Middleware para Cidades Inteligentes baseado em um Barramento de Serviços

Rafael Borja, Kiev Gama

Centro de Informática – Universidade Federal de Pernambuco (UFPE)
A. Jornalista Aníbal Fernandes, S/N – Cidade Universitária – Recife – PE – Brasil

{rmbg,kiev}@cin.ufpe.br

Abstract. *The growing popularity of communication-enabled devices has led to the eminent emergence of a so-called Internet of Things. In this context, the urban data collection has allowed the emergence of Smart Cities that need to integrate countless systems and devices for urban monitoring generating relevant information. This article proposes an architecture model based on a Service Bus that allows the rapid development of applications using these systems and data.*

Resumo. *A popularização de dispositivos com capacidade de comunicação tem levado ao eminente surgimento de uma chamada Internet das Coisas. Neste contexto, a coleta de dados urbanos tem permitido o surgimento de Cidades Inteligentes que necessitam integrar uma miríade sistemas e dispositivos para o monitoramento urbano, gerando informação relevante. Este artigo propõe um modelo arquitetura baseada em um barramento de serviços que permita o rápido desenvolvimento de aplicações.*

1. Introdução

Recentes avanços tecnológicos em redes de comunicação e de eletrônica digital permitiram o desenvolvimento de sensores de baixo consumo capazes de se comunicar a pequenas distâncias (Akyildiz et al, 2002), levando à definição do termo Internet das Coisas (Internet of Things – IoT) como "uma rede onde objetos que podem ser individualmente e instantaneamente identificados por etiquetas RFID" (Ashton, 2009). Ao longo do tempo, esta definição incluiu dispositivos mais complexos com memória interna e capacidade de processamento como dispositivos NFC (Near Field Communication), GPS, smartphones dentre outros objetos. O uso desta tecnologia vem sendo empregado no monitoramento urbano, viabilizando o conceito de Cidades Inteligentes (Smart Cities), que é uma tendência mundial relativa ao uso de soluções intensivas de tecnologia de informação e comunicação (TIC) como instrumento para tornar cidades mais inteligentes (Hernández-Muñoz et al. 2011).

Uma gestão urbana focada em TIC é baseada na integração de informações provenientes de diversos sistemas complexos, que atendem a domínios específicos. Esta integração é feita com o objetivo de centralizar o monitoramento e controle, além de auxiliar no apoio a decisão. Estes sistemas coletam dados através de sensores ou através de *crowdsensing*, este último uma alternativa para reduzir custos com a utilização de dispositivos pessoais que auxiliam na coleta de dados urbanos (Ganti et al, 2011). Os dados obtidos são disponibilizados tipicamente sem correlacionar fontes de dados de outros sistemas que poderiam ser relevantes para a tomada de decisão naquele contexto.

Tais sistemas devem utilizar mecanismos que permitam que estes dados, sejam analisados através de diversos mecanismos (ex.: mineração de dados, contexto computacional, etc.) que estabeleçam correlações e gerem informações de maior valor

agregado. Neste âmbito, uma miríade de sistemas e dispositivos precisam ser facilmente integrados, lidando com os padrões e formatos de dados heterogêneos. Tais dados, proveniente de fontes diferentes, devem ser reunidos e podem ser mantidos em uma forma escalável de armazenamento com a ajuda de, por exemplo, infraestruturas de Cloud Computing e bases de dados NoSQL. Estes dados podem ser utilizados para análise, seja detectando informações em tempo real ou pela análise posterior por motores de raciocínio, gerando valor agregado (ex.: causalidade entre dados, tendências analíticas).

Este artigo propõe uma plataforma baseada em um barramento de serviços, voltada para facilitar a integração de fontes de sistemas e dispositivos, de modo que aplicações de terceiros possam ser construídas a partir de dados urbanos coletados de tais fontes de dados heterogêneas. A implementação da plataforma proposta foi validada em uma simulação no domínio da mobilidade urbana, utilizando dados reais e processamento de eventos complexos. O restante deste artigo está organizado da seguinte forma: a seção 2 discute trabalhos relacionados; a seção 3 introduz a proposta da plataforma; a seção 4 traz detalhes da implementação e apresenta o caso de estudo que valida a plataforma, e finalmente, a seção 5 traz conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Haller et al. (2009) propõe um mundo onde "smart objects" são integrados a redes e podem ser participantes ativos de processos de negócio. Baseado nesta visão, é proposto expandir os paradigmas de sistemas corporativos utilizados por empresas no gerenciamento e tomada de decisões, especialmente BI (Business Intelligence) e CEP (Complex Event Processing) em aplicações para Smart Cities utilizando Internet das Coisas, uma vez que a simples aquisição de dados, por si só não é capaz de fornecer elementos adequados para tomada de decisão, seja em empresas, seja em grandes cidades. Questões de escalabilidade podem inviabilizar este objetivo através de uma abordagem tradicional enquanto arquiteturas orientadas a eventos (Event Driven Architectures) e CEP devem se tornar grandes ferramentas para gerenciar a avaliação e extração de informações de dados complexos (Haller et al, 2009).

Gama et al. (2012) propôs uma arquitetura para um middleware baseada em computação orientada à serviços com uma arquitetura modular e de baixo acoplamento permitindo a integração de uma miríade heterogênea de dispositivos e tecnologias sem a necessidade de parada ou reconfiguração do sistema. Sua arquitetura permite que novos módulos e dispositivos evoluam ou sejam agregados de maneira independente, entretanto esta ainda não utiliza um ESB (Enterprise Service Bus), o que inviabiliza soluções para orquestração de processos e dificulta o uso de CEP no middleware.

Valente et al. (2011) propôs o framework de middleware MufFIN (Middleware For the Internet of thiNgs) para redes de dispositivos heterogêneos baseado em eventos que abstrai a interação ente aplicações e smart objects utilizando o Fuse Enterprise Service Bus e ActiveMQ como serviço de mensageria (Valente & Martins, 2011). A abordagem proposta vai no sentido proposto por este autor, porém pretende investigar de maneira mais aprofundada as possibilidades advindas desta prática considerando outras plataformas de ESBs aplicadas ao middleware para dispositivos heterogêneos proposto por (Gama et al, 2012).

3. Proposta

Visando o contexto de Cidades Inteligentes, a plataforma proposta também engloba a integração com dados abertos governamentais para criar as mais diversas aplicações para a sociedade. O middleware proposto viabiliza uma extensão desta filosofia, uma filosofia de "dispositivos abertos" como uma extensão destes princípios. Neste cenário teríamos governos compartilhando não apenas dados históricos, mas também informações de dispositivos em tempo real. Seria possível que prefeituras disponibilizassem dados dos sensores de velocidade das vias da cidade, a localização dos ônibus, status de detectores de enchentes, etc., permitindo à comunidade de desenvolvedores a rápida integração destes dados sem que sejam necessários conhecimentos intrínsecos de cada dispositivo nem os demorados trâmites legais para liberação de dados. Para governos, estes fluxos de informação poderiam ser disponibilizados sem a necessidade da manutenção da complexa infraestrutura capaz de suportar uma quantidade virtualmente ilimitada de consumidores.

A utilização de *crowdsensing* abre a possibilidade da construção de aplicativos em tempos e custos antes impossíveis através da participação de fornecedores e consumidores anônimos. Em razão do uso de muitas ideias da Arquitetura Orientada a Serviços (SOA), foi utilizado um ESB para implementar o middleware, valendo-se das capacidades de alta disponibilidade, distribuição e orquestração oferecidas por um ESB. Outra possibilidade para a plataforma é a monetização dos serviços de dados. O usuário poderia adquirir créditos e utilizá-los para assinar um *feed* de dados. Uma alternativa à compra de créditos é a disponibilização por parte de um consumidor de outros *feeds* monetizados, gerando créditos para que o usuário possa consumir outros *feeds*. Um exemplo seria um *feed* de localização de ônibus gerado a partir da agregação dos *feeds* fornecidos por municípios vizinhos em um único *feed* ou um serviço que calcule a velocidade média de vias gerado a partir da análise destes dois *feeds*. O consumo destes dados geraria receita para o agregador que, a depender de preço e utilização do seu *feed*, geraria valores superiores aos dos serviços consumidos, viabilizando um mercado de derivadores de informação.

Apesar de ser centrado na ideia da integração e disponibilização de dispositivos, o Middleware proposto permite que fontes de dados oriundas não apenas de dispositivos físicos sejam integradas. Um exemplo destas fontes seria um *feed* do Twitter: as buscas utilizadas por usuários ou qualquer outro tipo de informação sendo disponibilizada para desenvolvedores, na construção de uma internet de todas as coisas. A arquitetura proposta baseia-se na ideia de componentes de software chamados neste trabalho de Adapters (Figura 1). A lógica de integração com dispositivos físicos é tratada e encapsulada pelos Adapters, fazendo destes interfaces padronizadas de controladores heterogêneos de sistemas, fontes de dado abertos ou oriundos de diversos dispositivos. Dessa forma, cada dispositivo possui um identificador único no middleware composto pelo identificador do *Adapter* seguido pelo identificador do dispositivo. As mensagens geradas por diferentes *Adapters* são encapsuladas em um formato padrão do middleware de integração, que as roteia, persiste e prioriza de formas distintas.

Para realizar a integração de um dispositivo ou sistema, o integrador (ou produtor da informação) deve implementar as responsabilidades definidas na superclasse AbstractAdapter listadas abaixo:

- **Conectar**: conecta-se ao controlador do dispositivo, iniciando a comunicação com o dispositivo;

- **Registrar/Desregistrar**: registra suas meta-informações no Registro;
- **Testar**: testa se a conexão está ativa. A simples leitura da informação pode ser utilizada como teste de conexão;
- **Buscar**: realiza a busca de dispositivos individuais controlados pelo *Adapter* que se enquadrem no critério de busca especificado.
- **Ler**: realiza a leitura do valor atual do dispositivo representado pelo identificador;

Após a implantação, a camada core é responsável por inicializar, conectar e registrar os *Adapters* no Registro, que passam então a ser indexados pelo serviço de busca. O serviço de busca localiza dispositivos através de parâmetros tais como localização, tipo de dispositivo, produtor e confiabilidade. Uma vez de posse do identificador do *Adapter* é possível registrar-se para o consumo das informações de maneira síncrona ou assíncrona:

- **Síncrona**: Permite ao usuário solicitar a um *Adapter* que leia a informação de um determinado dispositivo sob demanda, de maneira bloqueante, isto é, o cliente fica à espera da mensagem de resposta de um dispositivo especifico.
- **Assíncrona**: O consumidor registra o interesse em um fluxo de dados e quando a informação é lida pelo *Adapter*, esta é roteada para todos os observadores registrados. Esta abordagem possibilita que aplicações recebam apenas as mensagens que sejam do seu interesse.

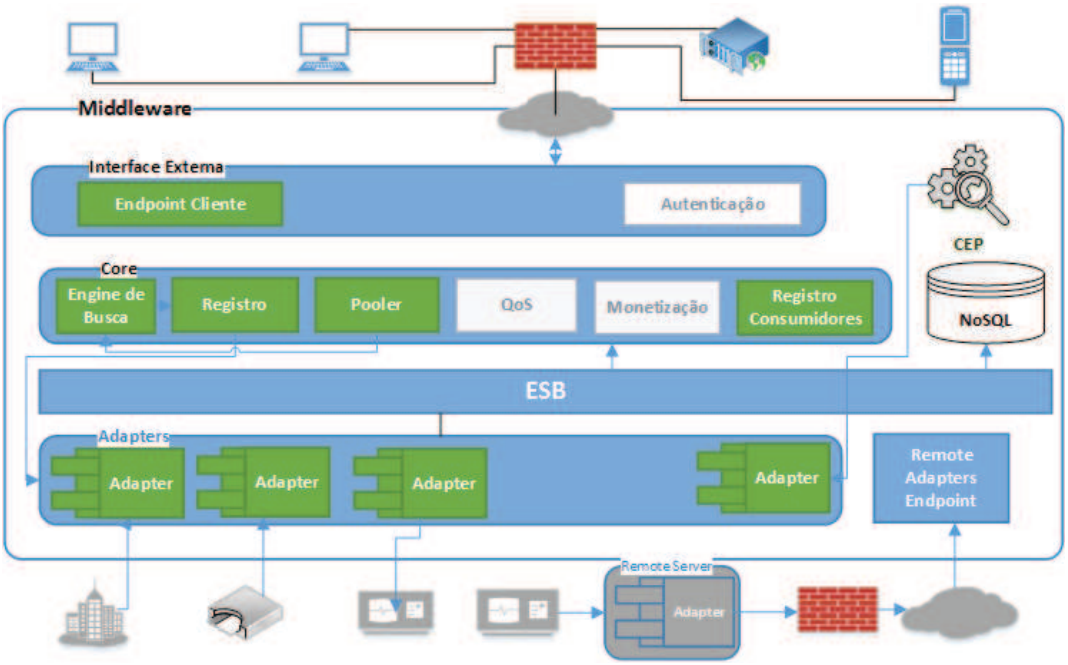

Figura 1. Arquitetura da plataforma

A arquitetura descrita acima envolve os componentes, brevemente descritos abaixo:

- **Pooler**: Responsável por verificar novas mensagens periodicamente em todos os *Adapters* assíncronos, transformá-las e enviá-las ao barramento;
- **QoS**: Responsável por medir a disponibilidade de cada serviço, gerando índices de confiabilidade, que permitem aos desenvolvedores escolher entre serviços

semelhantes baseado em preço e disponibilidade. O feedback dos usuários também é usado como entrada para esse serviço;

- **Persistência**: Responsável por persistir o fluxo de informação em uma base de dados NoSQL para posterior análise de dados;
- **Engine de Busca**: Permite à comunidade de consumidores de informação, buscar por feeds de informação utilizando atributos de busca (ex.: localização, tipo de dispositivo, produtor, etc.);
- **Registro de consumidores**: Registra o interesse de consumidores em dados seguindo um critério de busca;
- **Registro**: Registra os *Adapters*, agindo como base de dados da Engine de Busca;
- **Deploy**: Permite a produtores de informação realizarem o deploy dos *Adapters* de maneira simplificada, de maneira análoga aos serviços de cloud computing;
- **CEP**: Realiza processamentos complexos de maneira escalável. As capacidades de CEP são disponibilizadas através de interface para o usuário para a composição de novos *Adapters*;
- **Monetização**: Responsável por contabilizar a quantidade de mensagens consumidas para fins de tarifação e geração de créditos;

4. Implementação e Caso de Estudo

4.1 Tecnologias utilizadas

Para a implementação da plataforma foi utilizado o Mule ESB por se tratar de um ESB aberto e extensível. Foi usado o OpenMQ para prover o middleware de disponibilidade e robustez. Para a implementação do serviço de CEP foi utilizado o EsperCEP. A comunicação com as aplicações clientes utilizou JSON sobre HTTP implementando o estilo RESTFul.

4.2 Estudo de caso: Monitoramento de ônibus utilizando CEP

Visando validar a proposta do middleware apresentado, foi desenvolvida uma aplicação (Figura 2) para monitoramento e roteamento de transporte público em cidades (vídeo de disponível online[1]).

Problema: Quando se pensa na construção de cidades inteligentes uma grande quantidade de recursos é despendida no sensoriamento e construção de aplicações centralizadas, construídas e mantidas por uma única entidade. O gerenciamento da frota de ônibus com base na demanda de usuários e condições de trânsito tem se mostrado um desafio para as grandes cidades. As técnicas comumente utilizadas fazem uso de uma fonte de dados única para monitoramento e geração de rotas de ônibus, tornando as medições imprecisas, incompletas.

Solução: A aplicação de "monitoramento de ônibus para cidades inteligentes" utiliza-se de dados reais de GPS dos ônibus da região metropolitana do Recife. A partir destes dados é feito o monitoramento da velocidade média de cada um dos ônibus da

[1] http://snipurl.com/28kefua

cidade. Os dados de GPS são disponibilizados na plataforma através de um *Adapter* que realiza a leitura dos dados simulando um fluxo de informações de GPS em tempo real.

A aplicação cliente registra no Serviço de Busca uma consulta assíncrona contendo a posição geográfica, o raio e tipo de dispositivo (GPS_ONIBUS) de interesse. A partir de então o middleware filtra o *feed* de informações para este cliente que exibe a posição de cada ônibus no Mapa da aplicação de monitoramento. A aplicação também faz uso do serviço de CEP para calcular a velocidade média dos ônibus. A cada detecção de baixa velocidade é gerada uma mensagem para um *Adapter* que dispara alertas sobre as condições de trânsito dos ônibus da cidade (trânsito lento/trânsito rápido), que podem ser utilizados por outras aplicações.

Figura 2. Screenshots das aplicações construídas sobre a plataforma

Uma segunda fonte de dados para aplicação é o *feed* de consultas realizadas pelos usuários para determinado trajeto através da aplicação de roteamento de ônibus ciente de contexto. Os dados provenientes deste *Adapter* são analisados pelo CEP, que alerta a central de monitoramento quanto à necessidade de mais linhas em determinada rota de acordo com a quantidade de usuários consultado roteiros para essa linha. Esta aplicação indica as melhores rotas de ônibus entre dois pontos, levando em consideração as condições de trânsito (fornecidas pelo CEP da primeira aplicação coletadas através de um *Adapter*) e do contexto do usuário (inferido a partir da localização, compromissos, acelerômetro, etc.).

As duas aplicações validaram o cenário de consumidores (GPS) e produtores (consultas dos usuários por determinada linha) e agregadores de informação (análise do tráfego através do CEP, que gera novos dados). A plataforma proposta possibilitou a rápida disponibilização de dados com da integração e a criação de novas fontes de dados através da utilização de *Adapters*. Questões de roteamento, priorização e consulta e consumo dos dados ficam a cargo da plataforma.

6. Conclusão e Trabalhos Futuros

Este artigo apresentou um middleware para integração de dados urbanos de fontes heterogêneas de dados urbanos, com foco na Internet das Coisas. A arquitetura é baseada em um barramento de serviços, e o objetivo principal é possibilitar a disponibilização das informações abstraindo complexidades de diversas tecnologias e protocolos de comunicação existentes de forma a retirar do produtor da informação a complexa infraestrutura necessária para disponibilizar estas informações em tempo real para um público de desenvolvedores.

Como trabalhos futuros esperam-se o desenvolvimento e refinamento de todos os módulos da plataforma, testes de carga, integração transparente ao padrões e protocolos

utilizados para a Internet das Coisas. Uma outra questão em aberto é o desenvolvimento de interfaces que permitam que os dispositivos possam ser acionados utilizando o middleware, ao invés de apenas lê-los. No tangente à análise dos dados, vislumbra-se o desenvolvimento de uma linguagem que permita a análise de dados em tempo real (ex.: através do CEP) e dados de históricos (ex.: através de NoSQL).

Referências

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: a survey. Computer networks, 38(4), 393-422.

Ashton, K. (2009). That 'Internet of Things' Thing. RFiD Journal, 22, 97-114.

Hernández-Muñoz, J. M., Vercher, J. B., Muñoz, L., Galache, J. A., Presser, M., Gómez, L. A. H., & Pettersson, J. (2011). Smart cities at the forefront of the future internet. In The future internet (pp. 447-462). Springer Berlin Heidelberg.

Gama, K., Touseau, L., & Donsez, D. (2012). Combining heterogeneous service technologies for building an Internet of Things middleware. Computer Communications, 35(4), 405-417.

Ganti, R. K., Ye, F., & Lei, H. (2011). Mobile crowdsensing: Current state and future challenges. Communications Magazine, IEEE, 49(11), 32-39.

Haller, S., Karnouskos, S., & Schroth, C. (2009). The internet of things in an enterprise context. In Future Internet–FIS 2008 (pp. 14-28). Springer Berlin Heidelberg.

Schmidt, M. T. (2005). The enterprise service bus: making service-oriented architecture real. IBM Systems Journal, pp. 781-797.

Valente, B., & Martins, F. (2011, August). A middleware framework for the Internet of Things. In AFIN 2011, The Third International Conference on Advances in Future Internet (pp. 139-144).