sensor em medir a quantidade aproximada ao valor real. (WEBSTER, 1999) assinala que *inaccuracy*, ou erro absoluto de um sensor físico, pode ser calculado pela diferença entre o valor real e o valor mensurado pelo sensor. Temos, então, a seguinte equação,

$$E = T - M$$

na qual, E é o erro na medição, T é o valor real e M é o valor mensurado pelo sensor. Geralmente, o valor real é acordado antes da medição ou em alguns casos é uma verdade absoluta. Já a *accuracy* de um sensor físico, tem seu valor calculado através da equação,

$$Accuracy = 1 - \frac{|E|}{T}$$

sendo E o valor da *inaccuracy* e T, o valor real. Entende-se que a *accuracy* é calculada subtraindo o erro relativo de 1. Outra *accuracy* importante nesta definição é a do sensor virtual, o qual pode ser calculado com a seguinte equação

$$Accuracy = \frac{N\acute{u}mero\ de\ inst\hat{a}ncias\ classificadas\ como\ corretas}{N\acute{u}mero\ total\ de\ inst\hat{a}ncias}$$

2.4.2 Precision

Segundo Manzoor (MANZOOR, 2010), precision é o grau de exatidão de uma medição. Isto indica a capacidade de um sensor para dar a mesma leitura que a mesma quantidade de medição sob as mesmas condições. Ao contrário do parâmetro accuracy, que apresenta a proximidade de uma medição do valor real, precision apresenta as proximidades sucessivas das leituras dos sensores da mesma quantidade sob as mesmas condições. Um exemplo de teste de precision é sob o sensor físico, o qual pode ser medido através da repetição dos ensaios em uma série de vezes e examinando a variação dos dados. Manzoor (MANZOOR, 2010) utiliza a seguinte equação para avaliar esse parâmetro.

$$Precision = \frac{n\acute{u}mero\ de\ positivos\ verdadeiros}{n\acute{u}mero\ total\ de\ positivos\ verdadeiros\ e\ positivos\ falsos} \tag{2.4}$$

Na equação, o número de positivos verdadeiros representa os casos que tenham sido corretamente reconhecidos como positivos e os positivos falsos, que tenham sido reconhecidos incorretamente como positivos.

2.4.3 Probability of Correctness

Segundo Buchholz et. al. (BUCHHOLZ; SCHIFFERS, 2003), esse parâmetro mede a probabilidade de uma parte da informação de contexto estar correta. Suponha que exista uma sala com uma rede de sensores de temperatura. Um desses sensores pode falhar e começar a mandar a informação errada, por exemplo, medir 50°C, enquanto o valor correto é de 25°C. Com a utilização deste parâmetro, a fonte de informação de contexto original calcula quantas vezes esse sensor irá mandar a informação errada para o provedor de contexto por causa de problemas internos.

Já Huebscher et. al.,(HUEBSCHER; MCCANN, 2004), por sua vez, concluem, quanto à medição, através da análise da postura de uma pessoa (em pé, sentado, deitado no chão em perigo), que o parâmetro *Probability of Correctness* tem resultado diferente quando os tipos de sensores são diferentes, como a utilização de uma câmera de vídeo irá expressar um resultado diferente de um sensor de movimento. Os autores também afirmam que os parâmetros *Trust-worthiness* e *Probability of Correctness* são similares. Entretanto, enquanto a *Probability of Correctness* é fornecida pelo provedor de contexto, a *Trust-worthiness* é fornecida por agente externo para o provedor de contexto.

2.4.4 Temporal Resolution

Sheikh et. al.(Sheikh; Wegdam; Sinderen, 2007), definem este parâmetro como o período de tempo para que uma única instância de informação de contexto seja aplicável. Isso varia devido a duas principais razões: a primeira é que a fonte da informação de contexto pode ser limitada pela sua frequência de coleta (por exemplo, a coleta de temperatura de uma sala é realizada a cada oito horas, então a informação é valida por um período de oito horas após a coleta). A segunda limitação pode ser para proteger a privacidade do usuário (a entrada de um funcionário em uma sala pode ter uma precisão maior do que a saída dele desta sala). A precisão pode ser ofuscada, para garantir a privacidade do funcionário.

2.4.5 Spatial Resolution

Segundo Sheikh et. al. (Sheikh; Wegdam; Sinderen, 2007), é a precisão com que é expressa a área física para a qual uma instância de informação de contexto é aplicável.

Assim, ela refere-se à área de um espaço físico a que a informação de contexto está associada. Por exemplo, quando dizemos que a temperatura da cidade de São Luís está em 30ºC, isso não quer dizer que toda cidade está com a mesma temperatura.

Da mesma forma, quando se tem um conjunto de sensores em uma sala, tendo-se a temperatura maior registrada em um sensor localizado ao lado da janela e a temperatura menor, em um sensor ao lado do ar condicionado, deve-se atentar para o fato de que ambas as medições de temperatura podem não refletir corretamente a temperatura real do ambiente. Uma característica importante desse parâmetro é a garantia da privacidade do usuário em uma determinada região, por exemplo, em um sistema de monitoramento de localização, a informação será transmitida com um grau menor de precisão, então a informação será exibida dentro de um raio aceitável da região, garantindo, dessa forma, a privacidade do usuário.

2.4.6 Trustworthiness

Segundo Russell (HARDIN, 2002) *trustworthiness*, está relacionada ao indivíduo que recebe a confiança, ou seja, ao Trustee. A confiança é a expectativa que o Trustor terá no comportamento do Trustee, isto é, se o Trustee irá se portar de maneira que ele considera confiável. Não nos aprofundaremos aqui neste assunto, pois separamos um seção para abordar a confiança.

2.5 Confiança computacional

2.5.1 Definição

Woolthuis (WOOLTHUIS B HILLEBRAND, 2005) afirma que a psicologia descreve confiança como uma relação em três partes: A confia em B a respeito de X, no qual A representa quem dará a confiança (Trustor); B, quem vai receber a confiança (Trustee); e X, o assunto de confiança. Exemplificando: o usuário (A) confia em um sensor B para lhe fornecer a localização (X), no momento em que o mesmo está parado. Entretanto, não confia no mesmo para lhe fornecer a mesma informação quando ele está em movimento.

Para que haja um depósito de confiança no Trustee, deve-se ter a oportunidade de trair a confiança do trustor, e, ao mesmo tempo, não querer fazê-lo (WOOLTHUIS

B HILLEBRAND, 2005). Nesse sentido, se o Trustor quiser garantir a prestação de serviço do Trustee, ele irá introduzir um elemento chamado "controle". Após tal procedimento, a confiança será removida do relacionamento. O controle é todo elemento utilizado para forçar o trustee a ser confiável, limitando ou removendo por completo a oportunidade de traição do mesmo.

Suponhamos: Uma empresa X contrata um serviço de armazenamento em nuvem de uma empresa Y, para suas informações. Nesse momento, a empresa X está confiando na empresa Y para gerenciar suas informações. No entanto, no contrato consta que, se a empresa Y fornecer ou alterar essas informações, a mesma pagará uma multa de U$ 20.000.000. Essa multa representa o controle de confiança que a empresa X terá com a empresa Y.

Outra forma de estabelecer a confiança é através da reputação. Essa é fornecida por meio da opinião de terceiros, ou seja, a confiança dada ao Trustee irá depender das informações passadas ao Trustor por terceiros que já com ele interagiram anteriormente.

Uma outra forma de confiança constante da literatura é a dependência. Trata-se de uma forma mais fraca de confiança, na qual a oportunidade que o Trustee tem de trair o Trustor não existe. Ela ocorre quando a quebra de confiança resulta na decepção do Trustor, não chegando, entretanto, o mesmo a prejudicar-se. Podemos exemplificar da seguinte forma: dois indivíduos concordam em se encontrar para uma partida de futebol e um deles não aparece, então caracteriza-se dependência. Porém, emprestar uma quantia em dinheiro para outro indivíduo, indica um caso de confiança.

2.5.2 Condições de confiança

Na literatura o conceito de condição de confiança se sobrepõe com o conceito de grau de confiança. Conforme Hardin (HARDIN, 2002) condição de confiança pode ser vista como o que o Trustor espera de um Trustee para ser fiel. Já grau de confiança, por sua vez, é um conjunto de condições de confiança para saber se o Trustee terá mais ou menos confiança em um determinado assunto.

Podemos citar novamente o exemplo de confiança do sensor de movimento, em que o usuário A confia em um sensor B para lhe fornecer sua localização, desde que o mesmo esteja parado e não em movimento, a movimentação do usuário é a condição de

confiança.

2.5.3 Construindo a confiança

A confiança é dada ao outro através do conhecimento prévio ou através de um histórico de interações. Dessa forma, é impossível estabelecer confiança quando nunca houver interação, pois, como sabemos, a confiança precisa ser construída. Entretanto, a literatura discute sobre três interações iniciais entre indivíduos: confiança generalizada, controle e reputação. A primeira é utilizada quando o risco de prejudicar-se é muito baixo; já o controle e a reputação são utilizados para diminuir o risco do Trustor (STARK, 2014).

Como citado acima, a confiança é construída aos poucos, então julgamos prudente aplicar a confiança generalizada para iniciarmos o processo de construção da confiança no Trustee, pois nela o nível de prejuízo do Trustor é baixo. Com isso, podemos saber através de pequenas doses de confiança se o Trustee é ou não confiável.

2.5.4 Gerenciamento de confiança

Conforme Josang et. al. (JøSANG; KESER; DIMITRAKOS, 2005), gerenciamento de confiança é a atividade de criação de sistemas e métodos que permite com que partes confiantes possam fazer avaliações e tomar decisões a respeito da confiabilidade das transações que envolvem alguns potenciais riscos, viabilizando igualmente que os usuários e proprietários do sistema possam aumentar e representar corretamente a confiabilidade de si mesmos e de seus sistemas.

Neisse (Neisse, 2012a) classifica o gerenciamento de confiança em 3 partes, senão vejamos:

1. Autorização e autenticação distribuída: valores de confiança associados a identidades e credenciais são computadores em um sistema distribuído e utilizam juntos um conjunto de regras para decidir se as ações solicitadas pelos assuntos devem ser autorizadas ou negadas;

2. Medições de reputação e confiabilidade: valores de confiança com foco em aspectos específicos são calculados com base em experiências direta ou indiretamente, através

de recomendações de terceiros. Esses valores de confiança são utilizados para apoiar a seleção de entidades, serviços ou produtos;

3. Atestando e checando a integridade, utilizando a prova de alteração do hardware: utiliza soluções técnicas que são, teoricamente, ou praticamente comprovadas seguras, tais como, Chips de computação confiável (*Trusted Platform Module* (TPM)) e soluções de cartões inteligentes.

2.6 MobileHealthNet

O projeto *MobileHelathNet* é fruto de uma parceria entre o Laboratório de Sistemas Distribuídos[1] (LSD) da Universidade Federal do Maranhao (UFMA) e o *Laboratory for Advanced Collaboration*[2] (LAC) da Pontificia Universidade Catolica do Rio de Janeiro (Puc-Rio). Conta-se igualmente com o apoio do Hospital Universitário da UFMA (HU-UFMA), responsável por fornecer o conhecimento da área de saúde necessário para o seu desenvolvimento. Existem duas principais unidades do HU-UFMA que contribuem com a iniciativa: o Programa de Assistencia a Pacientes Asmaticos (PAPA) e a Casa da Dor (BATISTA, 2013). O PAPA está condicionado ao tratamento e monitoramento de pacientes com este tipo de patologia crônica; já a Casa da Dor não faz distinção entre patologias, ficando responsável pelo tratamento de pacientes com qualquer tipo de dor aguda.

O projeto tem sua composição constituída por camadas, como podemos observar na figura 2.4. A primeira camada é a de comunicação, chamada de *MobileHealthNet Communication Framework*, a qual contém todos os mecanismos para facilitar o compartilhamento de dados nas RSMs. Baseia-se no *Scalable Data Distribution Layer* (SDDL) (DAVID et al., 2012a) que trabalha com dois protocolos de comunicação: o Mobile Reliable UDP (MR-UDP) (SILVA; ENDLER; RORIZ, 2013) e o *Data Distribution Service* (DDS) da *Object Manage Group* (OMG). O primeiro é responsável pela comunicação de entrada e saída entre a rede principal e os nós moveis; o DDS, pela comunicação dentro do *MobileHealthNet*. Por ser a camada na qual iremos focar nessa dissertação, separamos um tópico para explicá-la detalhadamente.

[1] http://www.lsd.ufma.br

[2] http://lac-rio.com/

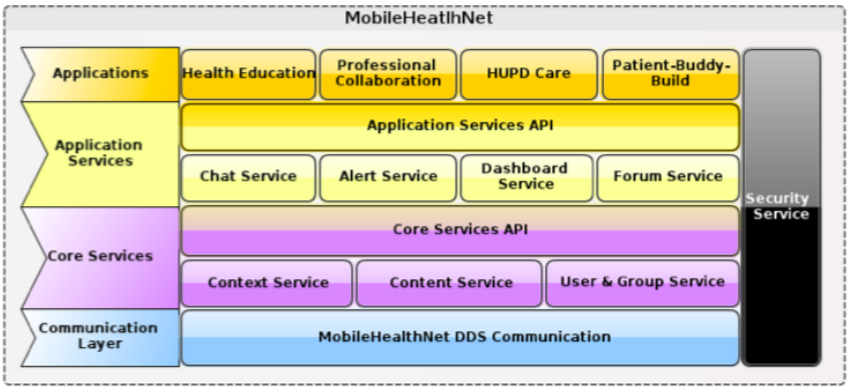

Figura 2.4: Arquitetura do MobileHealthNet

A camada seguinte é a *Core Services*, através da qual são disponibilizados os serviços básicos que as aplicações e os serviços criados a partir do MobileHealthNet utilizarão. A camada é composta por três serviços, a saber:

- **Context Service** - responsável por armazenar e disponibilizar as informações de contexto;
- **Content Service** - responsável pela publicação de mídias (imagens, vídeos, áudios, textos, etc) nas RSMs. Este serviço permite a nomeação de cada mídia com meta informações definidas pela aplicação (ex.: um modelo de sensor de ECG que um texto referencia);
- **User & Group Service** - serviço que gerencia os usuários e grupos das RSMs.

Já a camada *Application Service* disponibiliza serviços típicos de redes sociais, como o serviço de alertas, chat, fórum e até mesmo um serviço de publicação em murais. Em transversal com todas as camadas, está a camada *Security Services*, tratando de mecanismos de privacidade e segurança. Por fim, tem-se a camada *Applications*, que representa os possíveis tipos de aplicações a serem desenvolvidas.

2.6.1 Arquitetura da Infraestrutura do middleware MobileHealthNet Context Service (MHNCS)

A figura 2.5 sintetiza os aspectos estruturais do MHNCS.

O MHNCS constitui um conjunto de componentes necessários para auxiliar a integração do dispositivo com o ambiente pervasivo e disponibilizar os serviços necessários

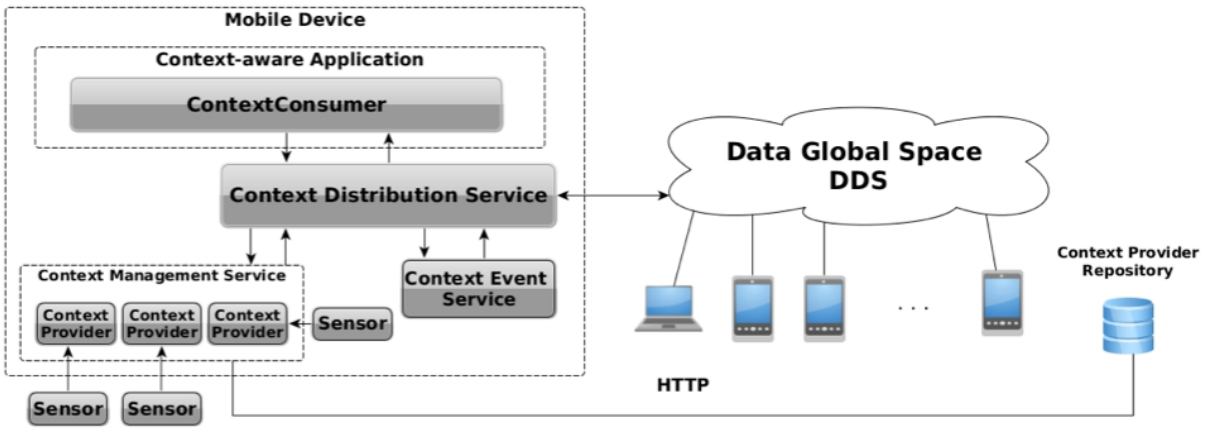

Figura 2.5: Arquitetura do MHNCS (PINHEIRO, 2014)

que integram a arquitetura. Conforme observamos na Figura 2.5, o MHNCS é formado por 3 componentes básicos: Módulo de Distribuição (*Context Distribuition Service* (CDS)), que é responsável por gerenciar a distribuição de contexto com os dispositivos móveis em um ambiente pervasivo; Gerenciador de Eventos (*Context Event Service* (CES)), que fornece mecanismo avaliadores das informacoes de contexto e notifica as aplicações somente quando as condições definidas forem satisfeitas; e parte da arquitetura proposta e formada por componentes modificados, oriundos de um middleware voltado para o gerenciamento de provedores de contexto denominado *Context Management Service* (CMS) (MALCHER et al.,).

3 *Confidere*

No capítulo 1.1, foi exposta a necessidade de um alto nível de confiança nos componentes e nas informações geradas por eles. O presente trabalho tem como objetivo avaliar o nível de confiança de um componente específico da infraestrutura, o Context Provider (CP). É ele o responsável por trazer todas as informações de contexto dos usuários envolvidos.

Como exemplo considera-se um cenário em que um paciente chamado João, possui uma doença crônica, a fibrilacão atrial [1], em que o mesmo necessita de atenção médica em tempo integral. Entretanto, João mora em uma região remota do estado do Maranhão, além disso João não pode se locomover, pois possui uma grave fratura na coluna. Seria muito custoso levar uma equipe de profissionais médicos para acompanhar um único paciente. Pensado nisso, a equipe do Hospital Universitário do Maranhão enviou um sensor capaz de aferir os batimentos cardíacos de João e enviá-los para o dispositivo móvel do mesmo, que por sua vez envia esses dados para uma equipe de profissionais da saúde envolvidos no caso.

Para iniciar a coleta dos batimentos cardíacos de João, o mesmo necessita solicitar um CP responsável para essa funcionalidade. Em outros tempos, como não havia o *confidere* na arquitetura do sistema utilizado por João, ele correria o risco de baixar um componente que utilizasse outros recursos do seu dispositivo móvel e além dessa utilização indevida por parte do componente, o mesmo poderia sobrecarregar alguns recursos essenciais para o bom funcionamento do processo de envio de informações sobre João aos profissionais da saúde. Reforça-se a ideia de que o comportamento indevido desse componente implica diretamente as informações enviadas ao médico e profissionais da saúde envolvidos. Sendo assim, essas informações podem ocasionar uma tomada de decisão equivocada em relação ao paciente em questão, atingindo, assim, o seu estado de saúde, o que poderia levar até ao óbito.

Com o *confidere* implantado na arquitetura, no momento em que João solicita um CP para aferir seus batimentos cardíacos, o *confidere* se conecta à sua base de dados

[1] http://dx.doi.org/10.1590/S0066-782X2009001500007

de confiança, seleciona o CP que melhor se adéqua às exigências da aplicação utilizada por João e realiza o download do mesmo. João não precisa mais se preocupar com o que o CP irá utilizar em seu dispositivo, pois as permissões que o CP deverá ter sobre o dispositivo móvel de João, já foram previamente configuradas pelo desenvolvedor da aplicação cliente.

O *confidere* irá avaliar o nível de confiança de cada CP que será baixado e instanciado dinamicamente pela aplicação cliente. Nesse contexto, o Trustor será o *confidere*, ou seja, quem irá avaliar o nível de confiança, e o Trustee, o CP, isto é, quem será avaliado.

O presente trabalho propõe uma abordagem em que a avaliação da confiança é realizada não somente pelas interações diretas (interações anteriores do trustor com o trustee) e indiretas (interações anteriores de terceiros com o trustee) do CP, mas também pelo comportamento do mesmo em tempo de execução, ou seja, quais e como ele utiliza os recursos computacionais do dispositivo móvel, enquanto o mesmo coleta os dados do usuário e os fornece para os interessados, que, em nosso contexto, são representados pelos profissionais da saúde envolvidos.

Um dos principais problemas destacado neste trabalho é a inexistência de confiança nos CPs da arquitetura do projeto *MobileHealthNet*. Eles são baixados de repositórios localizados em diferentes servidores e instanciados dinamicamente.

Para explicar melhor a proposta, optou-se por uma divisão em duas partes: a) a representação matemática da avaliação de confiança, ou seja, como será calculada o nível de confiança do CP em Sistemas Sensíveis ao Contexto; e b) a aplicação desenvolvida a partir dessa representação.

A aplicação *confidere* foi desenvolvida no sentido de apoiar no gerenciamento dos componentes da camada de contexto do projeto *MobileHealthNet*, em específico os CPs. Ela é responsável por garantir a confiança nesses componentes, ou seja, garantir a qualidade do funcionamento do CP após a instalação nos dispositivos móveis. A figura 3.1 contempla o local em que a aplicação em comento ficará.

A aplicação servirá de apoio para o componente *Context Management Service* (CMS), pois ela ficará responsável por especificar e monitorar o CP que mais se adequar ao usuário.

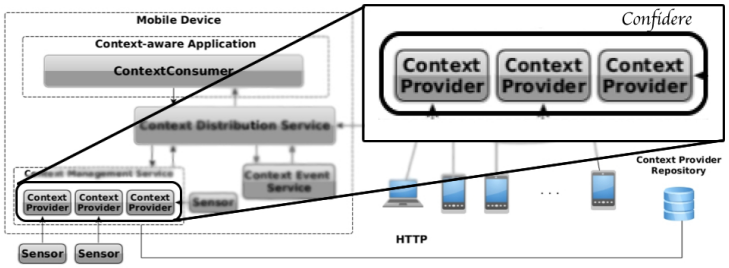

Figura 3.1: Local do *Confidere* no MHNCS

3.1 Representação matemática

O modelo de confiança definido neste trabalho baseia-se em um conceito de Russell (HARDIN, 2002), pelo qual a confiança é dada através do comportamento que o Trustee terá em relação ao Trustor. Entretanto, esse comportamento só será tido como confiável se o Trustee se comportar da maneira esperada pelo Trustor. Outro fator determinante para o modelo é o conceito de reputação, no qual a confiança é obtida através da opinião de terceiros.

O primeiro tipo de confiança é obtido pelas interações diretas, ou seja, as transações que o Trustor teve com o Trustee. A cada nova interação com o Trustee, o Trustor terá um nível diferente de confiança. Já a reputação é obtida através das interações indiretas, isto é, das interações diretas que outras entidades tiveram com o Trustee.

Na equação 3.1, D representa as interações diretas; a e b representam a aplicação cliente (o Trustor) e o provedor de contexto (o Trustee), respectivamente; c representa o contexto em que b será avaliado. Esse contexto, por sua vez, é composto por propriedades, que são representadas pela quantidade máxima de uso do processador, memória RAM e Banda que a acha que b deverá ter para ser considerado confiável; i representa o número da interação. Se a considerar que b no contexto c é confiável, é adicionado um ponto ao valor atual de confiança de b. Caso contrário, retira-se um ponto do valor atual da confiança de b.

$$D_{ab}^{c}(i) = \begin{cases} +1\ , & if\ a\ trust\ in\ b \\ -1\ , & otherwise \end{cases} \tag{3.1}$$

O valor total das interações diretas é dado pela equação 3.2, na qual, T_{ab}^{c} representa o somatório de interações diretas que houve entre a e b no contexto c.

$$T_{ab}^{c} = \sum_{i=1}^{n} D_{ab}^{c}(i) \tag{3.2}$$

Já o valor da interação indireta, ou seja, da reputação de b é obtido através da equação 3.3,

$$R_{b}^{c} = \frac{\sum_{i=1}^{k} T_{ib}^{c}}{k} \tag{3.3}$$

na qual, R_{b}^{c} representa a média entre todas as interações realizadas por b no contexto c; e k representa a quantidade de usuários que interagiu com b no contexto c.

A confiança total é dada na equação 3.4, que se refere à soma das interações diretas e indiretas,

$$C_{ab}^{c} = T_{ab}^{c} \ + \ R_{b}^{c} \tag{3.4}$$

em que, C_{ab}^{c} corresponde ao valor da confiança final que a terá em b no contexto c. Como podemos observar na equação 3.4, a confiança é dada ao outro através do conhecimento prévio ou através de um histórico de interações. Dessa forma, é impossível estabelecer confiança quando nunca houve interação, pois, como sabemos, a confiança precisa ser construída.

3.2 Aplicação *Confidere*

O *Confidere* tem a função de avaliar o nível de confiança dos CPs, ou seja, detectar se o CP irá se comportar da forma esperada pela aplicação cliente. A arquitetura da aplicação é contemplada na figura 3.2.

A figura 3.2 especifica todos os componentes da aplicação *confidere*, que são: i) Trust_db, local em que ficam armazenados os níveis de confiança de cada CP; ii) Best Select Context Provider (BSCP), responsável por selecionar o provedor que melhor se adapta às exigências da aplicação cliente; iii) Monitor Context Provider (MCP), responsável por monitorar o CP durante a coleta de dados; e iv) Evaluation Context Provider (ECP), que possui a funcionalidade de avaliar o CP, ou seja, é o componente que dará a nota ao CP.

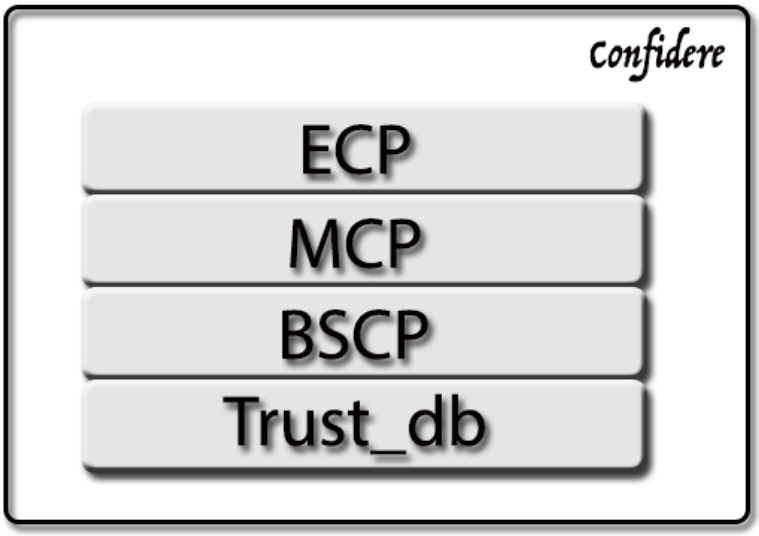

Figura 3.2: Arquitetura do *Confidere*

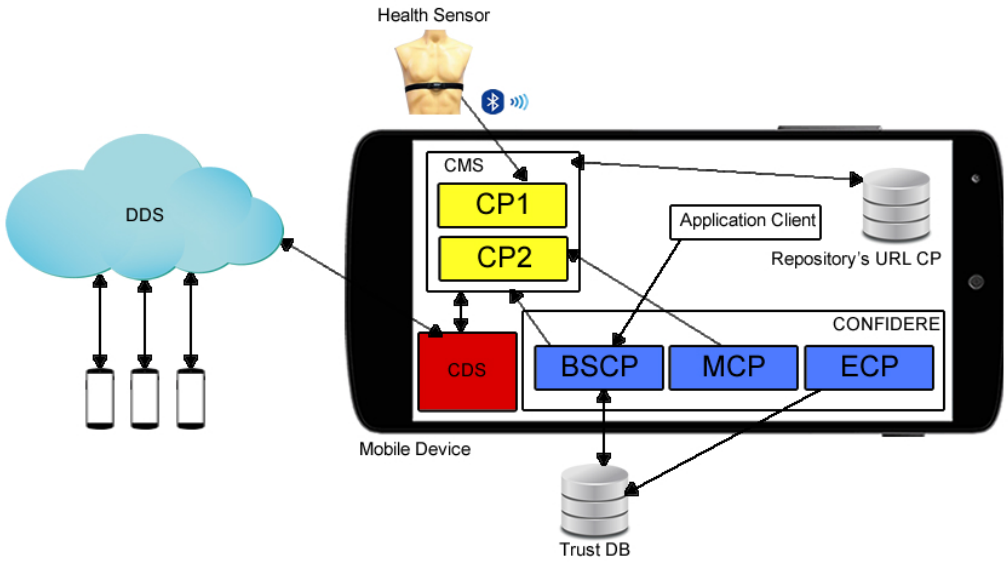

Figura 3.3: Arquitetura do *Confidere*

Um outro componente integrante dessa arquitetura é o CMS, elemento responsável por baixar e gerenciar todos os CPs. Juntamente a ele, temos também um repositório local, responsável por armazenar os endereços dos CPs. A figura 3.3 contempla a arquitetura do projeto em sua inteireza.

Para facilitar o entendimento do uso aplicação, foi desenvolvido um diagrama de sequência. A figura 3.4 demonstra toda a ação da aplicação, desde a solicitação do CP por parte da aplicação cliente até o seu posterior descarte.

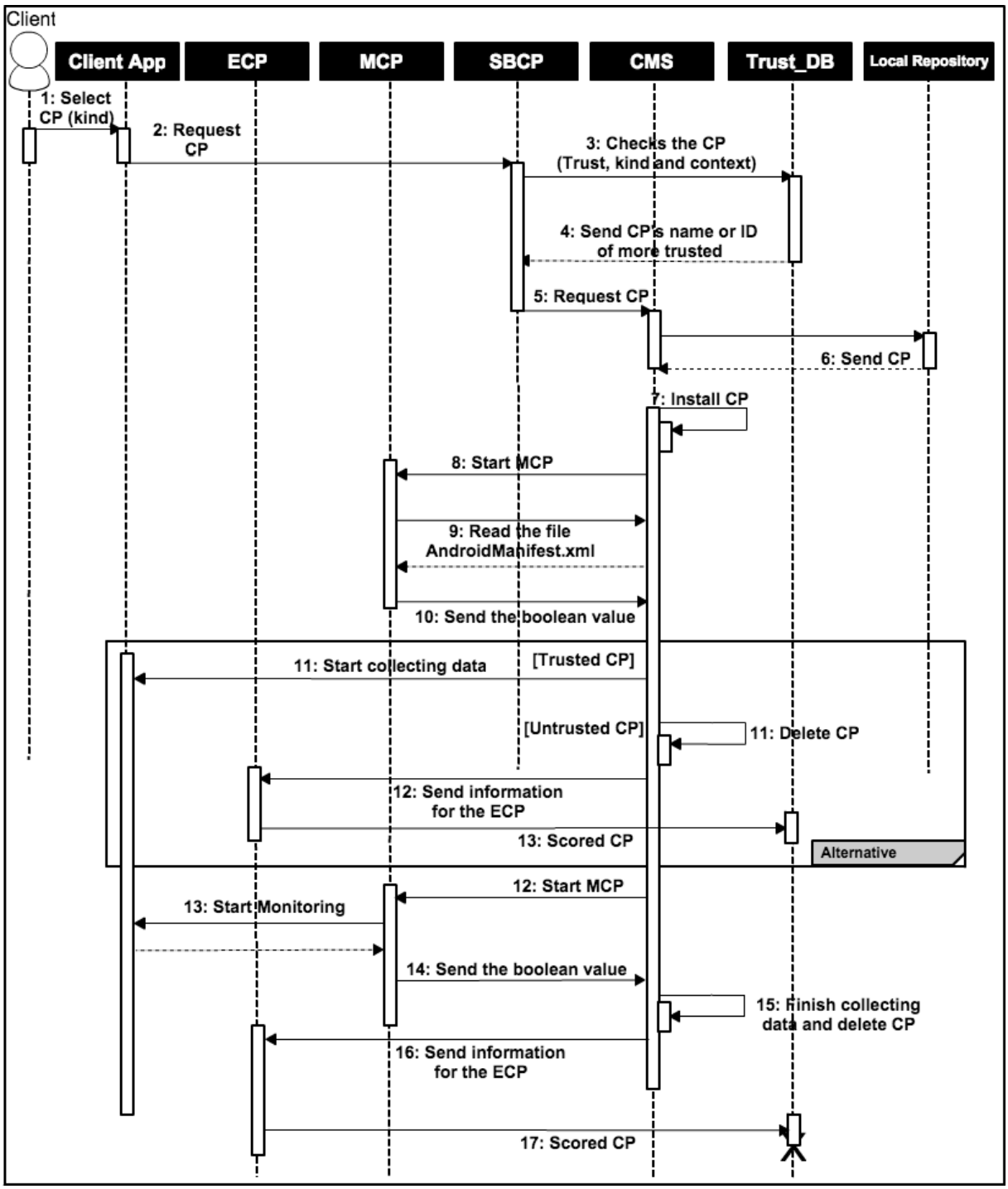

Figura 3.4: Diagrama de Sequência do *Confidere*

O processo do *confidere* funciona da seguinte forma: 1: O cliente solicita para a aplicação um CP esepecífico para sua necessidade; 2: A aplicação cliente solicita ao componente *SBCP* o tipo de CP com o nível de confiança que ele deverá ter e em qual tipo e contexto ele deverá estar inserido. O contexto é representado pelo conjunto de propriedades que o CP terá, ou seja, quais serão suas permissões (contidas no arquivo *Androidmanifest.xml*) e o quanto de CPU, RAM e Banda o CP consome. Suponhamos

que para *a* considerar **b** confiável, **b** deverá usar só 5% da CPU, 10% da memória RAM e 5 MB de banda. 3: O *SBCP* realiza uma consulta na *Trust_DB*; 4: A *Trust_DB* retorna o id do provedor que melhor se adapta às exigências da aplicação requisitante; 5: O *SBCP* envia ao CMS o nome do CP; 6: O CMS efetua uma consulta no repositório local pelo id do CP; 7: O CMS baixa e o instala. 8: O CMS ativa o componente MCP para a verificação das permissões do CP; 9: O MCP realiza uma leitura nas permissões do CP contidas no arquivo AndroidManifest.xml do mesmo; 10: O MCP retorna um valor booleano para o CMS, indicando se ele pode ou não instanciar o CP. Inicia-se a condição:

- **Caso o valor seja falso:** 11: O CMS desinstala o CP; 12: O CMS envia o valor booleano para o ECP; 13: O EVC, por sua vez, conecta-se a *Trust_DB* e incrementa um ponto no nível de confiança do CP.
- **Caso o valor seja verdadeiro:** 11: O CMS instancia o CP, ou seja, autoriza o inicio da coleta de dados; 12: O CMS ativa o componente MCP para iniciar o monitoramento do CP; 13: O MCP inicia o monitoramento; 14: Após o monitoramento, o MCP envia o dado booleano para CMS. Esse valor só será falso, caso o CP se comporte de maneira indevida durante esse monitoramento. Sendo assim, volta-se ao passo 11 do CP não confiável; 15: O CMS finaliza a coleta de dados e desinstala o CP; 16: O CMS envia o valor booleano para o ECP; 14: O EVC, por sua vez, conecta-se a *Trust_DB* e incrementa um ponto no nível de confiança do CP.

Nas subseções 3.2.1, 3.2.2, 3.2.3 e 3.2.4 os componentes Trust_db, Best Select Context Provider, Monitor Context Provider e Evaluation Context Provider serão abordados de forma mais detalhada.

3.2.1 Trust_db

A Trust_db é o componente em que se armazenam todas as informações sobre confiança relativas a todos os provedores de contexto existentes. A base é um banco de dados Mysql, cuja estrutura é demonstrada na figura a seguir 3.5.

A tabela principal do modelo é a *context_provider*,que representa as informações referentes ao CP. Ela é constituída pelo id, nome, url e o id do repositório

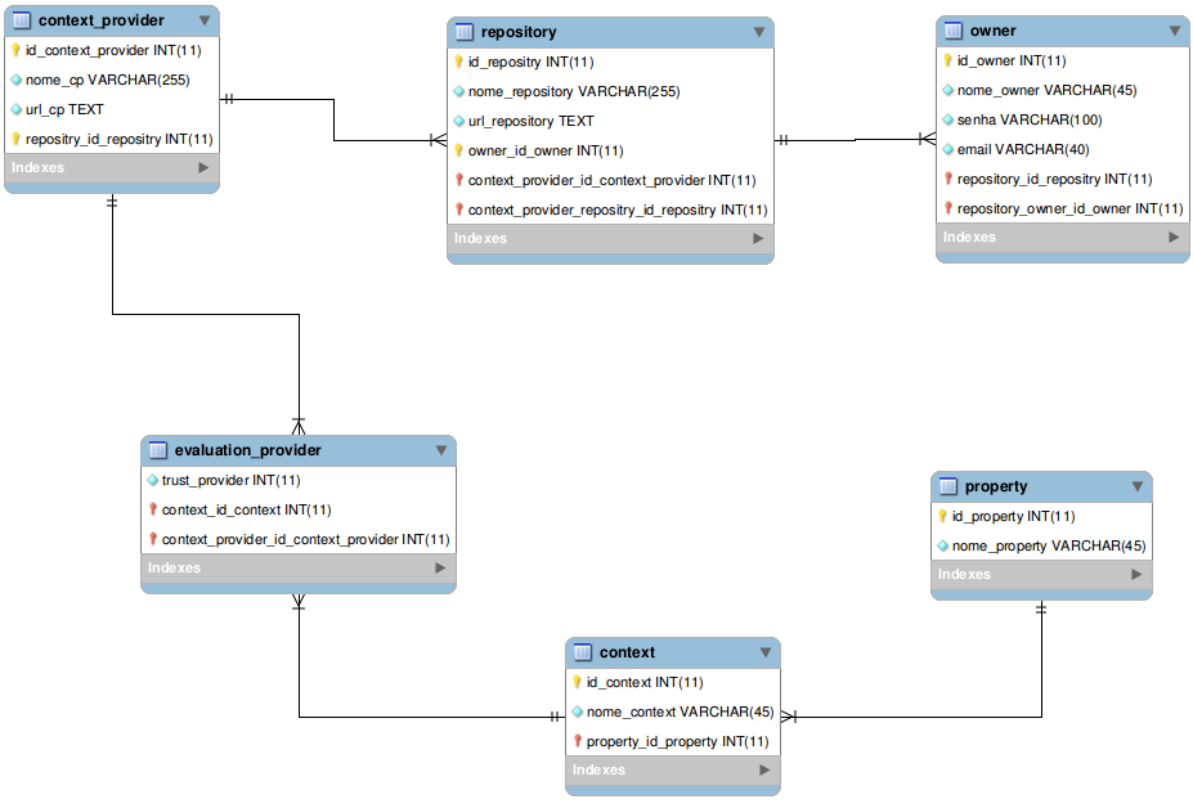

Figura 3.5: Esquema da base de dados de confiança (Trust_db)

do qual ele faz parte. Apesar de a tabela possuir um campo de endereço, o mesmo não é utilizado pela solução, pois o endereço real do CP é retirado da lista local de possíveis repositórios que se encontra no dispositivo móvel.

Outra tabela pertencente à base é a *repository* (repositório), que é responsável por armazenar o nome dos repositórios existentes. Ela foi desenvolvida para armazenar um conjunto de provedores de contexto em relação a seus proprietários, ou seja, organizar os CPs de acordo com os seus desenvolvedores. Ela também foi idealizada para suprir necessidades futuras, a exemplo da possibilidade de garantir a confiança nos repositórios, já que é possível que neles estejam um conjunto de CPs maliciosos.

Já a tabela Owner (Proprietário) tem a funcionalidade de armazenar o nome do proprietário do repositório e dos CPs, pois um desenvolvedor pode possuir vários repositórios e consequentemente vários CPs. Com ela, é possível saber quantos e quais repositórios/CPs o desenvolvedor possui. Como explicado anteriormente, ela também foi idealizada para garantir a confiança nos desenvolvedores dos CPs. Nota-se que a arquitetura proposta neste trabalho tem a possibilidade de garantir a confiança em 3 níveis, que são: a confiança no CP, no repositório e no desenvolvedor. Entretanto, o

trabalho foca na análise da confiança somente no CP.

A tabela *property* (propriedade) tem o papel de armazenar as informações das propriedades, que são representadas pelas permissões aos recursos computacionais do dispositivo móvel. Além das permissões, outro fator também é considerado propriedade na tabela, é a quantidade de uso em porcentagem do processador, memória RAM e banda consumida.

Uma das principais tabelas na base dados de confiança é a *context* (contexto), pois é responsável por armazenar as informações de contexto dos usuários. Essas informações são representadas por um conjunto de propriedades. Em exemplo, temos o contexto 1, que é constituído pelas seguintes permissões: *ACCESS_FINE_LOCATION*; *ACCESS_MOCK_LOCATION* e *INTERNET*. As duas primeiras permissões representam o acesso ao GPS do dispositivo local. A terceira permissão, por sua vez, representa o poder de acesso à internet. Outra informação presente nesse contexto é a porcentagem da utilização do processador, memória RAM e banda, que são 20%, 10% e 20 MB, respectivamente.

Finalmente, a tabela *evaluation_provider* (avaliação do provedor). Essa tabela possui um campo denominado *trust_provider*, que é o cerne do trabalho aqui proposto. Esse campo tem como funcionalidade armazenar o nível de confiança do CP referentes ao contexto em que o provedor está inserido no momento da avaliação. Outra particularidade dessa tabela é que é aqui que são empregados os conceitos da representação matemática citados acima. Então, localiza-se aqui o somatório das interação ocorridas, ou seja, a soma das interações diretas e indiretas em seus respectivos contextos.

3.2.2 Best Select Context Provider (BSCP)

Como relatado anteriormente, o BSCP é o componente responsável por selecionar o CP mais adequado à aplicação cliente. É ele que vai se conectar à base de dados de confiança, isto é, na Trust_db. Após sua conexão com o banco, o componente realiza uma busca por tipo e por contexto. Essa busca é realizada através de método específico da classe SelectBestContextProvider, denominado buscarProvedor que tem como retorno o nomeProvedor.

A figura 3.6 representa o diagrama de classe em que é contemplado os

atributos e métodos da classe SelectBestContextProvider. Observa-se que a classe do componente em questão possui os métodos: i) getConnector(): método responsável por chamar a conexão existente no sistema; ii)setConnector(): tem como função informar os parâmetros do banco para conexão; iii)updatedirectTrust(): função para atualizar a base de confiança com as experiências do mesmo em relação ao Trustee, caso existentes; e iv)buscarProvedor(): método responsável pela busca do CP que melhor se adapta ao contexto do usuário.

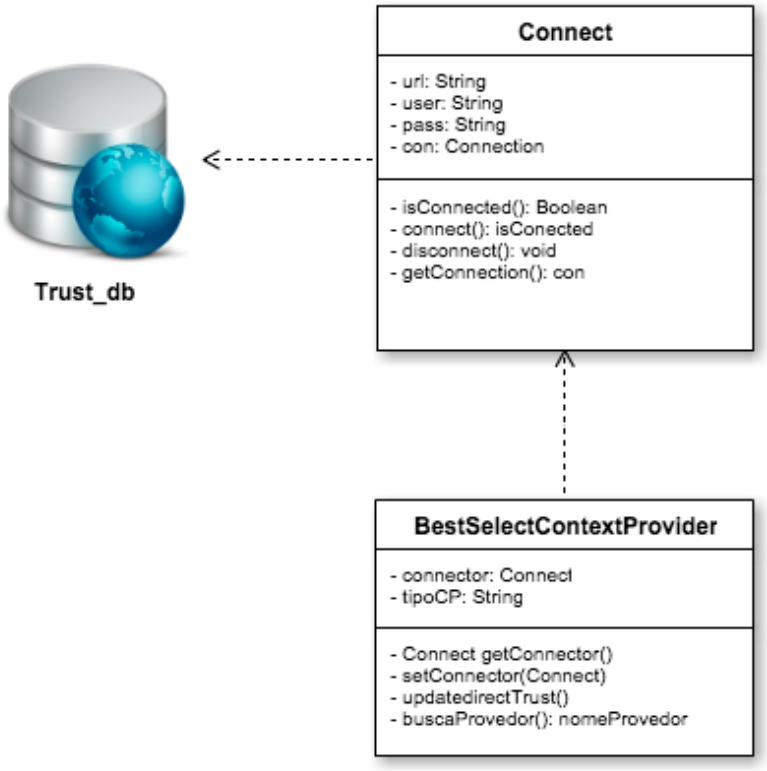

Figura 3.6: Diagrama de classe do SBCP

Outra classe presente na figura 3.6 é a Connect. As tabelas 3.1 e 3.2 descrevem seus atributos e métodos respectivamente.

A figura 3.7 detalha o comportamento do componente BSCP. Após a solicitação do usuário, o componente é iniciado. O mesmo conecta-se a base de confiança selecionado pelo desenvolvedor, ou seja, nossa arquitetura possibilita a existência de várias bases de confiança (Trust_db). Em seguida, é realizada uma consulta que utiliza como parâmetro o tipo de CP, isto é, se ele é de localização, tempo, temperatura, etc, juntamente com a busca pelo seu contexto. Esse contexto é composto por uma ou várias propriedades, que, por sua vez, são representadas pela permissões (contidas em um arquivo específico

Tabela 3.1: Descrição dos atributos da Classe Connect

Atributo	**Descrição**
url	Armazena o endereço da base de dados de confiança.
user	Guarda o login do usuário da base de dados.
pass	Armazena a senha do usuário que realiza a conexão com a base.
con	Objeto da classe Connection, responsável por realizar a conexão com a base de dados.

Tabela 3.2: Descrição dos métodos da Classe Connect

Método	**Descrição**	**Tipo**
isConnected	Verifica se existe conexão com o banco.	Público
connect	Estabelece a conexão com a base de dados e retorna se a mesma foi bem sucedida.	Público
disconnect	Finaliza a conexão com a base de dados.	Público
getConnection	Chama a conexão existente.	Público

do CP) e pela quantidade de uso dos recursos computacionais, em porcentagem, que são: CPU, memória RAM e banda utilizada.

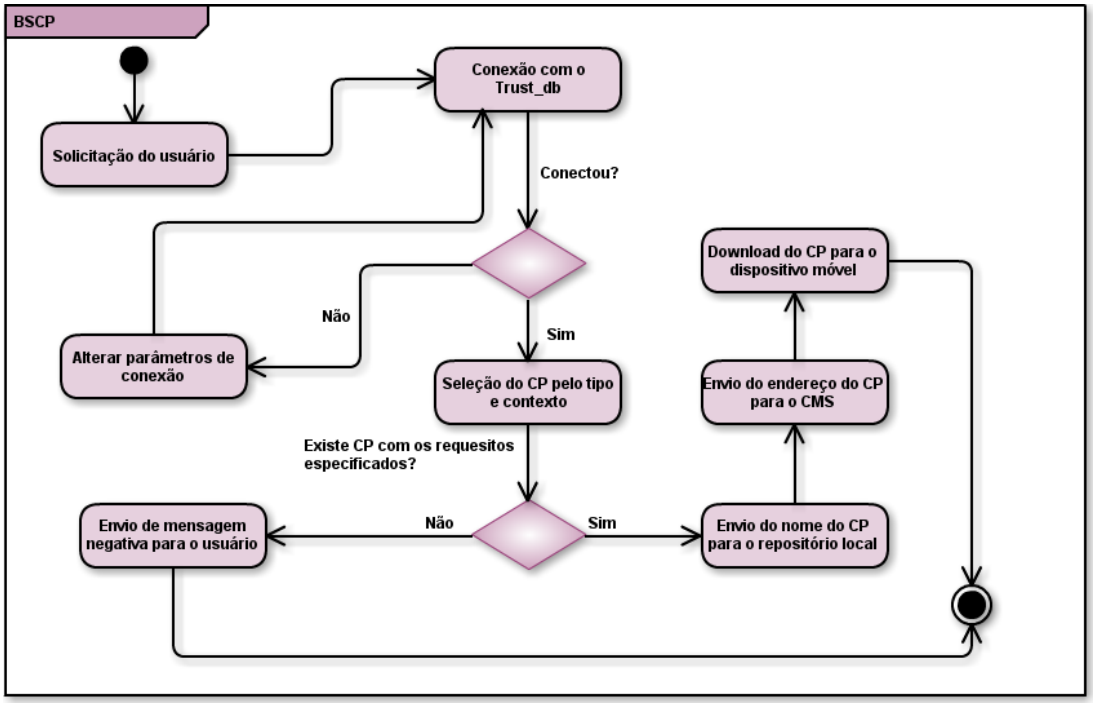

Figura 3.7: Diagrama de atividade do BSCP

Caso exista um CP que se adapte a todos os requisitos exigidos pelo

desenvolvedor, é retornado um nome do CP e inicia-se uma nova busca pelo seu endereço, só que agora a pesquisa é realizada em um repositório local. Após essa consulta, o endereço do CP é enviado ao CMS , que inicia o processo de Download do CP.

3.2.3 Monitor Context Provider (MCP)

Com o intuito de monitorar o comportamento do CP em tempo de execução, a fim de garantir a segurança do usuário e estabilidade do CP, o *confidere* fornece um componente chamado *Monitor Context Provider (MCP)*. Esse componente é responsável por monitorar o uso dos recursos computacionais utilizados pelo CP. Com esse componente ativo, é possível acompanhar o uso de três principais recursos do dispositivo móvel, que são: processador, memória RAM e a quantidade de banda utilizada pelo CP.

O limite de uso desses recursos é previamente estabelecido pelo desenvolvedor da aplicação cliente, ou seja, é responsabilidade do desenvolvedor expressar, em porcentagem, o uso do processador, memória RAM e a quantidade de banda que o CP deverá utilizar.

Outra responsabilidade do desenvolvedor é especificar a quais recursos, além dos citados acima, o CP terá acesso dentro do dispositivo.

A plataforma android possui uma medida de segurança para a utilização dos recursos do dispositivo móvel, com exceção dos recursos citados anteriormente. Para que uma aplicação tenha acesso a algum recurso do dispositivo, é necessário que o desenvolvedor especifique quais recursos a sua aplicação deseja utilizar.

Caso a aplicação tente acessar algum recursos que a mesma não possui permissão, a plataforma realiza uma preempção no seu funcionamento, ou seja, sua execução é cancelada.

No *MobileHealthNet* todos os CPs são aplicações android. Entretanto, as permissões atribuídas a cada um deles são as mesmas do CMS. Portanto, antes era possível criar um CP com acesso a qualquer recurso computacional do dispositivo móvel sem a necessidade da autorização do usuário. Isso deixava o usuário vulnerável a ataques, como por exemplo, o envio de informações confidenciais para terceiros.

Com a implementação do MCP ao *MobileHealthNet* essa lacuna foi preenchida, pois o mesmo verifica qual deverá ser o comportamento do CP especificado pelo

desenvolvedor, ou seja, é verificado antes da coleta de informações quais permissões o CP possui.

Caso o mesmo não possua as mesmas permissões estabelecidas pelo desenvolvedor, é criado um evento para solicitar ao CMS que o mesmo realize o cancelamento desse CP.

Na figura 3.8, é possível perceber o comportamento do componente em comento. Nota-se que após a instalação do CP, inicia-se o MCP, que, por sua vez, ativa um de seus métodos responsáveis pela leitura de um arquivo específico do CP, chamado *AndroidManifest.xml*. O trecho de código 3.1 contempla o conteúdo desse arquivo de um CP de tempo.

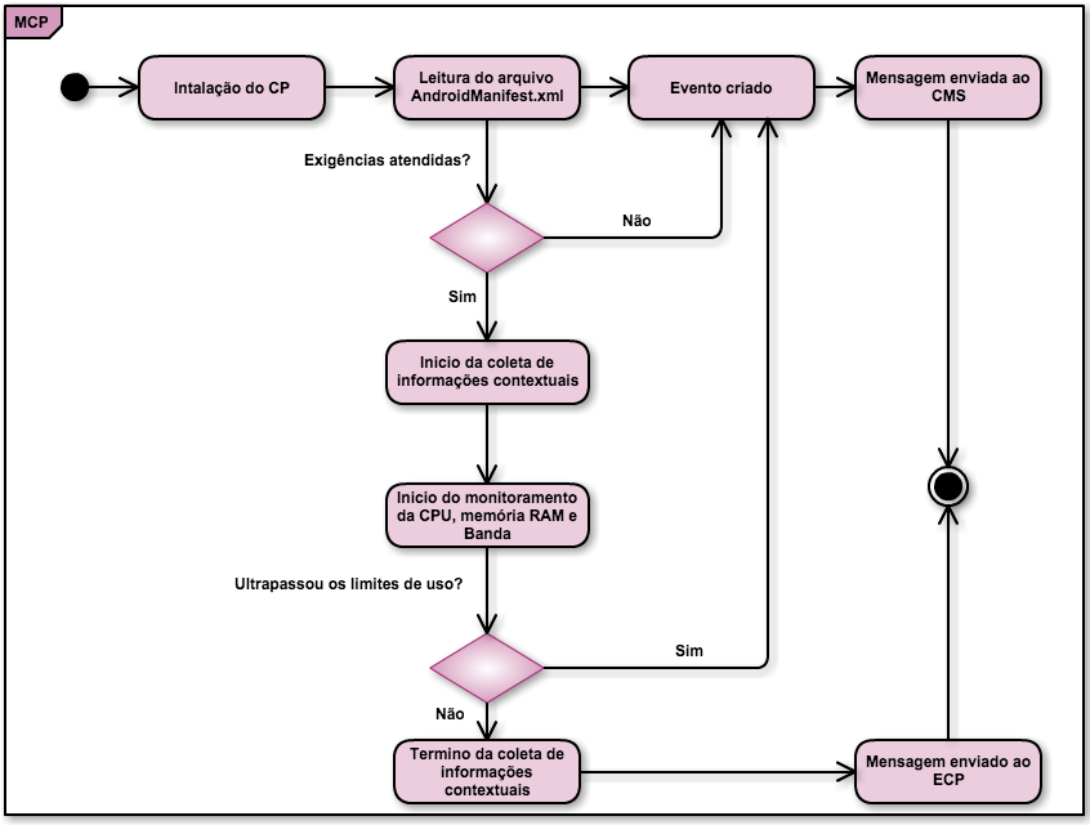

Figura 3.8: Diagrama de atividade do MCP

É nesse arquivo que o MCP verifica e compara se as necessidades do desenvolvedor são atendidas ou se o CP tem mais acesso do que deveria. É possível observar que no arquivo xml 3.1, especificamente nas linhas 14 e 15, ele possui uma tag denominada $< uses - permission >$ para especificar quais são as permissão que aquele CP terá acesso no dispositivo móvel, com exceção ao processador e memória RAM, pois todas as aplicações, por padrão, possuem permissões para acessar esses dois recursos. No

caso do exemplo do CP de tempo mostrado no arquivo xml, o mesmo só possui permissão para acessar e configurar o bluetooth do dispositivo móvel.

Listing 3.1: Conteúdo do AndroidManifest.xml de um CP de tempo

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3          package="br.rio.puc.inf.lac.mobilis.cms.provider"
4          android:versionCode="1"
5          android:versionName="1.0">
6
7          <application android:icon="@drawable/icon"
8            android:label="@string/app_name">
9
10
11       </application>
12       <uses-sdk android:minSdkVersion="3" />
13
14   <uses-permission android:name="android.permission.BLUETOOTH" />
15   <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
16
17 </manifest>
```

Caso essas tags não estejam de acordo com a necessidade do desenvolvedor ou ultrapassem as exigências do mesmo, a aplicação não chega a ser iniciada. Entretanto, se elas atenderem exatamente as necessidades do desenvolvedor, a coleta das informações é iniciada.

Em relação à implementação de mecanismo de monitoramento do CP, a figura 3.9 expõe o diagrama de classe, em que é possível observar que a interface *IMonitorContextProvider* é responsável por fornecer os métodos que a classe *MonitorContextProvider* necessitará para analisar o comportamento do CP em tempo de execução, ou seja, no momento em que o mesmo estará coletando as informações.

Na tabela 3.3 são detalhados todos os atributos da classe *MonitorContextProvider*. É importante ressaltar que o corpo de todos os métodos herdados da interface *IMonitorContextProvider* são preenchidos por funções já existentes dentro do *framework* Android. Em exemplo temos o getUsedBand(), que utiliza a classe *TrafficStats* para analisar os a quantidade de dados enviados e recebidos pela rede móvel.

Tabela 3.3: Descrição dos atributos da Classe *MonitorContextProvider*

Atributo	**Descrição**
receveid	Responsável por armazenar os dados recebidos pela rede móvel.
send	Guarda a quantidade de dados enviados pelo CP.
total	Quantidade total dos dados enviados e recebidos pela rede móvel.
cpu1	Armazena a quantidade total de uso da CPU pelo dispositivo móvel.
cpu2	Armazena a quantidade de uso da CPU pelo CP.
freeSize	Armazena a quantidade de memória RAM livre do dispositivo móvel.
totalSize	Guarda o total de memória RAM do dispositivo .
usedSize	Armazena a quantidade de memória utilizada pelo CP.

Tabela 3.4: Descrição dos métodos da Classe *MonitorContextProvider*

Método	**Descrição**
getUsedCPU	Retorna o valor da CPU utilizado pelo CP.
getFreeMemorySize	Utilizado para retornar o valor livre memória RAM.
getUsedMemorySize	Retorna o valor da memória RAM, utilizado pelo CP.
porcentual	Transforma os valor de uso dos recursos CPU e memória RAM, em porcentagem.
getUsedBand	Retorna o valor total de dados utilizado pelo CP.

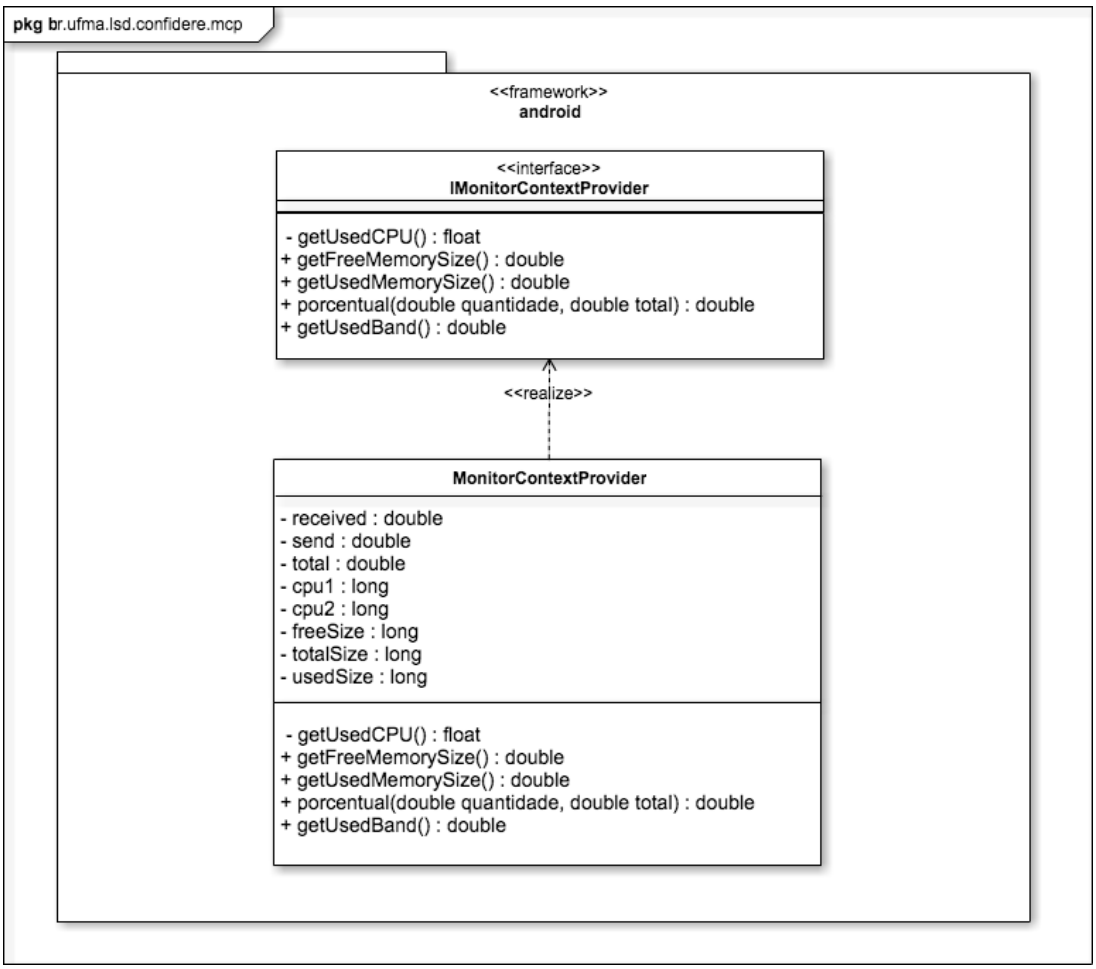

Figura 3.9: Diagrama de Classe do MCP

Em seguida, a tabela 3.4 descreve de forma detalhada os métodos dessa classe. Os principais métodos da classe MCP, são os: i) getUsedCPU(), em que ele utiliza os atributos cpu1 e cpu2 para calcular a quantidade de CPU utilizada pleo CP; ii) getUsedMemorySize(), em que o mesmo utiliza os atributos freeSize, totalSize e usedSize para calcular o valor total de memória RAM utilizada pelo CP; iii) getUsedBand(), que utiliza os métodos receveid, send e total para retornar a aplicação a quantidade de dados trafegados pela rede móvel, através das ações do CP em análise.

3.2.4 Evaluation Context Provider (ECP)

Após o processo de seleção do CP que melhor se adapta ao contexto do usuário e o monitoramento do mesmo durante a coleta das informações de contexto, torna-se necessária a exposição da experiência que o usuário teve com o CP. Para essa finalidade, o *confidere* fornece um componente denominado *Evaluation Context Provider* (ECP). A

figura 3.10 expõe o comportamento do componente em comento, desde a criação do evento que informa a finalização do monitoramento do CP, até a finalização da comunicação com a Trust_db.

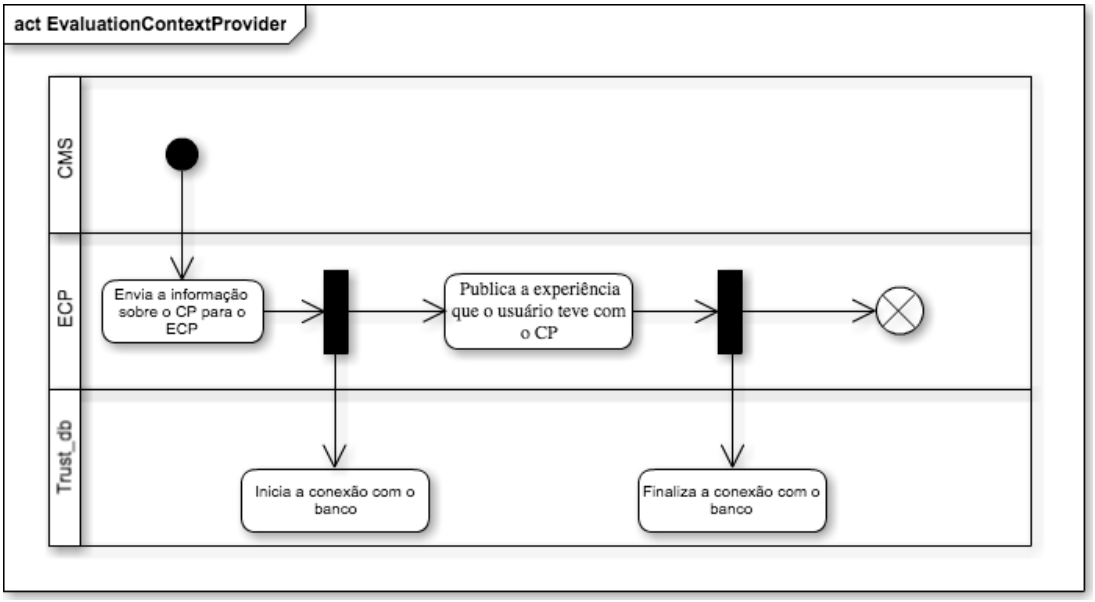

Figura 3.10: Diagrama de atividade do ECP

Com o processo de monitoramento finalizado, o CMS adquire a informação de confiança sobre o CP avaliado. Essa informação é enviada ao componente ECP, que, por sua vez, estabelece uma conexão com a base de dados de confiança. Após a conexão estabelecida, o ECP decrementa ou incrementa um ponto no nível de confiança de CP. Essa inserção ou retirada do ponto de confiança no CP está diretamente ligada ao comportamento que o mesmo teve durante a interação com o usuário. Caso o CP se comportou de maneira correta, ou seja, de acordo com o esperado pela aplicação cliente, o mesmo terá um um ponto a mais no seu nível de confiança, caso contrário terá um ponto a menos desse mesmo nível. Em seguida, o ECP finaliza a conexão com a Trust_db.

4 Trabalhos Relacionados

4.1 TValue

Neisse (Neisse, 2012b) propõe um modelo inovador de gerenciamento de confiança denominado *TValue*, instanciado em um conjunto de opiniões de Lógica Subjetiva (SL) (JØSANG, 2001), que suporta incertezas e fornece operadores para lidar com cálculos de opiniões de confiança, tais como consensos e discordâncias. De acordo com a teoria da SL, a confiança, em uma determinada proposição, é representada por uma tripla $(b, d, u) \in [0, 1]^3$, em que *(b)* representa crença;*(d)*, descrença; e *(u)*, incerteza.

$$A \xrightarrow[v]{*;a} B$$

em que, A representa o Trustor, ou seja, o elemento que dará a confiança, e B, o Trustee, o elemento que a receberá. Já o *"*"* corresponde às classes de relacionamentos de confiança, que são: i) *direct functional (df)* e ii) *indirect functional (if)*. Então, $* \in \{df, if\}$. E $a \in \{idp, pe, cip\}$, em que *idp* é *identity provisioning*; *pe, privacy enforcement*; e *cip, context information provisioning.*

Outra importante característica na solução do autor é a questão do conjunto de papéis da aplicação $R = \{CO, US, CP, IP, SP\}$, *Context Owner, User, Context Provider, Identity Provider e Service Provider* nessa ordem. Eles indicam qual papel a entidade está representando, já que podem exercer vários deles.

Um outro fator importante no cálculo da confiança é que o autor sempre assume $role(A) = US$, ou seja, o Trustor sempre vai ter o papel de usuário. Dada a seguinte equação:

$$\frac{[trust\_PE(A,B) = v]}{A \xrightarrow[v]{df;pe} B} \; role(B) \in \{CP, IP, SP\}$$

Em que, o autor avalia o grau v de confiança que A tem em B com o aspecto *pe (privacy enforcement)*. Esse resultado só é significante quando B representa os papeis de *Context Provider, Identity Provider e Service Provider.* O grau de confiança resulta em um

dos valores do conjunto ordenado $\{VT, T, U, VU\}$, denominados como *very untrustworthy (VU), untrustworthy (U), trustworthy (T), and very trustworthy (VT)*, em que, o autor assume $VT > T > U > VU$.

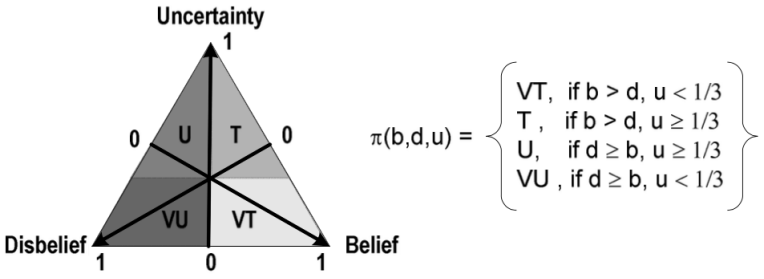

Figura 4.1: Função $\pi : [0, 1]^3 \rightarrow \{VT, T, U, VU\}$

Na Figura 4.1, o autor define as condições para cada valor do conjunto ordenado, na qual, a informação só é *very trustworthy* quando a descrença e a incerteza forem menores que a crença e 1/3 das opiniões. Esses valores podem variar de acordo com o Trustor, com seu foco (se é privacidade, na informação do contexto ou no provimento da identidade),Trustee, (se a confiança é direta ou indireta), ou seja, se mudar o contexto da avaliação, mudará o valor da confiança.

Um dos pontos fracos da proposta é a necessidade do *feedback* do usuário, pois a avaliação é feita para sistemas sensíveis ao contexto, ou seja, seria importante a transparência dessa avaliação para o mesmo.

4.2 MAETROID

Em Dini et. al. (DINI et al., 2013), os autores propõem um *framework* para avaliar o nível de confiança de uma aplicação Android, denominada de avaliação de confiança de aplicações móveis para Android (Mobile Application Evaluator of TRust for andrOID - MAETROID).

Nesse trabalho, foi avaliada a confiança através do arquivo manifest do android, pois é nesse arquivo que se encontram todas as permissões da aplicação, tais como: permissão para acessar o gps, os contatos, id ou até mesmo acessar a internet.

Os aplicativos são divididos em três categorias: *i) Bom App*, quando o aplicativo se comporta de maneira esperada, tanto na segurança, quanto sob ponto

de vista funcional, portanto ele é considerado confiável; *ii) App Infectado*, sempre que um aplicativo está infectado por um *malware*, neste caso, o aplicativo é considerado não confiável; *iii) Mau App*, um aplicativo que não funciona corretamente (como por exemplo, aplicações que deixam de funcionar muitas vezes) ou inutilizável, essa categoria é denominada enganosa.

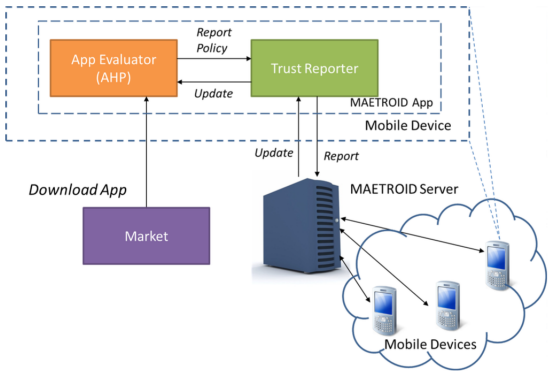

Figura 4.2: Arquitetura do MAETROID

Na Figura 4.2 pode-se notar a sua arquitetura, na qual, uma vez a aplicação instalada (MAETROID App), um módulo de avaliação implementa o algoritmo de Processo Analítico Hierárquico (*Analytic Hierarchy Process* (AHP)) (SAATY, 2003) (SAATY, 2002) em um aplicativo recém baixado, produzindo uma primeira decisão sobre o nível de confiança do aplicativo. Essa aplicação permite que o usuário colabore com uma rede, oferecendo informações sobre o comportamento de cada aplicação.

O relatório de confiança baseia-se em cinco parâmetros que compõem o seguinte conjunto: $(CR + BD + US + CL + MI)$ *(Crash, Battery Depletion, Usability, Credit Leakage, Misbehavior)* respectivamente. Cada valor encontra-se no intervalo entre $[0, 6] \in \mathbb{N}$.

O MAETROID-server recebe um valor para cada app avaliado pelo usuário, que varia entre $[1, 7]$ 1 para muito mau e 7 para muito bom, tendo-se 4 como valor neutro (s_o - pontuação inicial da confiança da aplicação). Tais resultados são fornecidos pelas seguintes equações:

$$r_i = 7 - (CR + BD + US + CL + MI)$$

$$s_i = (1 - \alpha)s_{i-1} + \alpha r_i, \quad s_i \in [1, 7]$$

Na equação 1, r_i representa a pontuação de *feedback* que está armazenada no MAETROID-server. Caso r_i seja negativo, o servidor altera o seu valor para 1, ou seja, uma aplicação com mau comportamento. Já na segunda equação, o valor de α é $\alpha(i) = \frac{\gamma}{i}, \quad \gamma = s_i - s_o$, na qual, $\alpha : \mathbb{R}^+ \rightarrow [0, 1]$ e s_i é o relatório completo do nível de confiança da aplicação avaliada.

Existem vários pontos que a solução proposta não aborda, tais como: reputação do usuário, servidores descentralizados e *feedback* de interações diretas sem a necessidade da intervenção do usuário.

4.3 DroidVulMon

O DroidVulMon – Android Based Mobile Device Vulnerability Analysis and Monitoring System (HAM et al., 2013) é um método efetivo para detectar vulnerabilidade em terminais móveis. O esquema proposto permite monitorar a existência de aplicativos maliciosos em vários terminais. Entretanto, na literatura, os trabalhos existentes abordam a detecção da maliciosidade em apenas um terminal móvel.

A proposta permite a coleta de informações relacionadas ao sistema, aos serviços, processos e redes de múltiplos terminais, com a finalidade de detectar *rootink attacks* (downloads, instalação e execução de códigos móveis com a autoridade do gerenciador do sistemas, ou seja, com autoridade de *root* (JANG et al., 2011)) e vulnerabilidade de segurança.

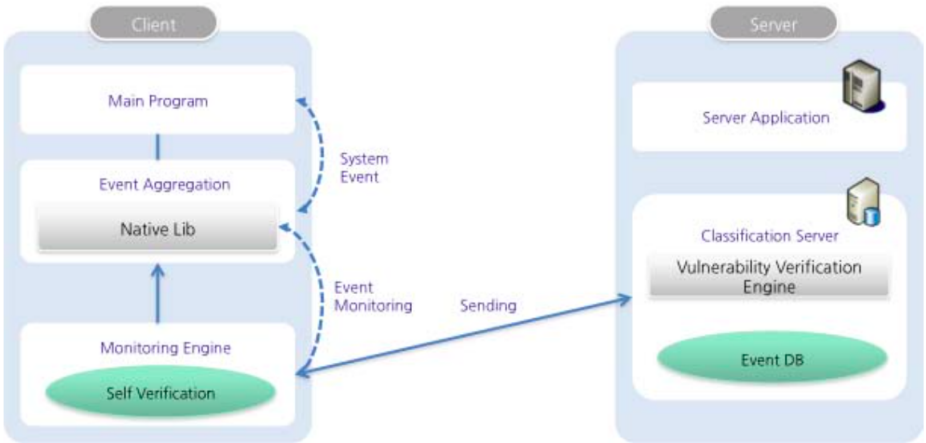

Figura 4.3: Arquitetura do DroidVulMon

Na Figura 4.3, os autores expõem sua arquitetura, constituída em duas partes

(cliente e servidor). Uma das principais partes é a do servidor, composta pela *engine* de verificação de vulnerabilidade e aplicação servidor. A *engine* do servidor é responsável pelo envio/recebimento de dados para/de *engine* de monitoramento de vulnerabilidade do cliente. Esses dados incluem informações atualizadas da *engine* e a lógica de verificação de gerenciamento de cada aplicação. Outra importante característica do lado servidor é diz respeito a um banco de dados que serve para armazenar todos os dados enviados e recebidos pelo servidor, ou seja, é nele que poderá ser verificado se o aplicativo é malicioso ou não.

Como mencionado anteriormente, a parte cliente possui uma *engine* de monitoramento de vulnerabilidade. Sua principal função é verificar se a aplicação está infectada. Caso a aplicação seja dada como maliciosa, cria-se um evento que é enviado para uma biblioteca nativa que possui um agente de verificação de *rooting attack*. Confirmada a informação de maliciosidade da aplicação, ela é intitulada como *"anormal"*. Após todo esse processo, o resultado do monitoramento da aplicação é compartilhado com o servidor e só então armazenado em seu banco para pesquisas posteriores. Um dos fatores que não são abordados na proposta é o fato de a análise de confiança não ser flexível, pois os autores focam apenas na detecção de *rooting attack*. Considerando-se que o termo confiança é subjetivo, a proposta em comento não contempla todas as necessidades do desenvolvedor.

4.4 Análise comparativa

Após observar os trabalhos relacionados à proposta, verifica-se que o *confidere* além de utilizar alguns recursos das outras propostas, consegue preencher lacunas que os mesmos possuem.

Em relação ao *TValue* (Neisse, 2012b), aproveitou-se a forma de avaliação da confiança, ou seja, essa avaliação é realizado através das interações diretas e indiretas realizadas pelo *trustee*. Entretanto, a fórmula proposta no *confidere* é diferente e qualquer componente da infraestrutura pode ser o trustee ou o trustor.

Quanto ao Maetroid, aproveitou-se a ideia de analisar o arquivo *AndroidManifest* da aplicação. Outra ideia utilizada do mesmo, foi a análise do comportamento da aplicação. Entretanto, a proposta não possibilita a criação de eventos

caso a aplicação se comporte de maneira inadequada ao que o usuário espera. O *confidere* possibilita a criação de eventos a partir de definições dos usuários, ou seja, caso a aplicação se comporte de maneira inesperada pelo usuário, é criado um evento e a aplicação sofre um interrupção.

E por fim, no que se refere ao DroidVulMon, aproveitou-se a analise do comportamento da aplicação em tempo de execução e a criação de eventos, caso a mesma se comporte de maneira indevida ao que o usuário espera. No entanto, a proposta não aplica-se para sistemas sensíveis ao contexto. Uma de suas limitações, é que a mesma torna suas regras de confiança como padrão para criar um evento, ou seja, o usuário não pode especificar qual comportamento ele considera inadequado. Já o *confidere*, possibilita ao desenvolvedor especificar o comportamento que o mesmo considera malicioso, ou seja, a proposta é flexível ao desenvolvedor.

Nota-se que todas os trabalhos expostos neste capítulo, contribuíram para a construção de uma proposta que consegue preencher boa parte das lacunas existentes em todos esses trabalhos.

5 Análise dos Resultados

Após o desenvolvimento do modelo de confiança proposto para validar componentes de Sistemas Sensíveis ao Contexto, foi realizado um experimento para garantir a competência do *confidere* em detectar a confiabilidade desses componentes. Neste capítulo, é descrito o experimento realizado.

5.1 Objetivos

O modelo proposto tem como principal objetivo garantir a segurança dos terminais móveis em relação aos códigos baixados dinamicamente. O *confidere* possui três funcionalidades, que são: a) selecionar o CP mais confiável e adequado às exigências da aplicação cliente; b) monitorar o comportamento do provedor; e c) pontuar o nível de confiança do mesmo.

O principal objetivo dessa avaliação é determinar a capacidade do *confidere* em detectar a maliciosidade do CP, isto é, se o CP comporta-se como esperado pela aplicação cliente e o tempo que o mesmo realiza essa atividade. Para essa finalidade, foram desenvolvidos CPs e aplicações que necessitam desses provedores para receber informações do ambiente do usuário.

5.2 Descrição do experimento

Para essa avaliação foram utilizados 6 dispositivos móveis com o sistema operacional android. Além disso, como exposto nas figuras 5.1, 5.3 e 5.2 foi criado um site que utilizou as seguintes tecnologias: HTML, CSS, JavaScript e PHP. Apartir desse site, o desenvolvedor do CP poderá criar seus repositórios e adicionar quantos CPs desejar.

Além disso foram desenvolvidos 9 CPs, dos quais, propositadamente, seis desses provedores não são considerados confiáveis em nenhum contexto cadastrado na base de dados de confiança já existente, a Tust_db. As informações contidas nessa base foram incluídas através dos feedbacks dos usuários em relação à utilização dos provedores

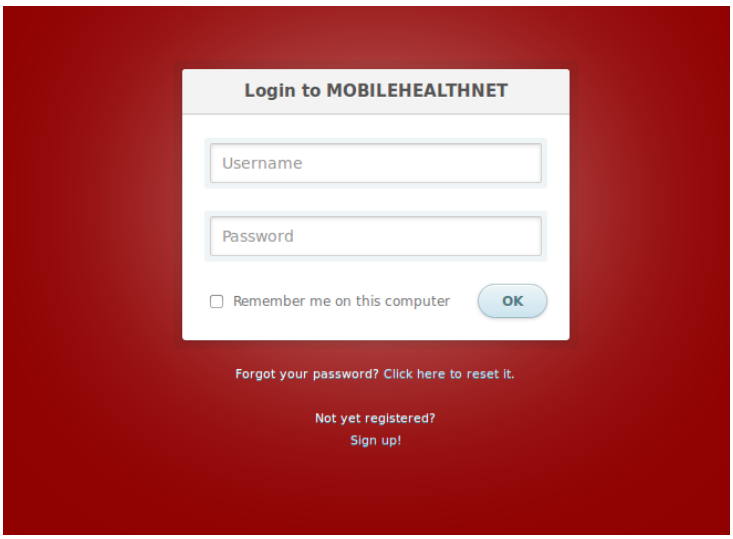

Figura 5.1: Login da Trust_db

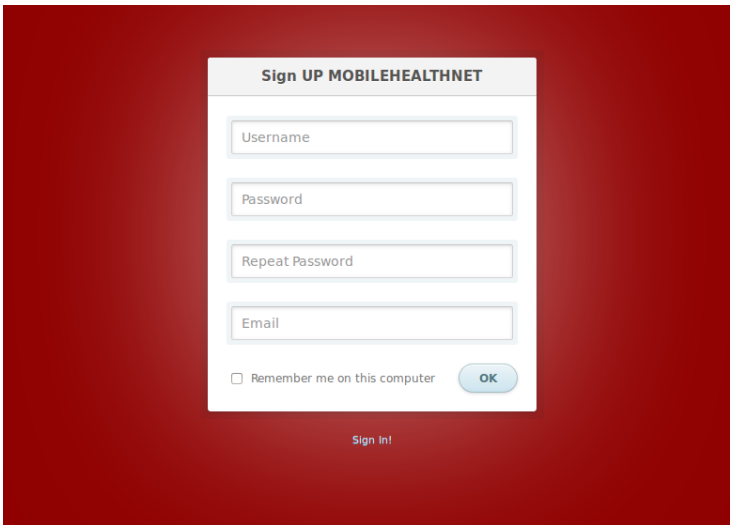

Figura 5.2: Tela de cadastro no Trust_db

em seus dispositivos móveis de acordo com seus respectivos contextos. Os demais CPs condizem com as exigências das aplicações e são considerados confiáveis em pelo menos um dos contextos.

Já as aplicações desenvolvidas para a avaliação possuem exigências particulares de CPs, por exemplo, a primeira aplicação exige que um CP de localização utilize somente de 10 a 15 por cento da CPU, 5 a 10 por cento da memória RAM e utilize o total de 1 a 3 mega de dados trafegados no tempo de um minuto e vinte segundos. Uma outra exigência da aplicação é que esse mesmo provedor só tenha permissão de acesso à internet e GPS. Todas essas informações constituem um contexto específico, o contexto 1. Os níveis de confiança desses provedores em cada contexto encontram-se armazenados na *Trust_DB*.

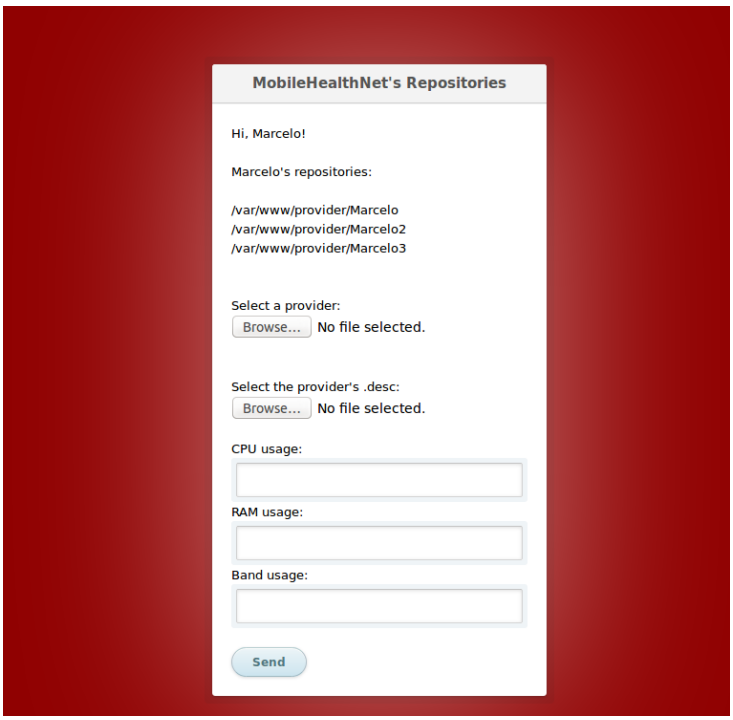

Figura 5.3: Tela de criação de repositório e envio de CP

Os experimentos de avaliação do *confidere* envolveram o monitoramento dos seguintes parâmetros: uso de CPU, consumo de memória RAM e largura de banda utilizada por cada CP. Caso o CP ultrapasse o uso do recurso especificado pela aplicação solicitante, o mesmo tem um ponto decrementado do seu nível de confiança e, além disso, é intitulado como não confiável. Caso contrário, é considerado confiável.

Além dessa verificação, o *confidere* ainda faz uma leitura das permissões da utilização dos recursos do dispositivo móvel em um arquivo chamado *AndroidManifest.xml* do CP no momento em que o mesmo é instalado pelo CMS. É nesse aquivo que o desenvolvedor colocará as permissões de acesso aos recursos do dispositivo móvel. Caso o CP solicite um recurso não permitido pela aplicação cliente, ele será excluído automaticamente e intitulado como não confiável no contexto em que o mesmo encontra-se em avaliação.

Apesar de o android já garantir essa segurança para seu usuário, os CPs da arquitetura do *MobileHealthNet*, ao contrário de qualquer aplicação android, não solicitam ao usuário a permissão da utilização dos recursos computacionais do dispositivo móvel, pois seu gerenciamento é realizado também por uma aplicação android, o CMS. Sendo assim, o mesmo herda todas as permissões dessa aplicação gerenciadora, ou seja, as permissões são herdadas do CMS. Com a nossa proposta, é realizada uma verificação

Tabela 5.1: Detalhamento dos provedores e suas respectivas permissões

Aplicação	**Provedores**	**Permissão**	**Uso da CPU**	**Uso de RAM**	**Uso de Banda**	**Contexto**
Aplicação 1	LOCATION 1	GPS	(10-15)%	(5-10)%	(1-3)MB	1
	TIME 1	SET_TIME	(10-15)%	(5-10)%	(1-3)MB	7
	BATTERY 1	BATTERY_STATS	(10-15)%	(5-10)%	(1-3)MB	4
Application 2	LOCATION 2	GPS AND CONTACTS	(16-20)%	(11-15)%	(1-5)MB	2
	TIME 2	SET_TIME AND BLUETOOTH	(16-20)%	(11-15)%	(1-5)MB	8
	BATTERY 2	BATTERY_STATS AND BLUETOOTH	(16-20)%	(11-15)%	(1-5)MB	5
Application 3	LOCATION 3	GPS, CONTACTS AND BLUETOOTH	(21-25)%	(16-20)%	(1-7)MB	3
	TIME 3	SET_TIME AND BLUETOOTH	(21-25)%	(16-20)%	(1-7)MB	9
	CP's BATERY 3	BATTERY_STATS, BLUETOOTH AND CONTACTS	(21-25)%	(16-20)%	(1-7)MB	6

das permissões antes do inicio da coleta das informações, pois caso as permissões existentes no arquivo *AndroidManifest.xml* do CP possuam permissões diferentes do exigido pela aplicação cliente, o mesmo sofre uma interrupção e é enviada uma mensagem ao componente ECP, que, por sua vez, irá decrementar um ponto em seu nível de confiança.

Como exposto na tabela 5.1, os contextos foram divididos da seguinte forma: i) 1, 2, 3: específicos para provedores de localização; ii) 4, 5 e 6: específicos para provedores de bateria; e iii) 7, 8, 9: específicos para provedores de tempo.

Apesar de a tabela 5.1 não informar a permissão de acesso à internet, todos os CPs em análise possuem essa permissão. Para facilitar a leitura das permissões, evitou-se colocar todas que o desenvolvedor exige para que um CP seja considerado confiável. Em exemplo, temos os provedores de localização que possuem a permissão para o acesso ao GPS, entretanto as permissões que são utilizadas para acessar o sensor de localização do dispositivo móvel contidas no arquivo *AndroidManifest.xml*, são as ACCESS_FINE_LOCATION e ACCESS_MOCK_LOCATION.

5.3 Resultados e Análise do experimento

A tabela 5.2 contempla todos os testes realizados para avaliar o nosso modelo de confiança. O tempo utilizado para avaliar o comportamento de cada CP foi de um minuto e vinte segundos. Outro fato importante é que durante os testes foi possível observar que o *confidere* obteve 100% de êxito ao conseguir detectar no tempo especificado para a avaliação, o nível de confiabilidade dos CPs de acordo com as exigências das aplicações.

Em exemplo, temos o CP de localização da aplicação 1, em que o *confidere* consegue detectar a maliciosidade do CP nos contextos 1 e 2 em 09 e 11 segundos, respectivamente. Em relação ao contexto 3, para que o *confidere* afirme que o mesmo é confiável, é necessário que o tempo estimado para a avaliação chegue ao fim, ou seja, durante a avaliação o CP não poderá agir de maneira diferente ao que a aplicação cliente considera confiável.

Já com o CP de bateria na mesma aplicação, houve um pequeno atraso se comparado ao de localização, pois o tempo de detecção da maliciosidade foi de 15 segundos no contexto 5 e 11 segundos no contexto 6. Como o contexto 4 estava de acordo

Tabela 5.2: Resultado das avaliações realizadas para detecção da maliciosidade dos CPs

Aplicação	**Provedor**	**Contexto**	**Nível de confiança**	**Tempo de Detecção - mm:ss**
Aplicação 1	LOCATION 1	1	-2	00:11
		2	-1	00:12
		3	4	01:20
	BATTERY 1	4	2	01:20
		5	-1	00:15
		6	-1	00:11
	TIME 1	7	3	01:20
		8	-3	00:11
		9	-3	00:13
Aplicação 2	LOCATION 2	1	-3	00:10
		2	-1	00:09
		3	1	01:20
	BATTERY 2	4	2	01:20
		5	-1	00:18
		6	-1	00:16
	TIME 2	7	3	01:20
		8	-2	00:07
		9	-1	00:05
Aplicação 3	LOCATION 3	1	-2	00:11
		2	-1	00:12
		3	4	01:20
	BATTERY 3	4	2	01:20
		5	-2	00:30
		6	-2	00:45
	TIME 3	7	1	01:20
		8	-1	00:06
		9	-2	00:09

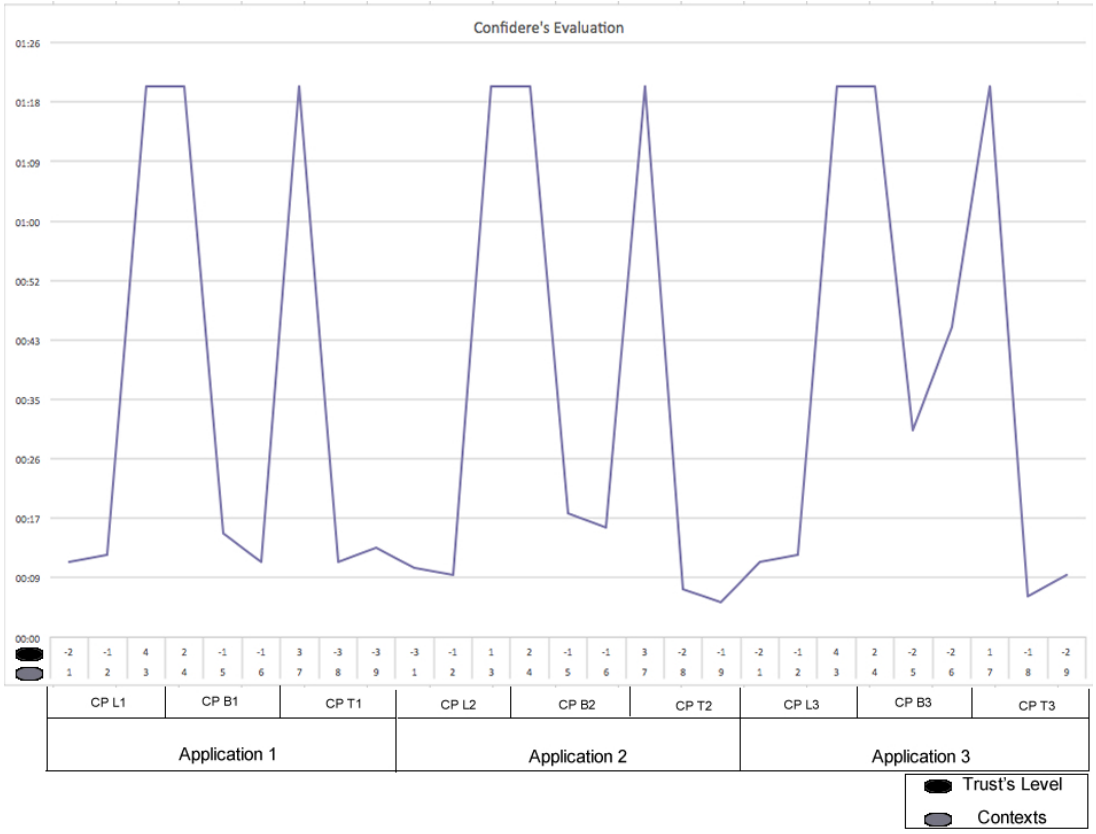

Figura 5.4: Avaliação do *Confidere*

com as especificações da aplicação cliente e também o *confidere*, não detectou nenhum comportamento indevido durante o tempo da avaliação, o mesmo foi considerado como confiável.

Observa-se que o *confidere* leva um tempo maior para detectar a maliciosidade nos CPs de bateria. Tal se deve à forma que o CP foi escrito, pois a sobrecarga nos recursos computacionais só é ativada depois de um determinado tempo após o inicio da coleta dos dados. Vale ressaltar que não podemos afirmar que o *confidere* irá se comportar do mesmo jeito, caso o tempo de análise seja maior, pois todos os testes realizados foram no intervalo de um minuto e vinte segundos.

Outra análise realizada pelo *confidere* foi o consumo de banda do CP, ou seja, a quantidade de dados enviados e recebidos utilizados pelo CP, em que o mesmo está diretamente ligado ao tempo que ele passa coletando informações. Na nossa avaliação, o uso de banda para todos os CPs é de no mínimo de 1 mb e no máximo de 7 mb, pois o tempo para a avaliação é curto, causando, assim, a impossibilidade de grandes quantidades de transferência de dados.

O tempo estipulado para avaliação do *confidere*, na detecção da maliciosidade

dos CPs, é curto, pois o mesmo está diretamente ligado ao modo como foi escrito os CPs. A sobrecarga do uso dos recursos computacionais do dispositivo móvel é iniciada logo após a autorização para o inicio da coleta de dados, ou seja, os métodos desenvolvidos para sobrecarregar o uso de CPU e memória são iniciados juntamente com os métodos que realizam a coleta de informações contextuais.

6 Conclusão e Trabalhos Futuros

Constata-se, atualmente, um crescimento significativo na utilização de aplicações ubíquas, o qual resultou na mudança das atividades diárias das pessoas. Aplicações para ambientes ubíquos obrigam-se a atender um conjunto de novos desafios, tal como a capacidade da aplicação se adaptar ao ambiente do usuário, ou seja, torná-las sensíveis ao contexto. Uma das proposta iniciais do projeto *MobileHealthNet* é reduzir o esforço no desenvolvimento de aplicações desse tipo.

Aplicações sensíveis ao contexto necessitam de qualidade em suas informações, principalmente as que são voltadas ao domínio da saúde, pois que serão cruciais para a tomada de decisão dos profissional da saúde em relação ao acompanhamento de pacientes. Visto isso, a implementação de parâmetros de QoC tornou-se essencial na infraestrutura do projeto, em específico o *Trustworthiness* (Confiabilidade).

Nessa dissertação foi proposto um modelo de confiança para Sistemas Sensíveis ao Contexto. Esse modelo foi baseado em conceitos de confiança tirados da psicologia e trazidos para computação através de uma representação matemática. Este trabalho também avaliou o estado da arte dos modelo de confiança já existentes, realizando um trabalho comparativo entre eles e o *Confidere*.

Visando suprir a falta de confiança nos componentes do *middleware* *MobileHealthNet*, em particular o CP, o trabalho também propõe uma arquitetura capaz de detectar a maliciosidade desse componente. Outra característica da arquitetura proposta é a flexibilidade, ou seja, ela é capaz de se adaptar às exigências de avaliações de confiança de qualquer desenvolvedor.

Finalmente, a validação da infraestrutura do *Confidere*, foi realizada através de um experimento que tem por cerne a detecção da maliciosidade do CP em diferentes contextos. Esse experimento nos permitiu afirmar o nível de confiabilidade do CP e ter a certeza que se o mesmo se comportar de maneira indevida, ele será excluído do dispositivo móvel.

6.1 Contribuições

No contexto desse trabalho dissertativo, é possível perceber as seguintes contribuições:

1. Levantamento e análise comparativa do estado da arte de modelos de confiança para Sistemas Sensíveis ao Contexto;
2. Concepção, implementação e avaliação de um modelo de confiança para Sistemas Sensíveis ao Contexto;
3. Projeto e implementação de uma infraestrutura de analise de componentes que garante o nível de confiança dos mesmos.

6.2 Resultados

Através do trabalho realizado, vários resultados foram alcançados. Destacam-se entre eles:

1. Elaboração de texto contendo uma análise comparativa de modelos de confiança para Sistemas Sensíveis ao Contexto;
2. Disponibilização de uma arquitetura para análise da confiança de componentes responsáveis por prover as informações contextuais de Sistemas Sensíveis ao Contexto;
3. Publicação de artigo científico:
    - Resposta dia 16 de Agosto de 2015

6.3 Trabalhos Futuros

A partir desse trabalho inicial, alguns trabalhos futuros podem ser apontados:

1. Complementar a garantia de confiança com a implantação da análise do consumo de banda ligados ao tempo que o CP passa consumindo informações contextuais;

2. Realizar experimentos com uma quantidade maior de repositórios e provedores de contexto, com o intuito de simular o comportamento do *confidere* em um ambiente real.

3. Otimizar o *confidere* para a infraestrutura da Internet das Coisas, com o objetivo de garantir a analise da confiança de todos os componentes de maneira customizada ao usuário.

Bibliografia

ADAMS, B.; PHUNG, D. Q.; VENKATESH, S. Sensing and using social context. *TOMCCAP*, v. 5, n. 2, 2008.

BALDAUF, M.; DUSTDAR, S.; ROSENBERG, F. A survey on context-aware systems. *Int. J. Ad Hoc Ubiquitous Comput.*, Inderscience Publishers, Inderscience Publishers, Geneva, SWITZERLAND, v. 2, n. 4, p. 263–277, jun. 2007. ISSN 1743-8225. Disponível em: <http://dx.doi.org/10.1504/IJAHUC.2007.014070>.

BATISTA, R. C. *Uma Infraestrutura de Comunicacao Centrada em Dados para Redes Sociais Moveis*. Dissertação (Mestrado) — LSD, Universidade Federal do Maranhão, 2013.

BELLAVISTA, P. et al. A survey of context data distribution for mobile ubiquitous systems. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 44, n. 4, p. 24:1–24:45, set. 2012. ISSN 0360-0300. Disponível em: <http://doi.acm.org/10.1145/2333112.2333119>.

BOLCHINI, C. et al. And what can context do for data? *Commun. ACM*, v. 52, n. 11, p. 136–140, 2009.

BUCHHOLZ, T.; SCHIFFERS, M. Quality of context: What it is and why we need it. In: *In Proceedings of the 10th Workshop of the OpenView University Association: OVUA'03*. [S.l.: s.n.], 2003.

CHEN, G.; KOTZ, D. *Solar: A pervasive-computing infrastructure for context-aware mobile applications*. [S.l.], 2002.

CHETAN, S.; RANGANATHAN, A.; CAMPBELL, R. Towards fault tolerance pervasive computing. *Technology and Society Magazine, IEEE*, v. 24, n. 1, p. 38–44, 2005. ISSN 0278-0097.

DAVID, L. et al. A communication middleware for scalable real-time mobile collaboration. In: *Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2012 IEEE 21st International Workshop on*. [S.l.: s.n.], 2012. p. 54–59. ISSN 1524-4547.

DAVID, L. et al. A large-scale communication middleware for fleet tracking and management. In: *XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, Sessão de Ferramentas*. Ouro Preto - MG, Brasil: [s.n.], 2012. (Anais do SBRC 2012).

DEY, A. K. Understanding and using context. *Personal Ubiquitous Comput.*, Springer-Verlag, London, UK, UK, v. 5, n. 1, p. 4–7, jan. 2001. ISSN 1617-4909. Disponível em: <http://dx.doi.org/10.1007/s007790170019>.

DINI, G. et al. Evaluating the trust of android applications through an adaptive and distributed multi-criteria approach. In: *Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on*. [S.l.: s.n.], 2013. p. 1541–1546.

FILHO, J. L. T. P. *Sistemas de Tipos para Capturar Informação de Contexto em Computação Pervasiva*. 2010. Monografia (Graduação em Ciência da Computação), UFSM (Universidade Federal de Santa Maria), RS, Brazil.

HAM, Y. J. et al. Droidvulmon – android based mobile device vulnerability analysis and monitoring system. In: *Next Generation Mobile Apps, Services and Technologies (NGMAST), 2013 Seventh International Conference on*. [S.l.: s.n.], 2013. p. 26–31.

HARDIN, R. *Trust and Trustworthiness*. [S.l.]: Series on Trust, 2002.

HELD, A.; BUCHHOLZ, S.; SCHILL, A. Modeling of context information for pervasive computing applications. In: *Procceding of the World Multiconference on Systemics, Cybernetics and Informatics*. [S.l.: s.n.], 2002.

HENRICKSEN, K.; INDULSKA, J.; RAKOTONIRAINY, A. Modeling context information in pervasive computing systems. In: *Proceedings of the First International Conference on Pervasive Computing*. London, UK, UK: Springer-Verlag, 2002. (Pervasive '02), p. 167–180. ISBN 3-540-44060-7. Disponível em: <http://dl.acm.org/citation.cfm?id=646867.706693>.

HUEBSCHER, M. C.; MCCANN, J. A. Adaptive middleware for context-aware applications in smart-homes. In: *Proceedings of the 2Nd Workshop on Middleware for Pervasive and Ad-hoc Computing*. New York, NY, USA:

ACM, 2004. (MPAC '04), p. 111–116. ISBN 1-58113-951-9. Disponível em: <http://doi.acm.org/10.1145/1028509.1028511>.

ISTEPANIAN ROBERT, S. L. S. C. P. C. S. R. *M-Health: Emerging Mobile Health Systems*. [S.l.]: Springer Science Business Media, Inc., 233 Spring Street, New York, NY 10013, USA, 2006.

JANG, W.-J. et al. Rooting attack detection method on the android-based smart phone. In: *Computer Science and Network Technology (ICCSNT), 2011 International Conference on*. [S.l.: s.n.], 2011. v. 1, p. 477–481.

JØSANG, A. A logic for uncertain probabilities. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, World Scientific Publishing Co., Inc., River Edge, NJ, USA, v. 9, n. 3, p. 279–311, jun. 2001. ISSN 0218-4885.

JøSANG, A.; KESER, C.; DIMITRAKOS, T. Can we manage trust? In: HERRMANN, P.; ISSARNY, V.; SHIU, S. (Ed.). *Trust Management*. [S.l.: s.n.], 2005, (Lecture Notes in Computer Science, v. 3477).

KARAM, A.; MOHAMED, N. Middleware for mobile social networks: A survey. In: *System Science (HICSS), 2012 45th Hawaii International Conference on*. [S.l.: s.n.], 2012. p. 1482–1490. ISSN 1530-1605.

KRAUSE, M.; HOCHSTATTER, I. Challenges in modelling and using quality of context (qoc). In: MAGEDANZ, T. et al. (Ed.). *Mobility Aware Technologies and Applications*. Springer Berlin Heidelberg, 2005, (Lecture Notes in Computer Science, v. 3744). p. 324–333. ISBN 978-3-540-29410-8. Disponível em: <http://dx.doi.org/10.1007/11569510$_3$1¿.

LUBKE, R.; SCHUSTER, D.; SCHILL, A. Mobilisgroups: Location-based group formation in mobile social networks. In: *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*. [S.l.: s.n.], 2011. p. 502–507.

MALCHER, M. et al. A middleware supporting adaptive and location-aware mobile collaboration. In: *Mobile Context Workshop: Capabilities, Challenges and Applications, Adjunct Proceedings of UbiComp*. [S.l.: s.n.].

MANZOOR, A. *Quality of context in pervasive systems: models, techniques, and applications*. Tese (Doutorado) — Computer Science, 2010.

MANZOOR, A.; TRUONG, H.-L.; DUSTDAR, S. Quality of context: Models and applications for context-aware systems in pervasive environments. *The Knowledge Engineering Review, Special Issue on Web and Mobile Information Services*, 2011.

MIRAOUI, M. et al. Dynamic context-aware and limited resources-aware service adaptation for pervasive computing. *Adv. Soft. Eng.*, Hindawi Publishing Corp., New York, NY, United States, v. 2011, p. 7:7–7:7, jan. 2011. ISSN 1687-8655. Disponível em: <http://dx.doi.org/10.1155/2011/649563>.

Neisse, R. *Trust and privacy management support for context-aware service platforms*. Tese (Doutorado) — University of Twente, Enschede, March 2012.

Neisse, R. *Trust and privacy management support for context-aware service platforms*. Tese (Doutorado) — University of Twente, Enschede, the Netherlands, March 2012. SIKS Dissertation Series No. 2012-09. Disponível em: <http://doc.utwente.nl/79970/>.

PESSOA, R. M. *Infraware: Um Middleware de Suporte a Aplicações Sensíveis ao Contexto*. Dissertação (Mestrado) — Universidade Federal do Espírito Santo (UFES), Vitória, ES, Brazil, 2006.

PINHEIRO, D. N. *MHNCS: Um middleware para o desenvolvimento de aplicacoes moveis cientes de contexto com requisitos de QoC*. Dissertação (Mestrado) — Universidade Federal do Maranhão, 2014.

SAATY, T. L. Decision making with the analytic hierarchy process. *Scientia Iranica*, 2002.

SAATY, T. L. Decision-making with the AHP: Why is the principal eigenvector necessary. *European Journal of Operational Research*, v. 145, p. 85–91, 2003.

Sheikh, K.; Wegdam, M.; Sinderen, M. van. Middleware support for quality of context in pervasive context-aware systems. In: *Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2007*. IEEE Computer Society Press, 2007. p. 461–466. Disponível em: <http://doc.utwente.nl/62009/>.

SILVA, L.; ENDLER, M.; RORIZ, M. Mr-udp: Yet another reliable user datagram protocol, now for mobile nodes. *Monografias em Ciencia da Computacao*, v. 6, 2013.

SOARES, P. R. da S. *Desenvolvimento de Propaganda Interativa e Sensível ao Contexto para TV Digital*. 2010. Monografia (Graduação em Ciência da Computação), UFPE (Universidade Federal de Pernambuco), Brazil.

STARK, J. E. *Trust in Distributed Computing*. Dissertação (Mestrado) — The University of Guelph, 2014.

STRANG, T.; LINNHOFF-POPIEN, C. A context modeling survey. In: *In: Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England*. [S.l.: s.n.], 2004.

TELES, A.; SILVA, F. J. da Silva e; BATISTA, R. Security and privacy in mobile social networks. In: _____. Springer: Security and Privacy Preserving in Social Networks, 2013. (Lecture Notes in Social Networks).

TELES, A. S. et al. Redes sociais moveis: Conceitos, aplicações e aspectos de segurança e privacidade. In: _____. [S.l.: s.n.], 2013. cap. 2.

VIEIRA, V.; TEDESCO, C. A. C. T. C. R. G. d. F. J. C. L. R. O. P. Uso e representação de contexto em sistemas computacionais. In: *Tópicos em Sistemas Interativos e Colaborativos*. São Carlos, São Paulo, Brazil: [s.n.], 2006. p. 127–166. ISBN 978-1-4244-3534-0.

VIEIRA, V.; TEDESCO, P.; SALGADO, A. C. A process for the design of context-sensitive systems. In: *Proceedings of the 2009 13th International Conference on Computer Supported Cooperative Work in Design*. Washington, DC, USA: IEEE Computer Society, 2009. (CSCWD '09), p. 143–148. ISBN 978-1-4244-3534-0. Disponível em: <http://dx.doi.org/10.1109/CSCWD.2009.4968049>.

VIEIRA, V.; TEDESCO, P.; SALGADO, A. C. Designing context-sensitive systems: An integrated approach. *Expert Syst. Appl.*, Pergamon Press, Inc., Tarrytown, NY, USA, v. 38, n. 2, p. 1119–1138, fev. 2011. ISSN 0957-4174. Disponível em: <http://dx.doi.org/10.1016/j.eswa.2010.05.006>.

WEBSTER, J. G. *The measurement, instrumentation, and sensors handbook*. [S.l.]: Berlin: Springer, 1999.

WEISER, M. The computer for the twenty-first century. *Scientific American*, v. 9, p. 94–100, set. 1991. Disponível em: <http://www.ubiq.com/hypertext/weiser/SciAmDraft3.html>.

WIBISONO, W.; ZASLAVSKY, A.; LING, S. Towards a service-oriented approach for managing context in mobile environment. In: BOUGUETTAYA, A.; KRUEGER, I.; MARGARIA, T. (Ed.). *Service-Oriented Computing – ICSOC 2008*. Springer Berlin Heidelberg, 2008, (Lecture Notes in Computer Science, v. 5364). p. 210–224. ISBN 978-3-540-89647-0. Disponível em: <http://dx.doi.org/10.1007/978-3-540-89652-4$_1$8¿.

WOOLTHUIS B HILLEBRAND, B. N. R. Trust, contract and relationship development. *Journal of Behavioral and Experimental Economics*, v. 34, n. 3, p. 421–424, 2005.