Marque a opção do tipo de trabalho que está inscrevendo:

(✓) Resumo **( ) Relato de Caso**

MODELO DE ARQUITETURA ORIENTADA A SERVIÇOS PARA GERENCIAMENTO E MANIPULAÇÃO DE DADOS AGRÍCOLAS

AUTOR PRINCIPAL: Felipe Quevedo Giovanoni
CO-AUTORES: Mauricio Alex Zientarski Karrei, Leandro Bonfante, Prof. Dr. Carlos Amaral Hölbig
ORIENTADOR: Prof. Dr. Willington Pavan
UNIVERSIDADE: Universidade de Passo Fundo

INTRODUÇÃO

A massa de dados gerada por sistemas utilizados na Agricultura de Precisão requer uma solução de recebimento, processamento e apresentação das informações coletadas. Para isso, se faz necessária a análise e avaliação de diferentes tecnologias disponíveis e capazes de suprir a demanda no desenvolvimento dos sistemas de informação agrícola, uma vez que há pouca disponibilidade de aplicações que atendam a esta necessidade (1).

Este projeto tem como objetivo o desenvolvimento de uma solução computacional aplicada à coleta, armazenamento e transmissão de dados agrícolas, obtidos a partir de dispositivos remotos, utilizando uma API de serviços para a *Web*. As tendências em tecnologias de desenvolvimento nesta área apontam para a utilização de modelos SOA (2), destacando-se, especialmente, as tecnologias que com estilo arquitetural RESTful - serviços baseados em transferência de estado representacional, o que foi escolhido para nortear o desenvolvimento deste projeto (3).

DESENVOLVIMENTO

Neste projeto, após a análise de requisitos, realizada de acordo com o padrão IEEE 830-1998 (4) foram testadas as tecnologias Jersey Framework e Node JS (Tabela 1) para desenvolvimento de aplicações para a *Web*. Foi selecionado o Node JS devido ao seu melhor desempenho em ambientes de alta escalabilidade e à melhor resposta a alto número de requisições concorrentes. Para desenvolvimento em modelo RESTful e com a tecnologia Node JS, foi utilizado o conjunto MEAN, isto é, o conjunto das tecnologias: Mongo DB, tecnologia para desenvolvimento de Banco de Dados; Express JS, *framework* de desenvolvimento para serviços da

Web em Node JS; Angular JS, módulo de desenvolvimento para interface de usuário (UI) para a *Web*; e Node JS, ambiente multi-plataforma para desenvolvimento de aplicações no lado do servidor.

Para a metodologia de projeto e desenvolvimento de software utilizou-se o método XP (*extreme programming*) e o padrão UML de documentação e modelagem (5). O modelo arquitetural RESTful foi escolhido e utilizado de modo a se criar uma API de serviços na *web*. Os *web services* (serviços par a *web*) criados consistem na definição de serviços autônomos, que são identificados por URL's (rotas) e com interfaces de processamento de mensagens, as quais são definidas por meio dos métodos fundamentais da arquitetura REST (6). Estes métodos são: *get* (para obter informações), *post* (para inserir informações novas), *put* (para alterar um registro existente) e *delete* (para remover um registro existente). Como ilustra, de maneira geral, a figura 1, na organização deste sistema os dispositivos eletrônicos em implementos agrícolas comunicam-se com o servidor através de requisições URL, utilizando os métodos citados e a administração do sistema é feita através de uma interface de usuário (UI) acessível pela *Web*.

Quanto a fase de desenvolvimento, foi desenvolvida uma API de serviços capaz de receber dados dos mais variados dispositivos, tais como implementos agrícolas (tratores, sameadoras, pulverizadores, etc) e estações meteorológicas automáticas, remotamente conectadas à internet, assim como disponibilizar estes dados para os usuários por meio da mesma API.

Os dados recebidos são encaminhados para verificação e validação das informações, além de permitir a identificação de qual tipo de dispositivo que se está recebendo os dados. Desta forma, o a estrutura criada permite adaptar o procedimento de processamento de cada requisição recebida, e responder adequada a cada dispositivo.

Identificou-se, também, a possibilidade da centralização do recebimento de dados agrícolas, utilizando uma estrutura dinâmica capaz de se adaptar a diferentes ambientes. Exemplos incluem a integração com sistemas de bancos de dados e de apresentação de dados, o que oferece alta performance e baixa latência na resposta a requisições através da rede.

CONSIDERAÇÕE S FINAIS:

A API desenvolvida, e aqui apresentada, mostra-se capaz de receber, processar e armazenar dados advindos dos mais varados dispositivos agrícolas, em alta escala. Por meio da interface *web* desenvolvida, permite-se a administração dos recursos e troca de mensagens com os dispositivos remotos, atendendo os requisitos iniciais do projeto.

REFERÊNCIAS

(1) MURAKAMI, Edson. Uma infra-estrutura de desenvolvimento de sistemas de informação orientados a serviços distribuídos para agricultura de precisão, 192 p. Tese (Doutorado). Universidade de São Paulo, 2006.
(2) RODRIGUEZ, A. Restful web services: The basics. IBM developerWorks, 2008.
(3) BROWN, A.; JOHNSTON, S.; KELLY, K. Using service-oriented architecture and component-based development to build web service applications. Rational Software Corporation, 2002.
(4) IEEE COMPUTER SOCIETY. IEEE Recommended Practice for Software Requirement Specification. IEEE Std 830-1998. New York - EUA, 2009.
(5) SOMMERVILLE, I. Engenharia de software. São Paulo: Pearson Addison Wesley, 2013. 9 ed.
(6) FIELDING, R. Architectural Styles and the Design of Network-based Software Architectures, 2000, 180p. Dissertação (Doutorado em Informação e Ciência da Computação) - University of California, Irvine, California, EUA.

ANEXOS

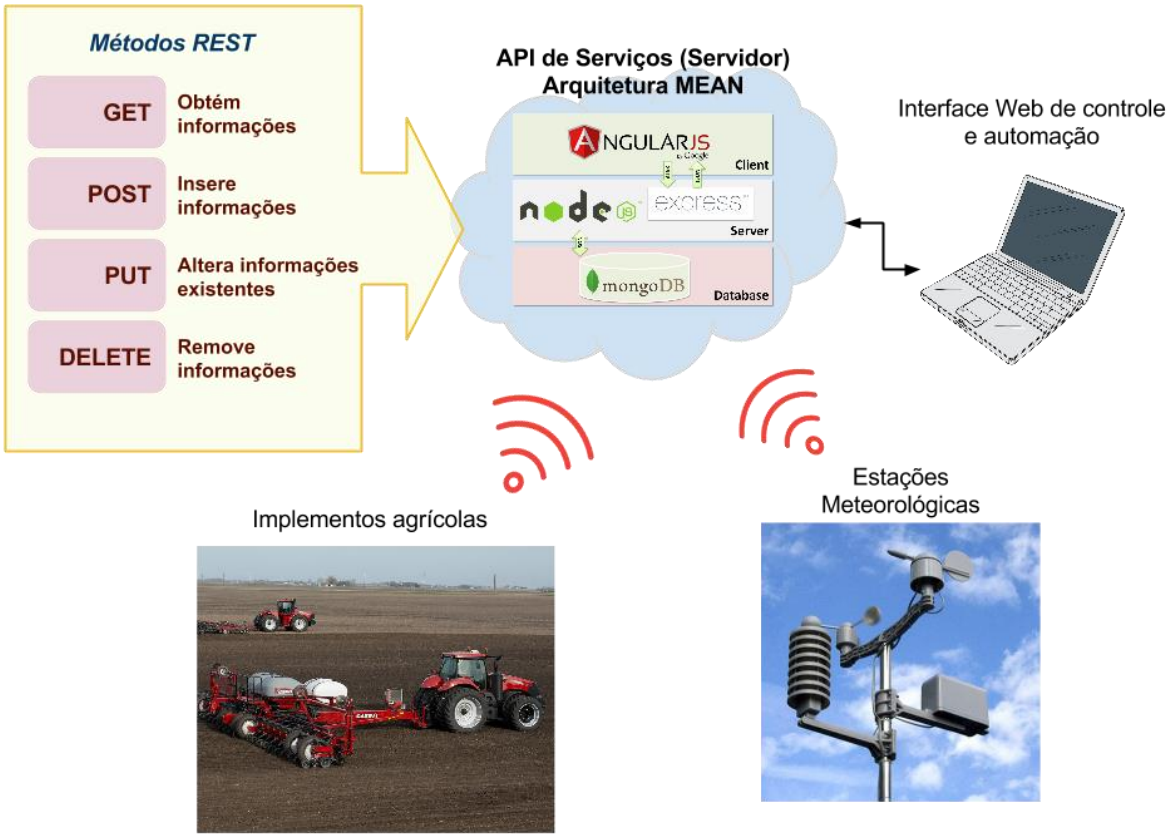

Figura 1 - Modelo arquitetural REST para um sistema de gerenciamento de implementos agrícolas

NODE JS – RESTful						Java Jersey – RESTful				
Número de Requisições	Nível de Concorrência	Tempo total da requisição (ms)	Tempo por requisição (ms) – NodeJS	Requisições por segundo		Número de Requisições	Nível de Concorrência	Tempo total da requisição (ms)	Tempo por requisição (ms) – Jersey	Requisições por segundo
GET										
100	1	1,203	1,203	831.45		100	1	3,356	3,356	297.98
100	10	5,783	0,578	1729.06		100	10	8,992	0,899	1112.10
100	50	32,54	0,651	1536.55		100	50	43,247	0,865	1156.15
100	100	51,142	0,511	1955.34		100	100	55,04	0,55	1816.86
1000	1	0,847	0,847	1180.62		1000	1	1,978	1,978	505.51
1000	10	5,219	0,522	1916.03		1000	10	5,706	0,571	1752.47
1000	50	22,238	0,445	2248.43		1000	50	19,802	0,396	2524.96
1000	100	43,104	0,431	2319.99		1000	100	388,25	0,388	2575.65
POST										
100	1	21,632	21,632	46.23		100	1	31,849	31,849	31.40
100	10	213,936	21,394	46.74		100	10	270,177	27,018	37.01
100	50	1134,184	22,684	44.08		100	50	1253,273	25,065	39.90
100	100	2127,919	21,279	46.99		100	100	2565,983	25,66	38.97
1000	1	22,457	22,457	44.53		1000	1	30,04	30,04	33.29
1000	10	216,508	21,651	46.19		1000	10	241,27	24,127	41.45
1000	50	1065,914	21,318	46.91		1000	50	1154,846	23,097	43.30
1000	100	2134,162	21,342	46.86		1000	100	2350,572	23,506	42.54

Tabela 1 - Resultados de testes entre as tecnologias de desenvolvimento REST (Fonte: Pesquisa do autor)