

# Implementação do protocolo CoAP para serviços de monitoramento em Redes de Sensores Sem Fio

Rafael de Lucena Valle

Universidade Federal de Santa Catarina

*rafaeldelucena@inf.ufsc.br*

July 8, 2014

# Agenda

## Introdução

- Contextualização

- Motivação

- Objetivos

## Conceitos Relacionados

## Trabalhos Relacionados

## CoAP

## Implementação

## Avaliação

- Ambiente de Testes

- Resultados Experimentais

## Conclusão

- Conclusões

- Trabalhos Futuros

# Contextualização

## Redes de sensores sem fio

Centenas ou milhares de nós sensores, características: pouca memória, pouco alcance do rádio, baixa capacidade de processamento e bateria, e custo reduzido.

## Internet das Coisas

Objetos do cotidiano que possuem representações de seus recursos na Internet e capazes de interação com pessoas e outros objetos.

## Protocolos de aplicação

- ▶ HTTP
- ▶ XMPP
- ▶ MQTT
- ▶ **CoAP**, RFC 7252 desde 26/06/2014.

# Motivação

## Interoperabilidade

Comunicação entre diversos dispositivos diferentes utilizando padrões.

## Avanço Tecnológico

A indústria dos semicondutores e a miniaturização dos componentes.

## Ambientes Inteligentes

Novas aplicações que consumam contexto.

## Vasta cobertura da tecnologia

Cerca de 5570 municípios no Brasil possuem pelo menos uma ERB.

# Objetivos

## Objetivo Geral

Descrever, implementar e integrar na Internet serviços web de uma rede sensores sem fio.

## Objetivos Específicos

- ▶ Portar CoAP para plataforma alvo.
- ▶ Desenvolver a aplicação gateway GPRS / 802.15.4.
- ▶ Desenvolver uma WEB API para coleta da informação.
- ▶ Avaliar a solução desenvolvida.

# Conceitos Relacionados

## REST

Conjunto de princípios e restrições arquiteturais para o desenvolvimento de sistemas distribuídos.

## Restful

Práticas de desenvolvimento para aplicações web baseados no REST.

# Trabalhos Relacionados

## Contiki

Sistema Operacional com solução de código aberto e possui pilha IP/UDP/CoAP completa para servidor e cliente. Possui suporte a diversas plataformas: Econotag, OpenMote, Micaz, MSP430, ...

## OpenWSN

Conjunto de bibliotecas de código aberto para montar uma aplicação que utilize a possui pilha IP/UDP/CoAP. Também possui suporte a diversas plataformas: OpenMote, Zolertia Z1, WSN430, ...

## Libcoap

Biblioteca de código aberto em C, que implementa o protocolo CoAP, que utiliza sockets POSIX.

# Constrained Application Protocol: parte 1

Protocolo proposto por CoRE group com as seguintes características:

- ▶ Baixo overhead na validação do pacote e do cabeçalho.
- ▶ Suporte a URI.
- ▶ Descoberta de recursos, GET para /well-know/core.
- ▶ Descoberta de serviços com socket multicast.
- ▶ Protocolo desenvolvido para minimizar complexidade do mapeamento para HTTP.



## Constrained Application Protocol: parte 2

- ▶ Suporte a diferentes mídias: text/plain, charset=utf-8, application/link-format, application/xml, application/octet-stream, application/exi, application/json.
- ▶ Cache simples baseado no tempo de vida (max-age) do dado captado. (opcional)
- ▶ Sistema de assinatura e publicação. (opcional)
- ▶ Confiabilidade na entrega de mensagens. (opcional)

# Constrained Application Protocol: Formato

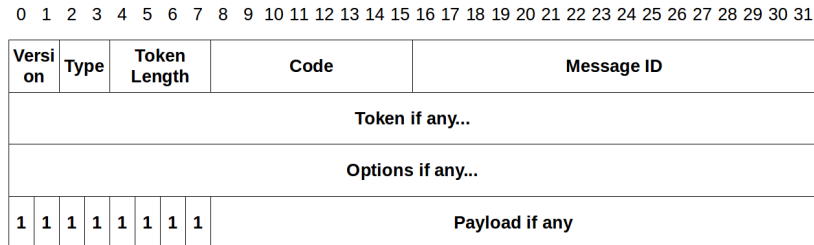
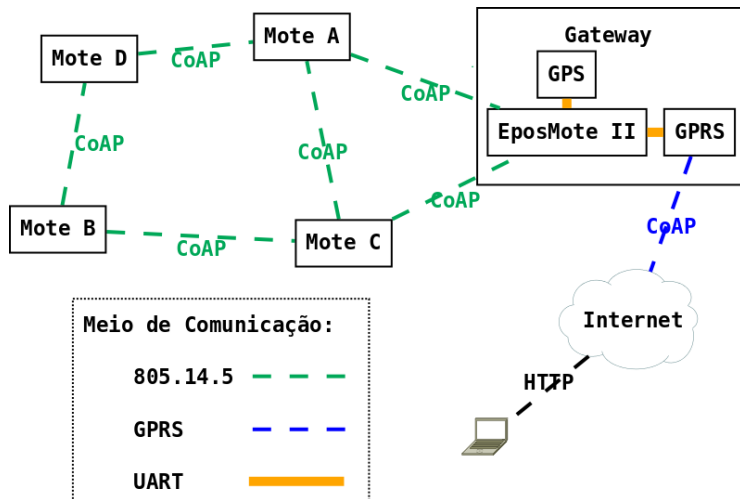


Figure : Formato do pacote em bits

# Implementação: Arquitetura



# Implementação: Requisitos

## Requisitos Funcionais:

- ▶ Envio de requisição e recebimento de respostas CoAP.
- ▶ Integração com a Internet utilizando GPRS.

## Requisitos Não Funcionais:

- ▶ Plataforma alvo com restrições de memória.

# Implementação: Desenvolvimento

- ▶ Levantamento de requisitos para porte de uma biblioteca CoAP. (libCoap, libCantCoap, microCoap, ...)
- ▶ Verificação do funcionamento da biblioteca CoAP na plataforma, utilizando testes unitários.
- ▶ A implementação do agendador de transmissão utiliza alarmes de disparo único.
- ▶ Testes de interoperabilidade utilizando exemplos funcionais (coap://coap:coap.me).

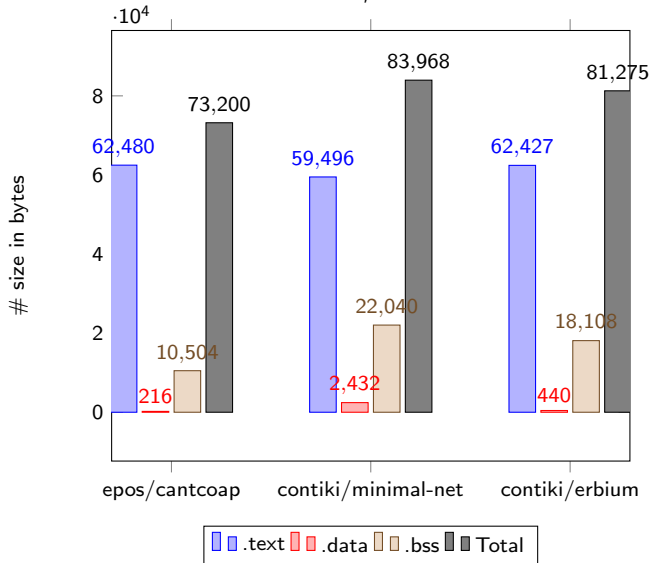
# Ambiente de Testes: requisitos mínimos da aplicação

Abaixo informações sobre as plataformas utilizadas para comparação.

	<b>contiki/client</b>	<b>epos/client</b>
<b>Microcontrolador</b>	Freescall MC13224V	Freescall MC13224V
<b>Plataforma Alvo</b>	Econotag	EposMote II
<b>Sistema Operacional</b>	Contiki	EPOS
<b>Ferramentas binárias</b>	GNU Binary Utilities (2.18.50-sg++)	arm-elf-binutils 4.4.4
<b>Compiladores</b>	GNU C & C++ Compilers (4.3.2-sg++)	arm-elf-g++ 4.4.4
<b>Bibliotecas C</b>	Newlib C (1.16.0-sg++)	Newlib C

# Resultados Experimentais: Requisitos Mínimos

Plataforma com 128KB Flash, 96KB RAM de memória.



# Ambiente de Testes: Interoperabilidade

## Máquina Host

<b>ISA</b>	x86 32
<b>SO</b>	GNU/LINUX 2.6.32-5-686
<b>Distro</b>	Debian 6.0.9

## Servidores

- ▶ Licoap: ETSI e exemplo.
- ▶ LibCantCoap exemplo.
- ▶ Contiki Erbium server e Minimal-Net server.



# Resultados Experimentais: Interoperabilidade 1

CENÁRIO	RESULTADO
<b>Testes para especificação básica CoAP</b>	
Tratar GET, confirmável.	✓
Tratar POST, confirmável.	✓
Tratar PUT, confirmável.	✓
Tratar DELETE, confirmável.	✓
Tratar GET, sem confirmação.	✓
Tratar POST, sem confirmação.	✓
Tratar PUT, sem confirmação.	✓
Tratar DELETE, sem confirmação.	✓
Tratar GET com resposta separada.	✓
Tratar requisição com Token.	✓
Tratar requisição sem Token.	✓
Tratar requisição opções URI-Path.	✓
Tratar requisição opções URI-Query.	✓
Interoperabilidade em contexto de perda (CON mode, piggybacked response)	✓
Interoperabilidade em contexto de perda (CON mode, delayed response)	✓
Tratar GET com resposta separada, sem confirmação.	✓

Table : Resultados IOT Plugtest.

## Resultados Experimentais: Interoperabilidade 2

CENÁRIO	RESULTADO
<b>Testes para validar o formato de dados CORE link Format</b>	
Descoberta de recursos well-known.	X
Utilização de consulta para filtrar resultados.	X
<b>Testes para validar a transferência de blocos</b>	
Transferência de blocos grandes utilizando GET (negociação antecipada).	X
Transferência de blocos grandes utilizando GET (negociação atrasada).	X
Transferência de blocos grandes utilizando o PUT.	X
Transferência de blocos grandes utilizando o POST.	X

Table : Resultados IOT Plugtest.

## Resultados Experimentais: Interoperabilidade 3

CENÁRIO	RESULTADO
<b>Testes para observação de recursos</b>	
Tratar observação de recursos.	X
Parar a observação de recursos.	X
Detecção de deregistro do cliente (Max-Age).	X
Detecção de deregistro do servidor (client OFF).	X
Detecção de deregistro do servidor (RESET explícito).	X

Table : Resultados IOT Plugtest.

# Conclusões

- ▶ Desenvolvimento de um gateway simplificado para redes de sensores sem fio e a Internet.
- ▶ Protocolo adequado para redes com um número grande de nós.
- ▶ A falta de um tipo de mídia específico para representação de medidas dos sensores. (SEML rascunho)

# Trabalhos Futuros

- ▶ A implementação da versão segura do protocolo CoAP, utilizando DTLS para comunicação segura entre os nós.
- ▶ Uma implementação de servidor CoAP completa para executar utilizando a plataforma do EPOSMote II.
- ▶ Melhorar as métricas de avaliação: vazão de mensagens, round-time trip e consumo de energia e memória ao longo do tempo.
- ▶ Desenvolvimento de um gateway que utilize Software Defined Transceiver.
- ▶ Um gerador de código que utiliza como entrada uma linguagem de especificação dos recursos e a saída código ANSI C mínimo de um servidor web utilizando CoAP e seus respectivos recursos.
- ▶ Modelo de apresentação de recursos para os usuários.

Perguntas?