Waldo: Serviço para Publicação e Descoberta de Produtores de Dados para *Middleware* de Cidades Inteligentes

Alternative Title: Waldo: Data Producers Registry and Discovery Service for Smart Cities Middleware

Marcelo Iury S. Oliveira
Centro de Informática, UFPE
UFRPE
Recife - PE, Brasil
miso@cin.ufpe.br

Kiev Santos da Gama
Centro de Informática, UFPE
Recife - PE, Brasil
kiev@cin.ufpe.br

Bernadette Farias Lóscio
Centro de Informática, UFPE
Recife - PE, Brasil
bfl@cin.ufpe.br

RESUMO

O conceito de Cidades Inteligentes tem despontado como a combinação do ambiente digital com comunidades reais, tornando mais eficiente a gestão dos espaços urbanos. A maioria das soluções propostas está voltada para a proposição de *middleware* de interconexão de dispositivos, coleta e mediação de dados, carecendo de mecanismos eficientes para a publicação e descoberta de produtores de dados. Este artigo apresenta o Waldo, um serviço de publicação e descoberta de produtores de dados voltado para o contexto de Cidades Inteligentes. O projeto do Waldo foi fundamentado no paradigma de Computação Orientada a Serviços, em soluções de bancos de dados NoSQL e na especificação SensorML. Através de simulações pode-se verificar a viabilidade do Waldo como solução para a publicação e descoberta de produtores heterogêneos.

Palavras-chave

Cidades inteligentes, serviço de descoberta, orientação a serviço, banco de dados NoSQL.

ABSTRACT

The smart cities concept has emerged from the combination of digital environments with real communities, thus making more efficient the urban spaces management. The majority of proposed solutions are focused on the proposition of middleware that interconnects devices, collects and mediates data. However, these solutions lack of effective mechanisms for publication and discovery of data producers. This paper presents the Waldo, a service for registry and discovery of data producers in the context of smart cities. Waldo follows the Service-Oriented Computing paradigm and adopts NoSQL database solutions and SensorML specification. Through simulations, we attested the feasibility of Waldo as a solution for publishing and discovery of heterogeneous data producers.

Categories and Subject Descriptors

H.2.4 [Database Management] - Systems - Parallel databases, Query processing, Distributed databases. H.4.m [Information Systems]: Miscellaneous

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SBSI 2015, May 26–29, 2015, Goiânia, Goiás, Brazil.
Copyright SBC 2015.

General Terms

Algorithms, Management, Standardization, Design

Keywords

Smart Cities, discovery service, service oriented computing, NoSQL databases systems

1. INTRODUÇÃO

A evolução das tecnologias de informação e comunicação (TICs) permitiu o desenvolvimento de novas formas de geração, aquisição e processamento de dados. Seguindo esta tendência, cidades também estão fazendo uso das TICs para tornar mais eficiente a gestão dos seus espaços urbanos, com o intuito de melhorar a qualidade de vida de seus habitantes, assim como aperfeiçoar o uso dos equipamentos e recursos públicos, como também preservar o meio ambiente. Este novo modelo de construção e gestão dos espaços urbanos é chamado de Cidades Inteligentes (do inglês, *Smart Cities*) [1, 2].

Soluções de Cidades Inteligentes, além da automatização de subsistemas específicos, a exemplo das aplicações UbiBus [8], Busão [10], PowerMatcher [9], devem buscar estabelecer a integração de dispositivos e sistemas individuais de forma a viabilizar visões mais abrangentes e um contexto integrado da cidade. Assim, um projeto de cidade inteligente deve ser pensado como uma rede de inter-relacionamentos, um sistema interligado, mais orgânico, no qual os relacionamentos são tão importantes quanto às partes [1].

Os sistemas e dispositivos participantes de uma solução de Cidade Inteligente podem atuar tanto como produtores de dados (PDs) quanto consumidores de dados (CD). Os produtores de dados podem ser qualquer tipo de recurso, físico ou virtual, que seja capaz de prover dados para os consumidores de dados. Os dados, neste caso, podem advir de sensores, *smartphones*, até de bases de dados abertos e aplicações da Web.

Diversas soluções têm sido propostas para tratar a integração de produtores e consumidores de dados no contexto de Cidades Inteligentes. Uma parte significativa destas soluções está centrada no uso de um *middleware* de interconexão de dispositivos, coleta e mediação de dados. Este tipo de *middleware* tem o objetivo de viabilizar o desenvolvimento rápido de aplicações urbanas, lidando com diversos tipos de heterogeneidade, incluindo os diferentes níveis de heterogeneidade de dados (estrutural, sintática e semântica) [18].

Apesar de serem eficientes no que tange a integração de dispositivos e mediação dos dados [3], estas soluções carecem de mecanismos eficientes para a publicação e descoberta de produtores de dados. Um serviço de publicação e descoberta deve permitir que produtores sejam

capazes de publicar detalhes sobre seus serviços para que possam ser encontrados por consumidores de dados. E, alternadamente, o serviço de descoberta deve permitir que consumidores sejam capazes de realizar buscas e facilmente descobrir os produtores mais adequados às suas necessidades.. Uma vez descobertos os produtores, o serviço de descoberta deve retornar aos consumidores dados suficientes para que possam avaliar os produtores de forma independentemente, antes de contatá-los ou integrá-los às suas aplicações.

Como a quantidade de potenciais recursos que podem atuar como produtor de dados em uma cidade inteligente é massiva e crescente, selecionar qual o produtor mais apropriado para o provimento de determonados dados se torna um novo desafio a ser resolvido. No mais, o problema da publicação e descoberta requer também a resolução de questões, tais como recursos autônomos e heterogêneos, ausência de estruturação ou padronização e interoperabilidade [4].

Este artigo propõe um serviço de descoberta de produtores de dados, denominado Waldo. O serviço proposto faz uso de um conjunto de especificações abertas, vocabulários de domínio e de banco de dados NoSQL a fim de facilitar a publicação e descoberta de produtores de dados para qualquer *middleware* de Cidades Inteligentes. Este serviço auxiliará o processo de desenvolvimento de novas aplicações construídas a partir de dados urbanos, visto que facilitará a descoberta de produtores de dados que possam atuar como fontes de dados para consumidores de dados. A viabilidade da solução proposta foi validada através de uma prova de conceito que consistiu da implementação de um protótipo da solução e avaliação do protótipo por meio da simulação de um ambiente real inspirado na cidade do Rio de Janeiro.

A organização deste documento segue da seguinte forma: após esta seção introdutória, na qual é feita uma explanação geral sobre a problemática que motiva a realização deste trabalho. Na Seção 2, é apresentado o método de pesquisa. Na Seção 3, são apresentados os conceitos que caracterizam a solução proposta, assim como, sua arquitetura e projeto. A avaliação do mecanismo proposto é apresentada na Seção 4 e, na Seção 5, são apresentados alguns trabalhos relacionados. A Seção 6 conclui o trabalho, com um breve sumário das contribuições e apontando possíveis trabalhos futuros.

2. MÉTODO DE PESQUISA

O presente trabalho foi guiado pela postura filosófica positivista que definiu as estratégias de investigação e métodos de pesquisa. O pensamento positivista é orientado pela busca da construção do conhecimento a partir de inferências lógicas de um fato observado por meio de uma visão determinista, racional e cartesiana. A escolha do positivismo se deve ao fato da área de pesquisa Cidades Inteligentes ainda estar em estágios iniciais, fazendo com que o reducionismo permita uma análise mais apropriada de um contexto complexo que ainda carece de teorias e conceitos amplamente aceitos.

Sob o ponto de vista de seus objetivos, este trabalho tem caráter exploratório, na medida em que busca proporcionar maior entendimento sobre um problema [11]. Especificamente, este trabalho busca investigar, projetar e desenvolver um serviço que permita a publicação de produtores heterogêneos e autônomos, assim como a descoberta dos mesmos por parte de consumidores de dados também heterogêneos e autônomos. Diante do exposto, o presente trabalho possui a seguinte pergunta de pesquisa.

> *A abordagem de registro e descoberta de produtores de dados proposta para o Waldo é uma solução viável para o contexto de Cidades Inteligentes?*

Segundo Yin, a escolha do método de pesquisa está relacionada com aspectos relativos ao tipo de investigação que se pretende executar [11]. Neste sentindo, devem ser considerados fatores, tais como tipo de questão de pesquisa e o nível de controle sobre a unidade de pesquisa e suas variáveis [11]. A pergunta de pesquisa proposta possui caráter causal. Este tipo de pergunta, usualmente, é respondida por meio de experimentos. Contudo, no contexto de Cidades Inteligentes, a realização de experimentos usando sistemas e componentes reais é muito complexa de ser viabilizada por causa de fatores como dificuldade de acesso a sistemas reais em funcionamento, ou simplesmente ausência de tais sistemas.

Sendo assim, neste trabalho, foi realizada uma prova de conceito. Provas de conceitos consistem da implementação de protótipos com as funções fundamentais da solução almejada, seguida de uma avaliação a fim de analisar a viabilidade das tecnologias. Além disso, busca-se aferir a complexidade, tempo necessário para o desenvolvimento completo, os custos envolvidos, assim como métricas de qualidade ou desempenho de avaliação para a solução [27]. No caso deste trabalho, a avaliação do protótipo foi realizada por meio de simulação.

3. SOLUÇÃO PROPOSTA

Este trabalho propõe um serviço, intitulado Waldo, para a publicação e descoberta de produtores de dados. O Waldo é um serviço independente de plataformas tecnológicas, podendo ser conectado a qualquer *middleware* de Cidades Inteligentes. Ademais, considerando que o universo de potenciais produtores é massivo, dinâmico e heterogêneo, o Waldo também tem como objetivos não funcionais ser escalável para atender a demanda crescente de produtores de dados; ser robusto para lidar com a dinamicidade dos participantes e ser extensível para lidar com a heterogeneidade.

O projeto do Waldo foi fundamentado nos princípios da Computação Orientada a Serviços (COS), que é um paradigma que determina que as funcionalidades implementadas pelas aplicações devem ser disponibilizadas na forma de serviços [14]. Serviços são módulos que possuem interfaces formalmente definidas que permitem o acesso remoto de seus métodos e a transmissão de dados por meio de trocas de mensagens. As interfaces, assim, abstraem a implementação do serviço. Inclusive, a COS prevê a utilização de um mecanismo de registro e descoberta de serviços que resulta em um tripé bem definido de provedor de serviço, cliente e catálogo.

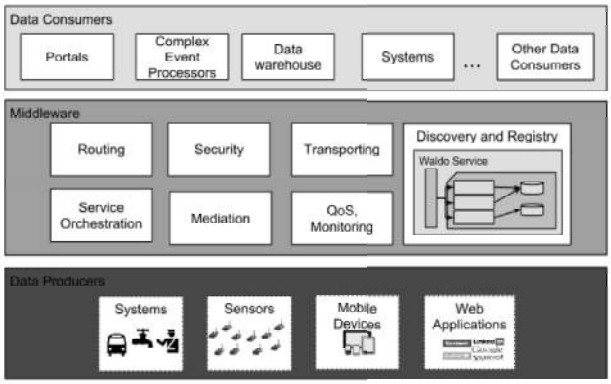

Figura 1. Integração do Waldo com *Middleware*

A abordagem COS traz duas vantagens adicionais. A primeira é permitir que o Waldo seja incorporado a qualquer *middleware* genérico de Cidades Inteligentes, tal qual o apresentado na Figura 1. A incorporação poderá ser realizada sem a necessidade de completa reimplementação de ambas as soluções, *middleware* e Waldo, podendo este último ser acoplado aos módulos de Registro e Descoberta do *middleware*. A segunda vantagem é que o Waldo também pode ser

agregado a outros catálogos de produtores de dados já existentes, permitindo, dessa forma, a troca de dados entre iniciativas de cidades inteligentes distintas, podendo atuar, assim, como um produtor de dados.

3.1 Arquitetura do Serviço Waldo

A arquitetura do serviço Waldo, ilustrada na Figura 2, possui dois catálogos para armazenamentos de metadados e vocabulários, três módulos principais (*i.e. Registry*, *Status Monitor* e *Vocabulary Registry*) e dois componentes auxiliares (*i.e. Adapter* e *Facade*).

Os catálogos são usados para armazenamento de informações sobre todos os PDs. O **PD Catalog** é usado para persistir os metadados dos PDs registrados no Waldo. Por sua vez, o **Vocabulary Catalog** é usado como dicionário de vocabulários usados para descrição dos PDs e de seus dados transmitidos pelos barramentos.

O **Registry** será responsável pelas operações de criação, atualização, remoção e consultas de instâncias de PD. Especificamente, as operações de consulta devem permitir a busca por PD de acordo com parâmetros, tais como fenômeno de interesse, limitação espacial, temporal, ID do sensor entre outros. Há também uma operação de *lookup*, que ao informar o ID (*i.e.* URI) de um PD, será retornado a sua descrição completa.

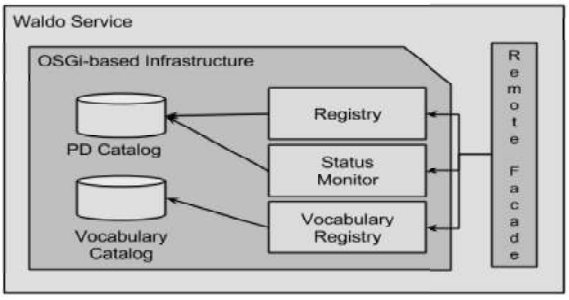

Figura 2. Arquitetura do Waldo

O Registry possui mecanismos de autorização para evitar que um participante não autorizado modifique ou remova um registro de PD. Um participante tem acesso apenas à porção de metadados para a qual possui permissão. O mecanismo de controle de acesso possibilita a manipulação dos metadados estritamente de acordo com a política de autorização especificada para o sistema. São pré-condições para o seu adequado funcionamento que o usuário que solicita acesso tenha sido corretamente autenticado.

O componente **Status Monitor** é o módulo responsável por monitorar a disponibilidade dos PDs. Como os PDs participantes de um barramento possuem comportamento autônomo, ocasionalmente, um PD pode se tornar temporariamente ou permanentemente indisponível. Caso o PD não solicite a remoção do registro, um CD pode, sem sucesso, tentar contatá-lo. Para evitar estes casos, o Status Monitor concede um "período de graça" durante o qual um registro de PD pode permanecer ativo no sistema. Quando expirado este período, o Status Monitor tenta contatar o PD. Caso a comunicação seja mal-sucedida, o Status Monitor marca o PD como *unavailable*. Se um PD marcado como *unavailable* voltar a se conectar no sistema, seu registro será marcado como disponível novamente.

O módulo **Vocabulary Registry** armazena vocabulários semânticos usados na descrição de PDs. A ideia é que usuários utilizem este módulo para descobrir as definições semânticas para os termos, conceitos e fenômenos tratados por um PD. Desta forma, o Vocabulary Registry auxilia tanto o processo de registro quanto de descoberta de PDs. Internamente, o Vocabulary Registry possuirá previamente cadastrado um conjunto de vocabulários mais utilizados no contexto de Cidades Inteligentes. Contudo, novos vocabulários poderão ser cadastrados para suprir a ausência de definições semânticas.

Ainda, o Vocabulary Registry oferece basicamente quatro métodos: três de consulta, e um para registro. Os métodos de consultas simples permitirão encontrar a definição semântica de um de termo com base em uma URI (*Uniform Resource Identifier*) ou em palavras-chaves. O método de registro permitirá o cadastro de novos vocabulários, podendo ser empregado algum método de moderação a fim de evitar inconsistências na base.

Os módulos auxiliares são na verdade implementações de padrões de projeto bem conhecidos e bastante utilizados em diversos sistemas. O componente ***Facade*** tem o objetivo de reunir as funcionalidades do Waldo, disponibilizando uma interface unificada para o conjunto de serviços providos pelos componentes Registry, Status Monitor e Vocabulary Registry. A *Facade* cria uma camada mais alta que facilita o uso dos componentes principais. A *Facade* poderá possuir um ou mais *proxies* remotos para permitir a invocação remota de suas funcionalidades por meio do modelo de comunicação *Request*/*Reply*. Desta forma, tanto o PD quanto o CD somente podem se relacionar com o Waldo através da *Facade* que, ao receber uma requisição, encaminha-as para o módulo responsável por atendê-las.

Por sua vez, o componente ***Adapter*** tem três funções. A primeira é permitir que PDs com interfaces incompatíveis consigam se conectar ao barramento e transmitir seus dados. No mais, os dados também poderão ser estruturados e processados pelo *adapter* para um formato mais apropriado para transmissão no barramento. A segunda função é permitir a agregação de um conjunto de dispositivos ou fontes de dados como um único PD, facilitando, assim, a integração, a exemplo de uma rede de sensores no barramento. A última função do *Adapter* é padronizar a forma de requisição de metadados e informações de um PD.

3.1.1 Relacionamento entre os Módulos

Os componentes principais e os auxiliares se relacionam como ilustrado na Figura 3. Todos os componentes, principais e auxiliares, são encapsulados como serviços. Basicamente, a interação se dá por intermédio de quatro relacionamentos, sendo eles:

1. Os PDs se registram no Waldo ao fazer uma submissão de requisição de cadastro na *Facade*, fornecendo informações relevantes quanto à utilização do mesmo, tais como *endpoint* de acesso, protocolos de transporte suportados, interfaces de interação, formato dos dados de entrada e saída. *Facade* encaminha a requisição para ser processada pelo Registry;
2. Os CDs solicitam ao Waldo a descoberta de PDs através da submissão de requisições de busca por PDs que satisfaçam certos parâmetros, tais como categoria, funcionalidade e características. A *Facade* ao receber a requisição encaminha para o Registry devolve uma lista com PDs que satisfazem às condições estabelecidas na requisição de descoberta;
3. O CD, após escolher um serviço dentre aqueles retornados pelo Registry, inicia o relacionamento com o PD provedor do serviço almejado. Requisições de dados serão enviadas ao respectivo PD por meio de sua interface *Adapter*. O retorno dos dados poderá ser encaminhado diretamente ao CD ou mediado por meio de uma interface de dados usada pelo middleware de cidade inteligente adjacente;
4. O Status Monitor ocasionalmente envia requisições de checagem para PDs que não atualizaram seus registros dentro do "período de graça" concedidos a eles.

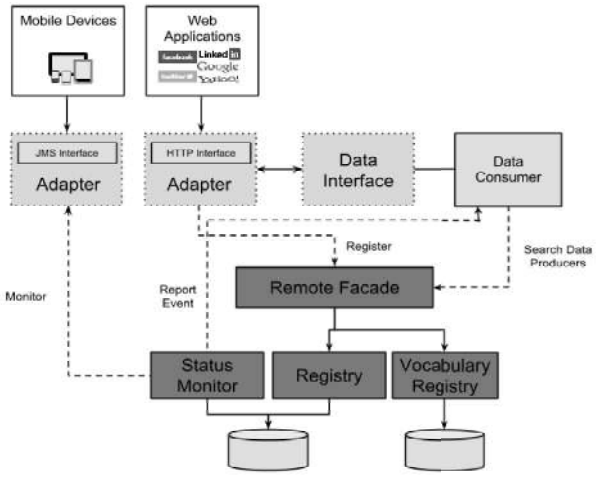

Figura 3. Interação entre os componentes

3.2 Escalabilidade e OSGi

Soluções de Cidades Inteligentes tem escalabilidade e robustez como requisitos inerentes a fim de suportar o aumento de carga devido ao crescimento da quantidade de produtores e consumidores de dados. O encapsulamento das funcionalidades do Waldo como serviços contribui para atender os requisitos de escalabilidade na medida em que aumenta a manutenibilidade. Contudo, essa abordagem não necessariamente permite atualização ou replicação de serviços de forma dinâmica. Assim, é necessário o uso de uma plataforma na qual serviços possam ser adicionados, atualizados e removidos em um ambiente de forma dinâmica em tempo real.

A solução proposta para o Waldo consiste na utilização de OSGi [16], uma plataforma que suporta tanto o desenvolvimento de aplicações a partir da composição de módulos reutilizáveis quanto o carregamento dinâmico de aplicações em tempo de execução. Os módulos possuem informações sobre os serviços que fornecem e sobre suas dependências que devem ser instanciadas ou importadas para o correto funcionamento.

Através do OSGI, novas instâncias (*i.e.* réplicas) dos módulos do Waldo podem ser disponibilizadas sem interrupções e em tempo de execução para lidar com a variação da carga. O provimento destas novas instâncias pode ser realizado pela alocação de novas máquinas sejam elas físicas ou virtuais. Ambas as abordagens permitem a chamada escalabilidade horizontal.

Desta forma, a *Facade* passa a incorporar uma fila de requisições e um conjunto de *threads* despachantes. Todas as requisições recebidas pela *Facade* são armazenadas em uma fila que é consumida pelas *threads* despachantes que, por sua vez, encaminharão as requisições para uma das instâncias disponíveis dos módulos principais. Por sua vez, este encaminhamento é também realizado pela invocação remota dos módulos.

3.3 Metadados de Descrição dos Produtores de Dados

No que tange a heterogeneidade, a arquitetura proposta deve ser flexível para permitir tanto o registro de produtores heterogêneos de dados, quanto o atendimento as requisições de busca oriundas de consumidores de dados também heterogêneos.

O Waldo faz uso do padrão *Sensor Web Enablement* (SWE) do grupo OGC (*Open Geographic Consortium*) [5] para descrever um PD, bem como o tipo de dado que produz. Apesar de haver outras estratégias propostas para tratar da heterogeneidade e interoperabilidade entre dados de sensores e qualquer tipo de aplicação, o SWE é uma das estratégias mais robustas e aceitas pela comunidade científica [13]. Vale ressaltar que o SWE não é um sistema, mas sim uma arquitetura que define padrões para formatos de dados, metadados e interfaces de serviços.

Especificamente, o Waldo utiliza a especificação *Sensor Model Language* (SensorML) para modelar um PD. Além disso, a especificação *Observations & Measurements* (O&M) será usada como um dos vocabulários para definição dos atributos de descrição de um registro de PD, a exemplo da descrição de parâmetros, outputs, localização, identificação, entre outros. Os outros padrões do SWE (*e.g.* SOS, SAS) definem as interfaces de componentes essenciais para o gerenciamento de sensores que não dizem respeito ao cerne do Waldo, não sendo, desta forma, usados por ele.

O Waldo não faz uso da especificação completa de SensorML na modelagem de um PD. O SensorML define diversos elementos/entidades, tais como *PhysicalComponent*, *PhysicalSystem*, *ProcessChain*, *Detector*, *Sensor* e outros. Porém, apenas o elemento *PhysicalComponent* é usado pelo Waldo. Esta entidade foi escolhida, pois permite a descrição de atributos de identificação, caracterização, funcionalidades, outputs, parâmetros e serviços. Estes atributos foram considerados suficientes para permitir a descoberta de PDs. Apesar de o SensorML representar os dados por meio de XML, o Waldo permite o registro, atualização e busca de PDs recebendo como parâmetros tanto objetos Java quanto dados em XML ou JSON.

4. PROVA DE CONCEITO

Nesta seção, serão apresentados detalhes sobre o processo de avaliação da solução proposta, incluindo apresentação das tecnologias utilizadas para o desenvolvimento do protótipo e a descrição do ambiente de simulação utilizado na avaliação.

4.1 Protótipo e Tecnologias Utilizadas

Para validar a solução proposta, foi desenvolvido um protótipo em linguagem Java, tendo sido escolhida por apresentar facilidades para a implementação do sistema. Dentre as facilidades providas, podem-se destacar: alta portabilidade do código-fonte, existência de suporte para comunicação através da rede, bibliotecas nativas para acesso a banco de dados e suporte a programação concorrente.

No mais, optou-se por usar o *framework* Apache CxF [15] para o desenvolvimento dos *frontends* remotos do Waldo. O Apache CxF é um *middleware* de código aberto que ajuda no desenvolvimento de serviços usando APIs de programação remota. O Apache CxF facilita o processo de integração do Waldo com os *middlewares* existentes de Cidades Inteligentes na medida em que oferece suporte para diversos tipos de protocolos como SOAP, XML/HTTP, HTTP RESTful, ou CORBA.

Utilizou-se a solução de banco de dados NoSQL MongoDB para a implementação do PD Catalog. O MongoDB apresenta características propícias ao contexto de Cidades Inteligentes tais como alta disponibilidade, menor tempo de resposta, suporte a paralelismo e flexibilidade de esquema [19]. Em especial, o MongoDB oferece suporte nativo à replicação automática e distribuição horizontal, permitindo, assim, atender mais facilmente requisitos de escalabilidade. Além disso, a não obrigatoriedade de utilização de um esquema comum para todos os registros, permite tanto a extensão e atualização dos esquemas de representação dos PDs, quanto o suporte a registros esparsos nos quais alguns campos não são preenchidos.

Conjuntamente com o MongoDB, foi utilizado o framework Apache JCS [24] para implementação do sistema de *cache* de consultas. JCS é um sistema de *cache* distribuído escrito em Java. Foi especialmente

construído para acelerar aplicações, fornecendo um meio para gerenciar dados em *cache* de diversas naturezas dinâmicas.

Foi utilizada a biblioteca SWE *Common Library* [20] como base para a implementação das funções de codificação, decodificação e validação dos registros dos PDs descritos em SensorML. Em virtude de o SensorML ser especificado em XML e o MongoDB utilizar um formato derivado do JSON, foi necessária a implementação de *parsers* que permitissem a conversão entre os dois formatos. A propósito, empregou-se a convenção *Badgerfish* [21] na tradução de XML e JSON. Essa convenção estabelece um conjunto de regras que, sobretudo, lida com os aspectos de conversão de atributos, elementos vazios e a ordem de disposição das informações.

4.2 Avaliação

O processo de avaliação teve como principal objetivo responder a pergunta de pesquisa apresentada na Seção 2. Porém, é importante notar que a análise de viabilidade envolve a avaliação de um conjunto de parâmetros quantitativos e qualitativos, a exemplo de custo-benefício, manunibilidade e segurança. Embora existam diversos fatores, no contexto de Cidades Inteligentes, a heterogeneidade e o volume massivo de produtores e consumidores fazem com que robustez e flexibilidade sejam requisitos essenciais para o atendimento da viabilidade de uma solução de registro e descoberta.

A robustez diz respeito a capacidade de atendimento a grandes volumes de requisições, sejam elas de registro ou descoberta, com os devidos critérios de desempenho e disponibilidade. Por sua vez, flexibilidade diz respeito a capacidade de suportar a ausência de padronização entre os dispositivos, seja em relação as suas características, aos serviços ofertados, aos protolocos de comunicação ou aos modelos de dados usados. Desta forma, esses requisitos derivaram as seguintes perguntas de pesquisa secundárias para avaliação da viabilidade:

1. Robustez:
    a) O Waldo é capaz de manipular requisições frequentes de cadastro e descoberta de PDs?
2. Flexibilidade
    a) O Waldo suporta o registro de PDs heterogêneos?
    b) O Waldo suporta a incorporação ou remoção de atributos de um PD?

Uma vez definidas as perguntas de pesquisa, é preciso projetar os experimentos que serão usados para respondê-las. Contudo, um problema que surge ao se projetar experimentos para soluções de Cidades Inteligentes é a complexidade em controlar e viabilizar as variáveis e unidades experimentais. A realização de experimentos usando sistemas e componentes reais pode ser impossível ou impraticável por causa do alto custo de prototipagem e testes, ou por causa da duração do experimento em tempo real ser impraticável.

A alternativa usada neste trabalho foi o emprego de simulação como um meio para a realização de experimentos. A simulação permite que se modele um sistema próximo do real, aumentando a confiabilidade nos resultados e nas decisões [5]. Desta forma, visando validar a solução proposta, foi desenvolvida uma aplicação que faz uso de *mocks* para simular a interação de PDs e CDs com o Waldo. Os *mocks* são objetos pré-programados que simulam partes do comportamento de um sistema real e são capazes de verificar se o sistema está funcionando conforme especificado.

Por fim, a avaliação do Waldo foi realizada utilizando-se o protótipo apresentado na Seção 4.1. As perguntas de pesquisa em estudo envolvem essencialmente a avaliação dos módulos *Facade*, Registry, PD Catalog e *Adapter*. Os outros módulos, apesar de relevantes, não foram considerados na avaliação pelo fato de possuírem funcionalidades secundárias no Waldo.

4.2.1 Modelo de Simulação

O ambiente de simulação foi composto por um conjunto de *mocks* PDs e CDs que, de forma programada, submetiam requisições de cadastro, atualização e consulta para o Waldo. No que tange o lado servidor da solução, o simulador dispunha de um conjunto de *threads* responsáveis por tratar as requisições. Cada *thread* executava uma instância do Waldo.

A simulação era executada em ciclos, na qual um determinado conjunto de CDs ou PDs escalonava requisições que eram submetidas individualmente em um tempo aleatório máximo de 1 segundo. A quantidade de ciclos executados por cada conjunto era determinada pela quantidade de requisições a serem submetidas por cada entidade. Por exemplo, 10.000 consumidores enviando 100 consultas equivaliam a 100 ciclos. Além disso, não havia sincronização entre os ciclos, assim podendo existir consultas oriundas de diferentes conjuntos sendo atendidas simultaneamente pelo Waldo.

Como o objetivo da simulação era verificar como o sistema se comportava frente ao recebimento intermitente de requisições, independentemente do tempo de transmissão das mesmas, foram abstraídos aspectos relativos à comunição, tais como topologia de rede, congestionamento e variação de tempo de transmissão. Desta forma, todas as entidades participantes da simulação executaram em uma única máquina, estando plenamente disponíveis e podendo se comunicar entre si sem a ocorrência de falhas.

O resultado obtido na simulação é composto por dois valores: o tempo total de execução e a vazão de atendimento de consultas. O tempo total é medido após a instanciação do *testbed*, até o momento do atendimento de todos os ciclos de requisição.

4.2.2 Cenários Simulados

Nas simulações, foram usados tipos distintos de CDs e PDs. Estas entidades são comumente usadas na produção de sistemas para Cidades Inteligentes [1]. A quantidade de PDs, conforme apresentado no Quadro 1, foi determinada com base em estimativas relacionadas a cidade do Rio de Janeiro, tendo sido escolhida como referência tanto pela dimensão da cidade, que é segunda maior cidade do Brasil, quanto pela significativa disponibilidade de bases de dados abertas sobre o Rio de Janeiro[1].

Há um conjunto de CDs para cada tipo de PD usado no *testbed*. Isto é, existe um conjunto de CD que buscam apenas Táxis, outros que buscam apenas ônibus e assim por diante. Todos CDs submetem requisições de descoberta de PDs que possuam um determinado valor, entre 8 valores possíveis, para o atributo WALDO_TESTE. Este atributo pertence ao grupo de atributos de características presente em todos os PDs. Desta forma, as requisições de descoberta submetidas na simulação recebiam como parâmetros o tipo de entidade a ser buscada e um atributo de característica a ser usado como predicado de filtragem. No total, foram geradas 80 consultas distintas.

A quantidade de tipos distintos de CDs foi proporcional a quantidade total de PDs e CDs disponíveis. Por exemplo, considerando um total de 10 mil CDs e 100 mil PDs, dos quais 34.230 (34,23%) representam PDs de Táxis, houve 3.423 CDs efetuando buscas relativas a taxis. Isto é, dos 10 mil consumidores, 34,23% deles efetuaram buscas sobre táxis, seguindo a mesma lógica, 900 (9%) e 160 (1,6%) efetuaram de ônibus e câmeras de tráfego, respectivamente.

[1] http://data.rio.rj.gov.br/

As quantidades totais de consumidores utilizadas foram 10.000 e 25.000. Estes valores foram definidos com base na quantidade média de usuários ativos do Waze em grandes cidades da Europa [24]. No mais, as simulações ocorriam com um percentual dos PDs já cadastrados e, ao longo da simulação, os PDs não registrados solicitavam o cadastro de seus metadados.

Quadro 1. Sumário de Produtores de Dados Usados na Avaliação

Tipo de Produtores	Atributos do Produtor	Quantidade de PDs
Táxi	Identificação, Localização, Características e Output	34.230
Ônibus	Identificação, Localização, Características e Output	9.000
Estação Meteorológica	Identificação, Localização, Características, Funcionalidades e Output	320
Câmera de Tráfego	Identificação, Localização, Características, Funcionalidades, Input e Output	1.600
Base de Dados Aberta	Identificação, Características, Funcionalidades, Informações de Contato e Output	28.500
Semáforos de Transito	Identificação, Localização, Características, Funcionalidades e Output	5.250
Viaturas de (Polícia e Transito)	Identificação, Localização, Características e Output	2.000
Metrô e Trem	Identificação, Localização, Características e Output	100
Sistemas de Alerta	Identificação, Localização, Características, Funcionalidades e Output	320
Outras Fontes (páginas web e aplicativos sociais)	Identificação, Características, Funcionalidades, Informações de Contato e Output	18.680
Total		**100.000**

De maneira a avaliar a flexibilidade da solução, utilizaram-se esquemas distintos para representação dos PDs que compunham o *testbed*, como pode ser visto na coluna Atributos do Quadro 1. Todos os PDs possuíam em comum a descrição de sua identificação, características e output, podendo apresentar outras descrições tais como localização e capacidades.

Quadro 2. Cenários Simulados

Id. Cenário	Quant. de CDs	Quant. de Ciclos	Política de BD
Cenário 1	10 mil	100	Sem índices
Cenário 2	10 mil	100	Com índices
Cenário 3	10 mil	500	Com índices
Cenário 4	10 mil	100	Com índices, subcoleções
Cenário 5	10 mil	500	Com índices, subcoleções
Cenário 6	10 mil	100	Com índices, subcoleções e *cache*
Cenário 6	10 mil	500	Com índices, subcoleções e *cache*
Cenário 7	25 mil	100	Com índices, subcoleções e *cache*
Cenário 8	25 mil	500	Com índices, subcoleções e *cache*

Visto que a solução de banco de dados tem um impacto significativo para o desempenho do Waldo, foram avaliadas diferentes configurações do MongoDB com o objetivo definir os limites do sistema de acordo com a máquina disponível e como o sistema responde a carga de trabalho. As abordagens utilizadas foram: criação de índices, uso de subcoleções e de resultados.

Em todos os cenários executados, foram considerados um percentual de 50% dos PDs previamente cadastrados e uma limitação de 100 registros retornados por consultas. Esta limitação é uma prática comum em diversos sistemas da Web [23].

Além dos parâmetros já citados, que definem uma configuração base para a nossa avaliação, definimos diferentes cenários por meio da variação de dois outros parâmetros do modelo de simulação: quantidade de CDs e quantidade de ciclos de requisições. O Quadro 2 apresenta os cenários executados.

Todos os experimentos foram executados em um notebook Asus S46C, - Processador Intel Core i5-3317U 1.7GHz, Memória RAM 6GB e Sistema Operacional Windows 8.1 64 bits. Cada experimento executou um mínimo de 4 rodadas para cada cenário executado. Para grande maioria dos cenários, as 4 rodadas foram suficientes para obtermos resultados com 95% de nível de confiança e erro máximo de 5%.

4.2.3 Resultados

Durante a realização dos experimentos, o Waldo executou uma carga de trabalho entre 1 milhão a aproximadamente 12 milhões de consultas. Em relação ao consumo de memória, o Waldo manteve um consumo de memória estável em torno de 2 GB. A memória RAM é um dos principais pontos de análise em um mecanismo de descoberta, pois está diretamente ligada ao número de requisições que o servidor é capaz de atender em determinado tempo. Além disso, todos os 100 mil PDs do *testbed* ocuparam apenas 300MB de dados, possuindo uma média de 4 KB por registro. Em virtude de tratar-se também de um sistema de registro de dados, o baixo consumo do Waldo para o armazenamento dos registros torna viável manipular uma quantidade maior de produtores de dados pertencentes a uma cidade inteligente.

No mais, apesar de consumir toda capacidade de processamento da máquina de teste, a aplicação não apresentou problemas de concorrência durante os experimentos. Contudo, a duração e a vazão dos experimentos variaram de acordo com os cenários simulados. As Figuras 4 e 5 mostram os resultados obtidos com a simulação.

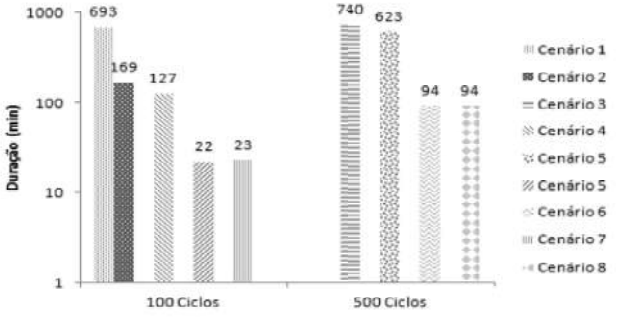

Figura 4. Duração da Execução

O Eixo y apresenta as latências e as vazões obtidas. Um resultado esperado que pode ser observado na Figura 4 é que, para a maioria dos cenários, a duração para o atendimento de todas as requisições aumenta à medida que as quantidades de consumidores e de ciclos de consultas também cresce. Entretanto, esse aumento da duração do experimento não está atrelado a uma melhoria na vazão do sistema. Pode-se perceber isso nos cenários 2 e 3, nos quais há apenas uma diferença de aproximadamente 9% (10 requisições), mesmo existindo aumento de 100 para 500 ciclos. Esse mesmo padrão pode ser percebido nos cenários 4 e 5.

Outro resultado que se destacou foi o ganho de eficiência com uso de *cache* para armazenamento das consultas, que apresentou resultados significativamente superiores ao restante das abordagens. Pode-se perceber isso fazendo uma análise dos cenários 5 ao 8. Nestes cenários,

o tempo máximo de duração foi de 23 e 94 minutos para, respectivamente, 100 e 500 ciclos. A mesma tendência pode ser observada na Figura 5. A menor vazão obtida com o uso de índices foi de 803 requisições/segundo. Esse resultado é aproximadamente 481% superior que a vazão obtida no cenário 4, que é o melhor dentre os quais não há uso de *cache*.

Analisando apenas os cenários nos quais há uso de *cache*, destaca-se outro resultado. Os cenários 7 e 8, apesar de possuírem uma carga de trabalho maior (25 mil consumidores em contraposição aos 10 mil consumidores utilizados nos outros cenários), apresentaram os melhores resultados em termos de duração e vazão, sendo de 1873 e 2223 requisições atendidas por segundo para 100 e 500 ciclos, respectivamente. Este resultado é explicado por dois fatores: pouca variação nas consultas submetidas e maior quantidade de consultas submetidas. O aumento do número de consumidores implica em mais consultas submetidas simultaneamente. Como o conjunto de tipos distintos de consultas é pequeno (*i.e.* 80 consultas), há um aumento da quantidade de *cache hit*, reduzindo a necessidade de busca em disco rígido e consequentemente diminuindo o tempo médio para atendimento das consultas.

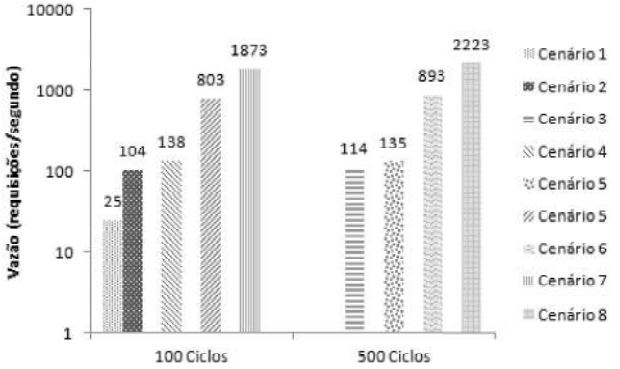

Figura 5. Vazão de Atendimento de Requisições

Aliado ao *cache*, a criação de índices se mostrou muito importante para alcançar um bom desempenho. Isso pode ser mais facilmente percebido no Cenário 1, que não fez uso de índices. Este cenário apresentou tempo de duração similar aos dos cenários 3, 5 e 7 que utilizaram índices e possuíam carga de trabalho 5 vezes superior ao Cenário 1. Por sinal, o uso de índices se torna mais determinante na medida em que a quantidade registros armazenados pelo Waldo cresce. Segundo os comandos de *profile* do MongoDB, as consultas executadas sem o suporte dos índices efetuam uma leitura completa dos registros existentes no sistema.

No que tange a avaliação da eficiência no tratamento de requisições de cadastro, a solução também apresentou bom desempenho. Durante a fase de preparação do *testbed,* que efetuava o registro de 50 mil PDs, também foi medida a vazão para o atendimento. O sistema apresentou uma vazão média de aproximadamente 4.800 requisições atendidas por segundo, tendo sido usadas 50 threads.

Em relação à avaliação da flexibilidade, o Waldo conseguiu lidar com o registro e busca de PDs heterogêneos. O SensorML, por ser baseado em XML e por oferecer um esquema plenamente extensível, permitiu a variação do conjunto de atributos de descrição, o preenchimento parcial de metadados, bem como, a adição e remoção de atributos. O MongoDB também contribuiu para o atendimento do requisito de flexibilidade ao suportar o registro de documentos heterogêneos (*i.e.* documentos com esquemas distintos) em uma mesma coleção.

No mais, em relação ao desempenho, não houve significativa diferença entre os tipos distintos de PDs registrados no sistema, embora possuíssem esquemas de dados também distintos. Contudo, é preciso ressaltar que a estrutura de modelagem complexa do SensorML, tornou também complexa a criação do indíces para otimização das consultas.

5. TRABALHOS RELACIONADOS

Algumas implementações do SWE foram propostas, dentre as mais robustas destacam-se os projetos OSIRIS [2] e GENESIS[3]. No entanto, estes projetos são restritos ao domínio de dados geoespaciais e a persistência é implementada com SGBD relacional. Dessa forma, além de não ser facilmente generalizável para outros tipos de aplicação, ainda há o problema de escalabilidade e rigidez no esquema de dados.

Um dos trabalhos mais recentes fundamentado no SWE foi desenvolvido por Fazio e colegas [6] e teve como objetivo propor uma arquitetura que permita o acesso a qualquer tipo de dados, obtido a partir de diferentes sites de sensoriamento. Esta arquitetura utiliza um conjunto de linguagens baseadas em XML e especificações de interface de serviços Web (a exemplo, de REST) com o intuito de facilitar a descoberta, acesso e pesquisa sobre os dados dos sensores. Entretanto, a solução apresentada nesse trabalho parte do princípio de que os sites administrativos dos sensores serão responsáveis pela publicação e configuração dos dados. Além disso, o trabalho assume que os problemas de integração serão resolvidos na medida em que a estrutura e semântica dos dados esteja construída.

Tendo como ponto de partida a especificação do SWE, o W3C (*World Wide Web Consortium*) criou um grupo chamado *Semantic Sensor Networks Incubator Group* (SSN-XG) cuja tarefa foi desenvolver o *Semantic Sensor Network* (SSN) [7], que é uma ontologia criada para modelar sensores, sistemas, processos e observações. Enquanto os padrões especificados pelo SWE apenas proveem descrição para acesso a sensores e seus metadados, eles não oferecem a capacidade de abstração, categorização e raciocínio por meio de tecnologias semânticas [7].

No contexto de COS, há também diversos trabalhos que apresentam soluções de publicação e descoberta de serviços [25], dentre eles destaca-se o WS-Discovery [26] que define protocolos de descoberta através de trocas de mensagem baseadas em IP-Multicast. Apesar de ser bastante usado e de existirem diversas implementações consolidadas, o WS-Discovery não oferece nenhuma especificação ou *guideline* de como descrever um produtor de dados.

Por fim, dentre as propostas voltadas para sensores e internet das coisas, o EPCglobal, que é um conjunto de tecnologias que permite a identificação e o compartilhamento de informações a partir de *tags* RFID também possui um mecanismo de descoberta [28]. Contudo, o mesmo possui restrições de privacidade e alcance estando mais voltado para aplicações do tipo *Machine-to-Machine* [28].

6. CONCLUSÃO E TRABALHOS FUTUROS

Cidades inteligentes são iniciativas que buscam tornar a gestão de seus espaços e recursos mais eficientes por meio do uso de tecnologias de informação e comunicação. Diversas soluções tem sido propostas para viabilizar o desenvolvimento de Cidades Inteligentes com o intuito de estabelecer a integração de dispositivos e sistemas individuais diversos por meio do uso *middleware* de interconexão de dispositivos, coleta e mediação de dados. Contudo, essas soluções carecem de mecanismos eficientes para a publicação e descoberta de produtores de dados.

Este artigo apresentou o Waldo, um serviço de publicação e descoberta de produtores de dados voltado para o contexto de Cidades

[2] http://www.osiris-fp6.eu

[3] http://www.genesys-project.eu/

Inteligentes. O Waldo possui um conjunto de componentes que permite aos consumidores de dados efetuarem a busca de produtores por meio de serviços simples, como busca por identificadores até a busca por produtores que atendam a determinados critérios, tais como tipo de dados fornecido e localidade do dispositivo. Para a avaliação do Waldo foi desenvolvido um protótipo em Java, auxiliado pelos *frameworks* Apache CxF e Apache JCS e por uma solução de banco de dados MongoDB.

As simulações confirmaram que o Waldo é uma solução viável quando usado o sistema de *cache* para armazenamento das consultas. Estes resultados também corroboram as suposições quanto à robustez e flexibilidade da solução no suporte a produtores e consumidores de dados heterogêneos.

É importante ressaltar que os resultados de desempenho apresentados neste trabalho também dependem diretamente da máquina utilizada. Os resultados apresentados neste artigo foram coletados através da execução do sistema em uma máquina doméstica. Certamente, em um servidor ou em um *cluster*, a capacidade do Waldo de atender consultas pode ser melhorada.

Dentre os trabalhos futuros, pode-se incluir a implementação do mecanismo de raciocínio do Vocabulary Registry que permitirar localizar definições semânticas com base em palavras-chave ou definições semelhantes a uma definição usada como parâmetro. Outro desafio é fazer a extensão do SensorML para a incorporação de metadados relativo a contexto [29] e outros aspectos mais voltados para Cidades Inteligentes. Além disso, pretende-se, o estudo de outras soluções de banco de dados NoSQL para o armazenamento dos metadados, assim comoa avaliação comparativa do Waldo em relação a outras soluções de publicação e descoberta de produtores de dados.

7. AGRADECIMENTOS

Este trabalho foi parcialmente suportado pelo Instituto Nacional de Ciência e Tecnologia para Engenharia de Software (INES). Marcelo Iury S. Oliveira é bolsista do CNPq–Brasil na modalidade doutorado.

8. REFERENCIAS

[1] KOMNINOS, Nicos. Intelligent cities: innovation, knowledge systems, and digital spaces. Taylor & Francis, 2002.

[2] SCHAFFERS, Hans et al. Smart cities and the future internet: Towards cooperation frameworks for open innovation. Springer Berlin Heidelberg, 2011.

[3] GAMA, K., TOUSEAU, L., e DONSEZ, D. (2012). Combining heterogeneous service technologies for building an Internet of Things middleware. Computer Communications, 35(4), 405-417.

[4] NAPHADE, Milind et al. Smarter cities and their innovation challenges. Computer, v. 44, n. 6, p. 32-39, 2011.

[5] BOTTS, Mi*ke et al.* OGC® sensor web enablement: Overview and high level architecture. In: GeoSensor networks. Springer Berlin Heidelberg, 2008. p. 175-190.

[6] FAZIO, Maria et al. Heterogeneous sensors become homogeneous things in smart cities. In: Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on. IEEE, 2012. p. 775-780.

[7] LEFORT, Laurent et al. Semantic sensor network xg final report. W3C Incubator Group Report, v. 28, 2011.

[8] VIEIRA, Vaninha et al. The UbiBus project: Using context and ubiquitous computing to build advanced public transportation systems to support bus passengers. In: SBSI, 2012, São Paulo.

[9] KOK, J. Koen; WARMER, Cor J.; KAMPHUIS, I. G. PowerMatcher: multiagent control in the electricity infrastructure. In: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems. ACM, 2005. p. 75-82.

[10] LEITE, Daniel Farias Batista; ROCHA, Júlio Henrique; DE SOUZA BAPTISTA, Cláudio. Busão: um Sistema de Informações Móvel para Auxílio à Mobilidade Urbana Através do Uso de Transporte Coletivo. In: SBSI, 2013, João Pessoa.

[11] YIN, Robert K. Estudo de Caso: Planejamento e Métodos. Bookman editora, 2015.

[12] KAMIENSKI, Carlos et al. Colaboraçao na internet e a tecnologia peer-to-peer. In: XXV Congresso da Sociedade Brasileira de Computação–SBC2005. 2005.

[13] BRÖRING, Arne et al. New generation sensor web enablement. Sensors, v. 11, n. 3, p. 2652-2699, 2011.

[14] PAPAZOGLOU, Mike P. Service-oriented computing: Concepts, characteristics and directions. In: Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on. IEEE, 2003. p. 3-12.

[15] APACHE, C. X. F. (2009). An open source service framework. See: http://cxf. apache. org, 111.

[16] ALLIANCE, T. O. "OSGi Service Platform Core Specification, Version 4.3." (2011).

[17] MongoDB. disponível em http://www.mongodb.org, acessado em 04-03-2015

[18] HASLHOFER, Bernhard; KLAS, Wolfgang. A survey of techniques for achieving metadata interoperability. ACM Computing Surveys (CSUR), v. 42, n. 2, p. 7, 2010.

[19] VIEIRA, Marcos Rodrigues et al. Bancos de Dados NoSQL: conceitos, ferramentas, linguagens e estudos de casos no contexto de Big Data. SBBD. São Paulo, 2012.

[20] Lib-swe-common, disponível em https://github.com/sensiasoft /lib-sensorml, acessado em 04-03-2015

[21] BADGERFISH. disponível em http://badgerfish.ning.com/ acessado em 04-03-2015

[22] WAZE STATS. disponível em http://wazestats.com/ acessado em 04-03-2015

[23] VESDAPUNT, Norases; GARCIA-MOLINA, Hector. Identifying Users in Social Networks with Limited Information. 2014.

[24] Apache JCF. disponível em http://commons.apache.org/proper /commons-jcs/ acessado em 04-03-2015

[25] MUKHOPADHYAY, Debajyoti; CHOUGULE, Archana. A survey on web service discovery approaches. In: Advances in Computer Science, Engineering & Applications. Springer Berlin Heidelberg, 2012. p. 1001-1012.

[26] BEATTY, John. Web services dynamic discovery (ws-discovery). http://msdn. microsoft. com/library/en-us/dnglobspec/html/ws-discovery1004. pdf, 2004.

[27] CHURCHMAN, Charles West; SCHAINBLATT, A. H. The researcher and the manager: A dialectic of implementation. Management Science, v. 11, n. 4, p. B-69-B-87, 1965.

[28] FABIAN, Benjamin; GÜNTHER, Oliver. Security challenges of the EPCglobal network. Communications of the ACM, v. 52, n. 7, p. 121-125, 2009.

[29] GU, Tao; PUNG, Hung Keng; ZHANG, Da Qing. A middleware for building context-aware mobile services. In: Vehicular Technology Conference, 2004. VTC 2004-Spring. 2004 IEEE 59th. IEEE, 2004. p. 2656-2660.