Apoio ao Desenvolvimento de Aplicações de Tempo Real Sensíveis a Contexto Semântico

Ernesto Fonseca Veiga, Guilherme Melo e Maranhão, Renato de Freitas Bulcão Neto
Instituto de Informática
Universidade Federal de Goiás
Goiânia – GO, Brasil
ernestofonsecaveiga@gmail.com, guilherme82@gmail.com, renato@inf.ufg.br

ABSTRACT

This paper describes the Hermes infrastructure to supporting the development of context-aware applications. Hermes provides software components that handle the semantics of context information as well as event management through a middleware for real-time distributed communication. A case study on monitoring of human vital signs has been developed for validation purposes of the Hermes infrastructure.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures

General Terms

Algorithms, Design, Management.

Keywords

Infrastructure, context, semantics, real-time, DDS.

1. CONTEXTO TEÓRICO

Sistemas sensíveis a contexto caracterizam-se por variar de um limitado número de fontes de contexto a dezenas de milhares de sensores [10], o que demonstra que os ambientes aos quais esses sistemas são implantados possuem uma natureza heterogênea e distribuída [2]. Essas duas características implicam em um aumento proporcional na complexidade e quantidade de regras de negócio para interpretar dados adquiridos de sensores, bem como para fornecer informações de relevância para aplicações de usuários [9], o que onera o desenvolvimento destas quanto ao tempo e à complexidade.

Neste interim, o trabalho clássico de [3] propôs um arcabouço conceitual com sete requisitos para facilitar o desenvolvimento de software dessa natureza. Ao implementá-los, o *Context Toolkit* [3] foi projetado com componentes que fornecem serviços para descoberta, aquisição, agregação e interpretação de contexto, cujo objetivo principal é fornecer o suporte arquitetural para apoio ao desenvolvimento de aplicações sensíveis a contexto, além de abstrair destas detalhes de aquisição e interpretação do contexto, notificação de eventos e comunicação distribuída.

WebMedia'14 November 18 - 21, 2014, Brazil

2. IDENTIFICAÇÃO DO PROBLEMA

Corroborando com requisitos propostos em [3], a literatura tem identificado a necessidade da evolução quanto a expressividade, formalidade e uso de padrões na representação e interpretação de informação contextual [10]. Isto se justifica mais uma vez em função da diversidade de sensores, criando demandas por regras mais complexas para interpretar e, ao mesmo tempo, por resultados dessa interpretação mais relevantes e precisos nas aplicações. Neste sentido, trabalhos como o de [9] sugerem o tratamento da semântica das informações de contexto e o desenvolvimento de infraestruturas de software inteligentes.

Aliado à modelagem semântica, outro tema de pesquisa relevante nesta área tem sido a comunicação em tempo real, ou com o mínimo de latência, requisito de várias aplicações [10]. Acrescenta-se que esse requisito de comunicação deva também ser projetado e implementado de forma a facilitar o desenvolvimento de aplicações que requerem esse aspecto.

3. OBJETIVO

Diante do exposto, propõe-se uma infraestrutura de apoio ao desenvolvimento de aplicações de tempo real sensíveis a contexto semântico, denominada *Hermes*, que possui como referência o clássico arcabouço conceitual de [3], bem como o modelo arquitetural de seu *Context Toolkit*, acrescentando a estes a modelagem semântica de contexto e a comunicação em tempo real recomendados em [9] e [10].

4. CONTRIBUIÇÕES ESPERADAS

Através da infraestrutura *Hermes*, espera-se contribuir com o desenvolvimento de aplicações sensíveis a contexto de forma desacoplada com serviços básicos, a interoperabilidade semântica entre aplicações, e a comunicação distribuída e transparente em tempo real. *Hermes* oferece os serviços de aquisição, modelagem, persistência, interpretação e disseminação de contexto aos desenvolvedores, que ficam responsáveis apenas pelas aplicações e suas regras de negócio.

5. METODOLOGIA ADOTADA

Com base na literatura revisada, a primeira etapa do trabalho consistiu em definir o escopo da infraestrutura *Hermes*, com base nos requisitos elencados [3] [9] [10]. A partir

desses estudos foram definidos os serviços oferecidos e também a arquitetura de componentes, fortemente influenciada pela arquitetura do *Context Toolkit* [3].

A segunda etapa, que ainda se encontra em desenvolvimento, é a implementação da infraestrutura Hermes. A implementação está sendo realizada na linguagem Java, sobre o arcabouço Apache Jena, que oferece APIs para manipulação semântica de dados. Também está sendo utilizado o middleware de tempo real CoreDX, que implementa o padrão OMG DDS (*Data Distribution Service*).

Para validar os componentes implementados da infraestrutura *Hermes*, tem sido desenvolvidos protótipos que simulam o comportamento de sensores e aplicações em um cenário de monitoramento de sinais vitais humanos, cenário que reúne as características de sensibilidade a contexto e comunicação distribuída em tempo real.

6. ESTÁGIO ATUAL DO TRABALHO

6.1 Arquitetura

A Figura 1 apresenta os componentes da infraestrutura *Hermes*, a saber: *Hermes Base*, *Hermes Widget*, *Hermes Interpreter* e *Hermes Aggregator*, responsáveis, respectivamente, pelos serviços de *i*) comunicação distribuída em tempo real; *ii*) aquisição, modelagem semântica, armazenamento e disseminação de contexto; *iii*) interpretação de contexto; e *iv*) agregação de contexto.

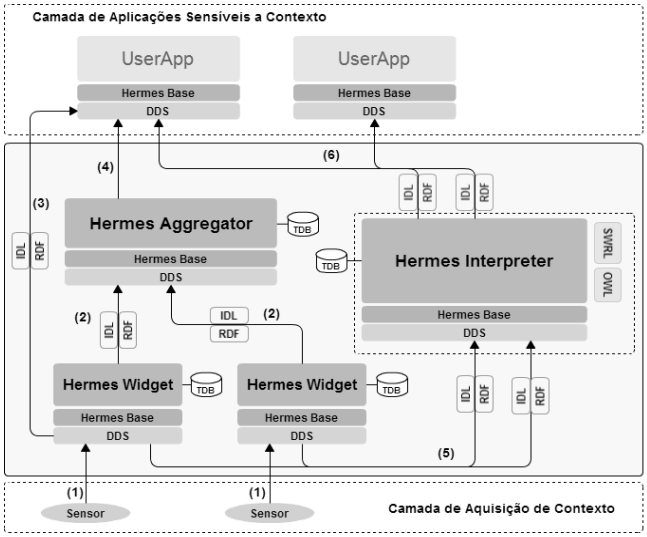

Figure 1: Arquitetura da infraestrutura *Hermes*.

Para atender aos requisitos de modelagem semântica e interoperabilidade de informação contextual, *Hermes* utiliza especificações padrão para intercâmbio de dados e para representação e lógica de informação da Web Semântica [6].

Para atender ao requisito de comunicação distribuída e transparente em tempo real, a infraestrutura implementa a comunicação e integração, entre serviços e destes com aplicações, conforme a especificação OMG DDS.

A infraestrutura *Hermes* ainda atende aos requisitos de projeto elencados em [9]: compartilhamento de informações em tempo-real, monitoramento e notificação de eventos, extensibilidade e fornecimento de API de fácil acesso.

6.1.1 *Hermes Base*

Desenvolvido com referência no projeto do componente *Base Object* do *Context Toolkit*, oferece uma interface para que os componentes *Hermes Widget*, *Hermes Aggregator*, *Hermes Interpreter* e aplicações de usuários se comuniquem com o *middleware* DDS. Esse desacoplamento confere ao *Hermes* a possibilidade de conectar outros sensores (com seus respectivos *widgets*), *interpreters* e aplicações de usuário sem que estes conheçam o *middleware* DDS, ou qualquer outro *middleware* que forneça a integração.

É por meio do *Hermes Base*, portanto, que os componentes supracitados se comunicam entre si, registrando, publicando ou assinando tópicos. Essa interface possibilita também aos componentes publicadores e assinantes que realizem a parametrização dos seus contratos de serviço (*QoS*). As aplicações e os demais componentes interagem somente com as classes e interfaces do *Hermes Base*, a saber: *Listener*, *BaseObject* e as classes referentes às políticas de *QoS*. Essa modelagem permite que uma alteração de *middleware* seja transparente para as aplicações e demais componentes.

A especificação DDS, além dos aspectos de tempo real, é baseada no modelo *publish/subscribe*. O componente *Hermes Base* utiliza este modelo para fornecer uma interface para a criação, assinatura e publicação de tópicos dentro da infraestrutura sensível a contexto. Cada componente ou aplicação que deseja criar/publicar/assinar tópicos, especifica os dados dos respectivos tópicos em um arquivo de configuração `topics.json`, que será utilizado por uma instância de *Hermes Base* para realizar as operações relativas a tópicos e comunicação de informações de contexto. O estado atual de implementação do componente *Hermes Base* é *concluído*.

6.1.2 *Hermes Widget*

Componente responsável pela: *i*) aquisição das informações de contexto, coletadas através de sensores físicos e/ou lógicos, como apresentado pelo item (1) da Figura 1; *ii*) modelagem semântica do contexto adquirido; *iii*) disponibilização das informações de contexto para as aplicações, como indicado pelos itens (2), (3) e (5); e *iv*) persistência das informações de contexto, provendo uma interface comum para consulta e notificação. A implementação deste componente é está *em andamento*, mas já apresenta uma versão funcional.

Aquisição

Cada *Hermes Widget* é responsável por uma parte específica do contexto e oferece interface para um determinado sensor, seja este físico ou lógico, se tornando responsável por todos os eventos registrados através deste. O tipo e o número de *Widgets* utilizados por uma aplicação depender ao do ambiente onde se deseja obter contexto, e das possíveis informações de contexto que podem ser disponibilizadas.

Modelagem Semântica

As informações coletadas dos sensores são consideradas de baixo nível, possuindo pouca expressividade. Entretanto, estas informações estão diretamente relacionadas a um determinado domínio, e dentro deste possuem significado, seja particularmente ou em conjunto com outras informações. Para prover esta expressividade às informações de contexto, o componente *Hermes Widget* realiza a modelagem semântica das informações adquiridas.

Em cada *Hermes Widget* é realizado um processo de conversão dos dados recebidos do respectivo sensor para o for-

mato padrão RDF[1] de intercâmbio de dados. Esta conversão possui como base o domínio de conhecimento no qual a informação está inserida. Como requisito, o componente *Hermes Widget* deve conhecer este domínio, que é representado de forma computacional através de uma ontologia.

Utilizando o conhecimento de um determinado domínio, é possível criar indivíduos que instanciam os conceitos e relações modeladas por uma ontologia e descrever as informação de contexto. O componente *Hermes Widget* utiliza este tipo de modelagem para prover, entre outras vantagens: forte validação, interoperabilidade e expressividade.

Disseminação

Para notificar as aplicações ou também outros componentes assinantes de informações de contexto, o *Hermes Widget* utiliza o *middleware* DDS através do componente *Hermes Base*, que fornece a interface para os serviços de comunicação distribuída em tempo real, como descrito na Subseção 6.1.1.

Persistência

Cada *Hermes Widget* possui um banco de dados local, no qual armazena todas as informações após a etapa de modelagem de contexto. Para isso, utiliza-se o repositório TDB do Apache Jena, que suporta o armazenamento nativo de triplas, possuindo melhor desempenho e escalabilidade em comparação a um banco de dados relacional. Para armazenar os dados, o TDB cria um conjunto de arquivos de índices e nós, sobre os quais também podem ser realizadas inserções, atualizações e consultas.

Assim, o *Hermes Widget* separa as camadas de aqui-sição e de utilização do contexto, sendo responsável por uma parte determinada do contexto de uma entidade, que é especificada através dos seus atributos. Aplicações de usuários e outros componentes podem assinar um *Hermes Widget* para receberem notificações quando um novo evento é registrado.

6.1.3 Hermes Interpreter

O componente *Hermes Interpreter* é responsável pela realização de inferências através na semântica de modelos de contexto baseados em ontologias e em regras. Componentes ou aplicações que necessitam dessas informações são notificadas, conforme o item (6) da Figura 1. As regras de negócio do contexto, descritas na ontologia, são encapsuladas e mapeadas as situações de contexto de alto nível, como atributos, subclasses e relações ontológicas. O componente ainda possui arquitetura flexível para a incorporação de novas técnicas de inferência, conforme necessário. A implementação deste componente está *em estágio avançado de desenvolvimento*, contando com uma versão funcional.

6.1.4 Hermes Aggregator

O componente *Hermes Aggregator* agrega diferentes tipos de informações de contexto obtidas por meio da fusão de *Hermes Widgets*. Dessa forma, como indicado pelo item (4) na Figura 1, o *Hermes Aggregator* pode fornecer a aplicações todo o contexto relacionado a uma entidade em particular, como por exemplo, aferições de diferentes tipos de sinais vitais (n) de um paciente x. Destacamos que, no estágio atual do trabalho, *Hermes Aggregator* é o único componente da infraestrutura que ainda não possui versão implementada.

[1]Informações adicionais sobre os padrões RDF, OWL e SWRL da Web Semântica podem ser obtidas em [6].

6.2 Validação de Hermes

Para demonstrar a utilização e orquestração dos componentes da arquitetura *Hermes*, foi realizado um estudo de caso em monitoramento de sinais vitais humanos. Neste cenário, são relevantes as informações de contexto (sinais vitais humanos de pacientes), obtidas através de sensores, e as restrições temporais impostas pelo quadro clínico de cada paciente e as anormalidades relacionadas a cada sinal vital.

O objetivo do estudo de caso em questão é avaliar e validar os serviços que a infraestrutura *Hermes* se propõe a oferecer. Para tal, tem-se utilizado a ontologia *Monitoramento de Sinais Vitais Humanos* (MSVH) [1], que modela o histórico de medições de sinais vitais de pacientes, bem como a detecção de alarmes associados a anormalidades nesses sinais.

Como descrito, o componente *Hermes Widget* adquire os dados de baixo nível através de sensores e realiza a sua modelagem semântica, de acordo com uma ou mais ontologias específicas para o domínio, escritas na linguagem OWL. Também são utilizadas regras na linguagem SWRL, que estendem a semântica das ontologias e descrevem, neste caso, as anormalidades nas medições dos sinais vitais, e são armazenadas no mesmo arquivo da ontologia MSVH.

Dada a dificuldade de se obter dados reais de sinais vitais, foram criadas instâncias com base na ontologia MSVH, simulando os dados obtidos de sensores e demais dados relacionados a pacientes. Estes indivíduos contê as entidades envolvidas no monitoramento, como dados de pacientes e de profissionais de saúde, local em que paciente se encontram, horário e medições dos sinais vitais monitorados.

Os indivíduos com as instâncias do monitoramento de sinais vitais são processados pelas classes do componente *Hermes Widget*, que realizam as seguintes etapas: 1) Registro de tópicos referentes a cada sinal vital; e 2) Registro do *Hermes Widget* como publicador dos tópicos, especificando os parâmetros de *QoS* que ele atenderá para essa publicação.

A cada sinal vital coletado pelos *Hermes Widgets*, neste caso simulados pelos indivíduos criados, as seguintes etapas de processamento ocorrem: 1) associação entre o sinal vital e o paciente monitorado no formato RDF, conforme a ontologia MSVH; 2) serialização do objeto Jena que representa as informações de contexto codificadas em RDF; 3) realização de inferência sobre o modelo, para detectar possíveis anormalidades; e 4) solicitação ao *Hermes Base* para publicação do objeto serializado nos tópicos interessados. A Figura 2 apresenta as etapas que o componente *Hermes Widget* realiza em conjunto com o componente *Hermes Base* para a entrega de informações de contexto às aplicações interessadas.

A utilização do *Hermes Base* e, de toda a implementação de comunicação da infraestrutura, é transparente para desenvolvedores, que utilizam apenas as interfaces do componente *Hermes Widget* para registrarem assinaturas para informações de interesse, ou consultas a dados de contexto armazenados em um repositório TDB local. Quando há novas publicações em um *Hermes Widget*, este verifica sua lista de assinantes e utiliza o *Hermes Base* para notificar aplicações, ou mesmo outros componentes da infraestrutura.

7. TRABALHOS RELACIONADOS

O trabalho de [8] apresenta um arquitetura distribuída para aplicações móveis sensíveis a contexto. A solução *Execution Environment for Highly Distributed Applications - Ubiquitous Context awareness* (EXEHDA-UC) apoia o geren-

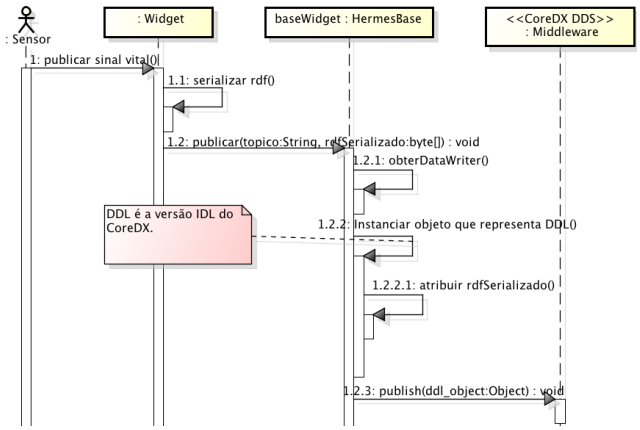

Figure 2: Publicação de sinal vital em um tópico pelo *Hermes Widget*.

ciamento da aquisição, modelagem baseada em regras e armazenamento de contexto, em ambientes distribuídos, independente de aplicação. Os componentes dessa arquitetura são *Border Server*, responsável pela interação com o ambiente através de sensores e atuadores, e *Context Server*, responsável pelo processamento da informação contextual.

A infraestrutura *Hermes* apresenta vantagens em relação ao EXEHDA-UC: a modelagem semântica do contexto e o suporte a parâmetros de *QoS* para a comunicação em tempo real ou em baixos níveis de latência.

Como soluções que realizam modelagem semântica do contexto, são destacadas a *Awareness and Notification Service* (ANS) [5], a arquitetura *Priamos* [7] e o arcabouço *SmartAndroid* [4]. As soluções ANS e *SmartAndroid* apesar de realizarem a modelagem do contexto baseada em ontologia, não suportam a comunicação em tempo real nem requisitos de *QoS*, sendo que esta última possui acoplamento entre a lógica de negócio e o processamento da informação, uma vez que esta lógica está codificada na estrutura de classes do próprio arcabouço. O *Priamus* estabelece que a comunicação em tempo real seja realizada via *Web Services*, o que denota um *overhead* na comunicação.

A infraestrutura *Hermes* oferece uma solução distribuída para a aquisição de informações de contexto, modelando semanticamente esta informação, para prover expressividade e interoperabilidade entre os componentes e aplicações que a utilizam, além de apoiar a comunicação em tempo real e a utilização de parâmetros de *QoS*, ambos aspectos baseados em padrões de indústria, como RDF, OWL e DDS.

8. CONSIDERAÇÕES FINAIS

A utilização do *Hermes* visa minimizar o trabalho de desenvolvedores de aplicações sensíveis a contexto, oferecendo componentes reutilizáveis que realizam cada um dos serviços de contexto apresentados. O modelo arquitetural do *Hermes* possui alta coesão e baixo acoplamento, permitindo que seus componentes possam ser utilizados sob demanda, de forma conjunta ou individual, conforme a necessidade dos desenvolvedores e aplicações, de forma flexível.

Hermes realiza a modelagem e interoperabilidade semântica do contexto, o que confere a separação entre conceitos e regras de negócio da implementação tanto da infraestrutura quanto das aplicações, que utilizam ontologias para esta finalidade. Dessa forma, para que *Hermes* ofereça suporte a um determinado domínio, basta que este conheça a ontologia que modela o conhecimento relacionado, tornando a infraestrutura genérica e independente de aplicação.

Como trabalhos futuros, têm-se a utilização de uma base de dados de referência que agrega sinais vitais humanos reais e o desenvolvimento de uma interface para captação direta de dados de sensores, para o componente *Hermes Widget* que, neste momento, utiliza dados simulados para validação apresentada no estudo de caso.

9. ACKNOWLEDGMENTS

Apoio financeiro do CNPq ao projeto n. 481402/2011-0.

10. REFERÊNCIAS

[1] A. B. Bastos, I. G. Sene Júnior, and R. F. Bulcão Neto. Modelagem e inferência baseadas na semântica de monitoramento de sinais vitais humanos. In *XX Simpósio Brasileiro de Sistemas Multimídia e Web*, pages 1–4, 2014.

[2] A. Bikakis, T. Patkos, G. Antoniou, and D. Plexousakis. A survey of semantics-based approaches for context reasoning in ambient intelligence. In *Constructing Ambient Intelligence*, volume 11 of *Communications in Computer and Information Science*, pages 14–23. Springer, 2008.

[3] A. K. Dey, G. D. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-computer interaction*, 16(2):97–166, 2001.

[4] M. Erthal, D. Mareli, D. B. Ferreira, and O. Loques. Interpretação de contexto em ambientes inteligentes. In *Simpósio Brasileiro de Computação Ubíqua e Pervasiva*, pages 1–10, 2013.

[5] R. Etter, P. Costa, and T. Broens. A rule-based approach towards context-aware user notification services. In *ACS/IEEE International Conference on Pervasive Services*, pages 281–284, 2006.

[6] J. Hebeler, M. Fisher, R. Blace, and A. Perez-Lopez. *Semantic Web Programming*. 2009. 652 pages.

[7] N. Konstantinou, E. Solidakis, A. Zafeiropoulos, P. Stathopoulos, and N. Mitrou. A context-aware middleware for real-time semantic enrichment of distributed multimedia metadata. *Multimedia Tools and Applications*, 46(2-3):425–461, 2010.

[8] J. Lopes, M. Gusmão, R. Souza, P. Davet, A. Souza, C. Costa, J. Barbosa, A. Pernas, A. Yamin, and C. Geyer. Towards a distributed architecture for context-aware mobile applications in UbiComp. In *XIX Simpósio Brasileiro de Sistemas Multimídia e Web*, pages 43–50, 2013.

[9] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context aware computing for the Internet of Things: A survey. *IEEE Communications Surveys & Tutorials*, 16(1):414–454, 2014.

[10] S. Sehic, F. Li, S. Nastic, and S. Dustdar. A programming model for context-aware applications in large-scale pervasive systems. In *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, pages 142–149, 2012.