GT-TeI
RT2 - Proposta do protótipo

**Silvana Rossetto[1], Bruno Silvestre[2], Noemi Rodriguez[3],
Adriano Branco[3], Leonardo K. L. S. Kaplan[3], Ian B. Lopes[3],
Douglas V. Santana[2], Vinicius B. da Silva Lima[2],
Raul G. M. de Freitas[1] e Denis Takeo Aoki[1]**

[1]Departamento de Ciência da Computação (DCC), Instituto de Matemática (IM)
Universidade Federal do Rio de Janeiro (UFRJ)

[2]Instituto de Informática
Universidade Federal de Goiás (UFG)

[3]Departamento de Informática
Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio)

23 de março de 2014

1. Arquitetura do protótipo

O protótipo que será desenvolvido no escopo deste projeto consistirá de: (1) o **testbed de sensores** localizado em um ambiente fechado e constituído de pelo menos dois tipos distintos de plataformas de sensores; e (2) um **portal Web** para acesso remoto e gerência do testbed de sensores.

O levantamento de requisitos para o projeto deste protótipo foi realizado com base no estudo de outros testbeds (de finalidades similares) e das nossas necessidades como usuários. Entre os testbeds estudados, o WISEBED [WISEBED 2008] e o Motelab (Harvard Sensor Network Testbed) [Werner-Allen et al. 2005] tiveram influência especial.

Nesta seção, descreveremos a arquitetura do testbed de sensores e do portal Web para acesso remoto a esse testbed.

1.1. Arquitetura do testbed de sensores

Um testbed para Redes de Sensores sem Fio (RSSF) é tipicamente composto por um conjunto de nós sensores previamente distribuídos numa área específica. Cada nó da rede é interligado a um computador de controle. O computador de controle, por sua vez, é responsável pela carga do código executável em cada nó e pela interface de interação com o usuário durante a execução dos testes.

A diversidade de plataformas de sensores disponíveis combinada com as diferentes possibilidades de distribuição dos nós de uma rede de sensores possibilitam a construção de testbeds com características bastante distintas. Por exemplo, podemos ter um testbed com os nós distribuídos dentro de uma sala e conectados através de um HUB USB, ou um testbed com os nós distribuídos em várias salas de um edifício e conectados pela rede Ethernet já existente. Outro exemplo é a fonte de energia dos nós. Podemos utilizar uma fonte local e que depende da existência de uma tomada de energia, ou utilizar o recurso PoE (*Power over Ethernet*) para alimentar o dispositivo via rede Ethernet.

Algumas decisões de projeto, como o tipo de mote (plataforma de sensor) utilizado ou topologia da rede, afetam diretamente os tipos de testes que podem ser executados em um testbed. Um exemplo é a disposição física dos nós, a qual define o alcance do rádio entre os nós da rede. Podemos ter redes em que todos os nós estão no raio de alcance dos demais nós, ou redes com uma topologia hierárquica e que exigem algoritmos para roteamento de mensagens.

O testbed proposto no escopo deste projeto é voltado para ambientes fechados, sendo constituído por nós sensores independentes e de tipos diferentes, dispostos em uma topologia que requer comunicação multi-saltos, acoplados a um *board* para carga de código e transferência de dados de execução, com fonte de energia constante, gerenciado por um sistema de controle executado numa máquina central com interface de acesso remoto via Web. A Figura 1 apresenta a arquitetura geral desse testbed, destacando seus módulos principais: kit Nó e kit Servidor.

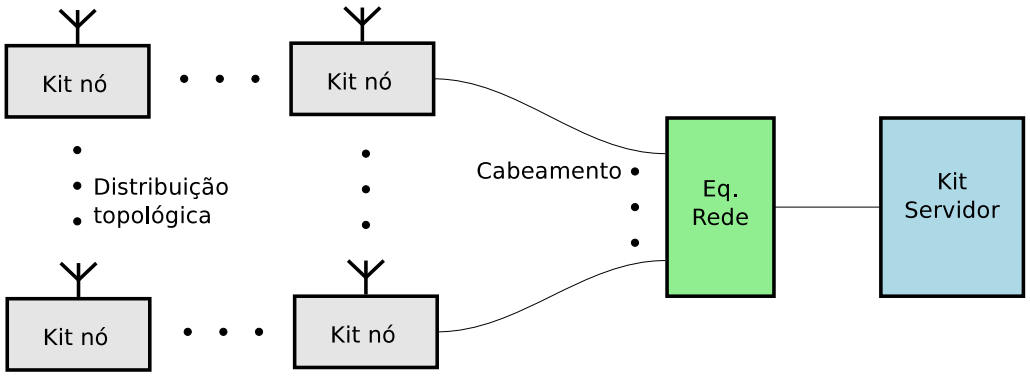

Figura 1. Arquitetura do testbed com seus módulos principais.

O núcleo da arquitetura é o **Kit Nó** que contém os componentes necessários para o funcionamento completo de um nó da rede do testbed: interfaces externas de rádio, sensores físicos, núcleo de processamento, interface de controle e fonte de alimentação de energia). Na Figura 2 apresentamos os elementos constituintes de um Kit Nó.

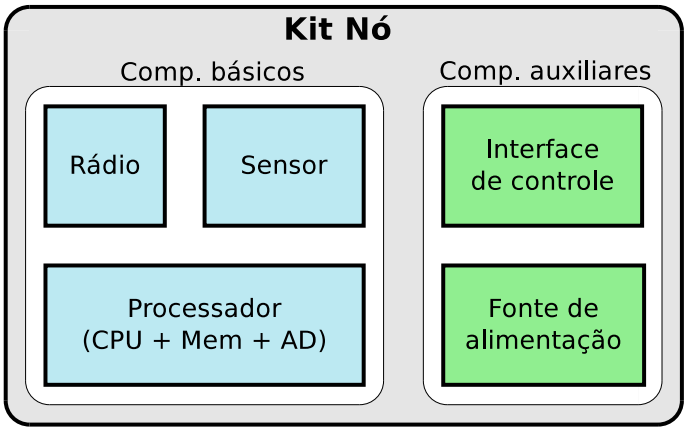

Figura 2. Componentes de um Kit Nó.

Cada Kit Nó será ligado, via cabo de rede, a um equipamento de rede (switch) conectado ao Kit Servidor. Essa ligação será usada para a carga das aplicações nos nós

sensores (direção Kit Servidor para Kit Nó) e transferência de informações de log de execução das aplicações (direção Kit Nó para Kit Servidor).

O **Kit Servidor** representa um equipamento ou conjunto de equipamentos (servidores e/ou desktops) utilizados no controle central e acesso remoto ao testbed. Este kit é responsável pelas funcionalidades de controle do testbed, servidor de aplicações e banco de dados.

O dimensionamento final do Kit Servidor depende da definição da arquitetura de software, da quantidade de nós da rede de sensores, das funcionalidades disponibilizadas e do grau de utilização do testbed. Para testbeds com acesso somente local, é necessária uma máquina simples com um conjunto de ferramentas para carga nos nós, controle e monitoração da execução. Essa máquina deve ter uma interface de conexão com a rede de controle e também de conexão com o nó da rede de sensores ligado a ela (estação base).

Para testbeds com acesso remoto é necessária uma arquitetura Web com um banco de dados para atender diversas funcionalidades como: controle de acesso, agendamento de janelas de execução, configuração da execução, controle e monitoração da execução e controle de logs. Nesse cenário, a máquina deve conter, além das interfaces de conexão com a rede de controle e estação base, conexão com a Internet.

1.2. Arquitetura do portal Web para acesso remoto e gerência do testbed

O portal Web deverá prover acesso remoto e gerência de uso do testbed de sensores. A Figura 3 apresenta a arquitetura funcional dos principais componentes do portal. Esses componentes serão agrupados em quatro funcionalidades: Agendamento, Configuração, Execução e Controle de Acesso.

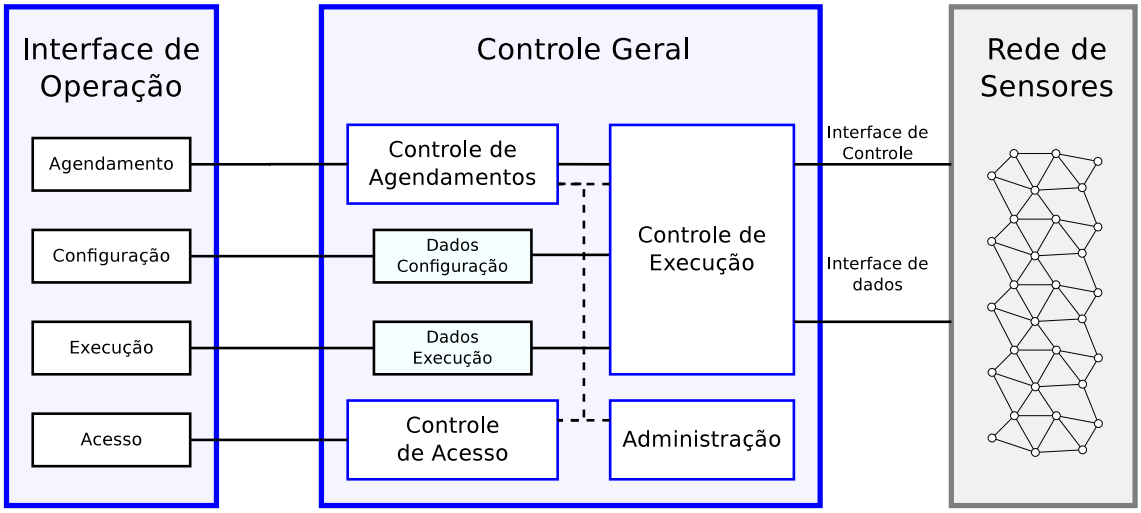

Figura 3. Arquitetura funcional do testbed.

Na Figura 4 apresentamos a **Interface de Operação** do portal que será composta pelos módulos Agendamento, Configuração, Execução e Acesso.

A Interface de Agendamento permite ao usuário consultar, reservar e cancelar as janelas de execução no testbed. Na Interface de Configuração o usuário pode configurar a topologia da rede com a ativação e desativação de determinados nós, carregar o seu código executável para execução na rede de sensores e definir condições de execução específicas

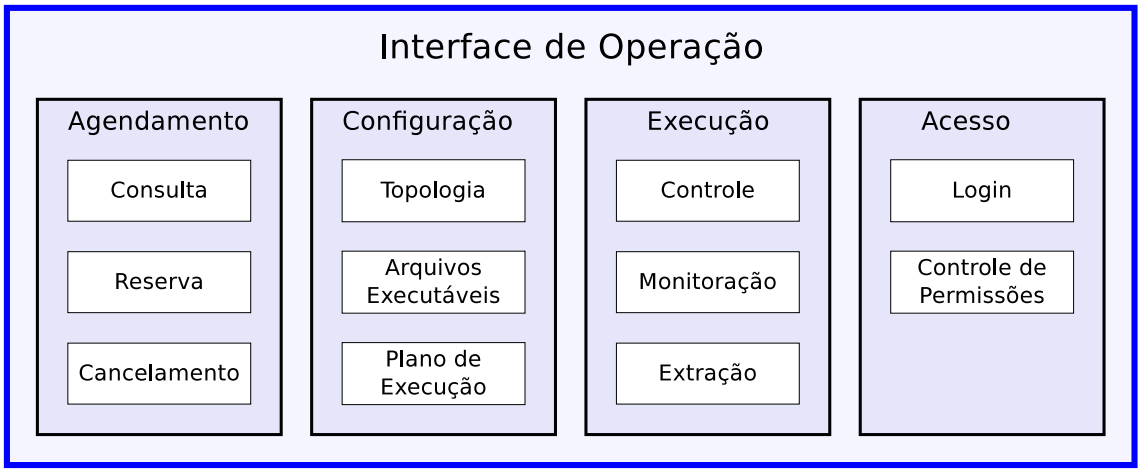

Figura 4. Interface de operação do portal Web.

para o seu experimento. A Interface de Execução permite ao usuário controlar e monitorar o andamento dos testes. A Interface de Acesso controla o acesso do usuário aos recursos do testbed.

1.2.1. Funcionalidade de Controle de Acesso

1. Interface de Acesso - Autenticação
   (a) Os acessos dos usuários serão controlados por meio de um cadastro de login e senha ou acesso federado (usando a Federação CAFe).
2. Interface de Acesso - Permissões
   (a) Todos usuários cadastrados terão permissão de execução.
   (b) O sistema de agendamento deverá prever a restrição de janelas de agendamento para diferentes tipos de usuários.

1.2.2. Funcionalidade de Agendamento

1. Interface de Agendamento - Consulta
   (a) Uma reserva da agenda abrange todos os nós do Testbed.
   (b) O usuário poderá consultar a situação atual do agendamento.
   (c) A unidade mínima de agendamento (janela) será de 30 minutos.
   (d) A visualização das janelas (Agenda) será sempre numa sequência de 24 horas com controle para avançar e retroceder na faixa de tempo.
   (e) A Agenda deverá ter um controle para selecionar um dia específico.
   (f) As janelas serão coloridas conforme a situação atual. Branco para janelas no passado ou ocupadas, verde-claro para janelas disponíveis, verde-escuro para janelas selecionadas e não salvas, cinza para janelas não ocupadas mas sem permissão de acesso, azul-claro para janelas reservadas pelo usuário e azul-escuro para seleção de janelas reservadas pelo usuário.
2. Interface de Agendamento - Reserva
   (a) O usuário poderá selecionar na Agenda as janelas disponíveis.

(b) Uma seleção deverá conter uma sequência de janelas, não podendo ter interrupções.
(c) O usuário poderá selecionar uma sequência por vez.
(d) O usuário deverá nomear a sequência selecionada. Este nome será utilizado na configuração da execução dos testes.
(e) O usuário poderá reservar uma sequência de no máximo 24 horas. Para sequências maiores, o usuário deverá contactar diretamente os responsáveis pelo Testbed.

3. Interface de Agendamento - Cancelamento
   (a) O usuário poderá selecionar uma das suas sequências na agenda.
   (b) O usuário poderá remover a sequência selecionada, apagando o registro da mesma e liberando as janelas.
   (c) A remoção sempre será de uma sequência completa.
   (d) A remoção de uma sequência deverá ter uma confirmação da ação.
   (e) Se a sequência tiver um plano de execução associado, o usuário deverá ser informado antes da confirmação da ação que o plano será invalidado.
   (f) Se o usuário remover uma sequência, o respectivo plano de execução, se existir, deverá ser invalidado.

1.2.3. Funcionalidade de Configuração

1. Interface de Configuração - Topologia
   (a) A topologia da rede deverá ser representada de forma textual (lista de nós).
   (b) A topologia poderá ser alterada com a ativação ou desativação de nós da rede.
   (c) Os nós deverão ser identificados na interface por IDs únicos. ID físico do nó no desenho da rede.
   (d) Cada nó terá um ID lógico para ser utilizado pelo programa do usuário (no padrão TOS_NODE_ID do TinyOS), o valor default será igual ao ID físico.
   (e) O usuário poderá alterar o valor do ID lógico de qualquer nó. A repetição de IDs lógico será permitida, mas o usuário deverá ser alertado.
   (f) O usuário poderá selecionar um nó ou grupo de nós da lista.
   (g) O usuário poderá configurar uma seleção de nós como ativo ou inativo.
   (h) O usuário deverá definir o controle da potência do rádio no seu programa.
   (i) A alteração de topologia obtida via potência do rádio não será visualizada no Portal.
   (j) O usuário poderá salvar diversas configurações de topologia.
   (k) O usuário poderá carregar e editar uma configuração de topologia previamente salva.
2. Interface de Configuração - Arquivos Executáveis
   (a) A interface deverá listar as combinações existentes de motes e sensores.
   (b) O usuário deverá associar um arquivo binário ".exe" default para cada combinação existente.
   (c) O usuário poderá associar um arquivo ".exe" específico para um ou mais nós.

(d) Os nós sem associação de arquivos individuais deverão carregar o arquivo default.
(e) O sistema executará a atualização automática do ID de mote durante a carga do código (conforme o procedimento padrão do TinyOS para TOS_NODE_ID). Essa atualização considera o ID lógico configurado.

3. Interface de Configuração - Plano de Execução
   (a) O plano de execução deve possibilitar a programação de paradas e reativações da rede e de nós individuais. Os estados do programa não serão mantidos nas paradas. Uma reativação é obtida com uma nova carga do programa executável.
   (b) A programação da execução sempre será feita no tempo relativo ao início da execução.
   (c) Uma configuração do plano de execução deve estar associada a uma configuração de topologia e a uma configuração de arquivos executáveis.
   (d) A configuração do plano de execução deverá ser salva para utilização futura. O usuário deverá indicar o nome e a descrição do plano.
   (e) O usuário poderá carregar e editar uma configuração da execução previamente salva.
   (f) Um plano de execução só será disparado se contiver uma janela de execução devidamente reservada pelo usuário.
   (g) Para janelas futuras, o plano será agendado para execução no horário previsto.
   (h) Para janelas já iniciadas e não finalizadas, a execução será disparada imediatamente.
   (i) A execução que ultrapassar o final da sua janela será automaticamente finalizada.

1.2.4. Funcionalidade de Execução

1. Interface de Controle
   (a) Deverá exibir os dados de uma execução do usuário.
   (b) Se existir uma execução em andamento, deverá exibir os dados dessa execução.
   (c) Se não existir uma execução em andamento, deverá exibir os dados de configuração da próxima execução planejada.
   (d) Uma execução será identificada pelo código de execução, código do usuário e descrição da execução.
   (e) O código de execução será criado na configuração da execução.
   (f) A interface deverá exibir a data, a hora de início, a hora de fim, a hora corrente e o tempo restante para finalização da janela de execução.
   (g) O usuário poderá parar e reiniciar uma execução em andamento.
2. Interface de Monitoração
   (a) A monitoração deverá exibir os comandos de controle do testbed e os dados da aplicação em execução.
   (b) Deverá exibir uma janela textual com o registro (log) dos comandos de controle do Testbed. (Timestamp+Comando)

(c) Deverá exibir uma janela textual com o registro (log) dos dados coletados na interface serial de cada nó. (Timestamp+Node+data)
(d) O usuário terá a opção de visualizar os dados específicos de cada nó.
(e) Para permitir a operação remota da aplicação, a porta de dados de alguns nós pode ser disponibilizada via um serviço TCP/IP.
(f) O serviço TCP/IP deverá disponibilizar o protocolo "AM Active Message" do TinyOS ou o protocolo IPV6.
(g) O usuário deverá selecionar qual protocolo irá utilizar para as portas de dados.
(h) A número da porta do serviço TCP/IP será o número do nó (ID físico) somado com 10000.
(i) O usuário poderá habilitar e desabilitar o serviço TCP/IP para os nós da rede.
(j) O usuário deverá executar localmente a sua aplicação específica que se conecta ao serviço TCP/IP do servidor do Testbed.

3. Interface de Extração
   (a) Todos os registros de controle e dados deverão ser salvos num arquivo na área do usuário no servidor do testbed. (Área de transferência de arquivos.)
   (b) O nome do arquivo de dados será composto pelo código do usuário e o código da execução.
   (c) O usuário terá a opção de transferir o arquivo para sua máquina local.
   (d) A interface de transferência de arquivo deve possibilitar a manutenção do diretório do usuário, como remover e transferir outros arquivos.

2. Requisitos de hardware e software

Nesta seção apresentamos os requisitos de hardware e software para a instalação e os testes do protótipo. Dividimos a seção em duas partes. Na primeira parte, apresentaremos o projeto físico do testbed de sensores e os recursos de hardware para a sua implantação. Na segunda parte, apresentaremos os requisitos de hardware e software para a instalação do portal Web de acesso ao testbed.

2.1. Projeto físico do testbed de sensores e recursos de hardware necessários

A Figura 5 apresenta uma visão da arquitetura física do testbed. Na visão apresentada, consideramos que a infraestrutura da rede Ethernet e da rede elétrica já existente está fora do escopo do nosso detalhamento de hardware.

O projeto considera dois tipos de Kit Nó, um com o mote **MicaZ** e o outro com o mote **TelosB**. A seguir especificamos cada um dos componentes dos dois tipos de Kit Nó.

O **Kit Nó MicaZ** é baseado em um mote padrão Berkeley, atualmente fabricado pela Memsic Corporation, com a seguinte configuração:

1. Módulo Processador + Rádio
   - Código fabricante - MPR2400CA
   - CPU - Atmel ATmega128L
   - ROM - 128K bytes
   - RAM - 4K bytes
   - Data - Flash memory 512K bytes

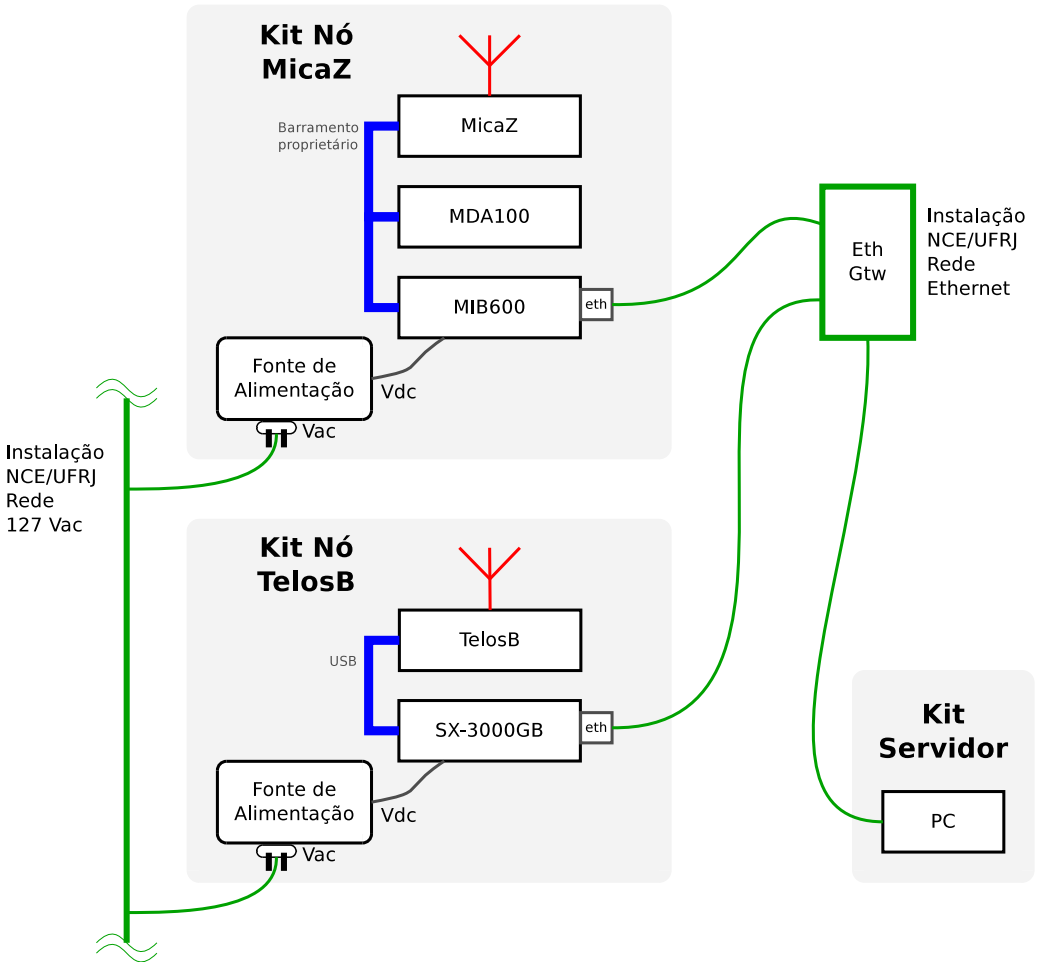

Figura 5. Projeto físico do testbed de sensores.

- Rádio - Chipcon CC2420 - 2.4 GHz IEEE 802.15.4
- Interface Controle - Conector Proprietário 51-pinos

2. Módulo Sensor
   - Código fabricante - MDA100CA
   - Sensor de temperatura
   - Sensor de luminosidade
   - Área para protótipo
   - Permite a conexão simultânea de mais 2 módulos com utilizando Conector Proprietário 51-pinos.
3. Módulo Interface de Rede
   - Código fabricante - MIB600
   - Conector RJ-45 100 Base
   - Interface IEEE 802.3 / IEEE 802.3af POE
   - Interface Controle - Conector Proprietário 51-pinos
   - Protocolos ARP, UDP/IP, TCP/IP, Telnet, DHCP, BOOTP, TFTP, Auto IP, and HTTP
4. Módulo de Alimentação
   - Disponibilizado em conjunto com o Módulo Interface de rede MIB600.
   - Compatível com o conjunto MicaZ + MDA100

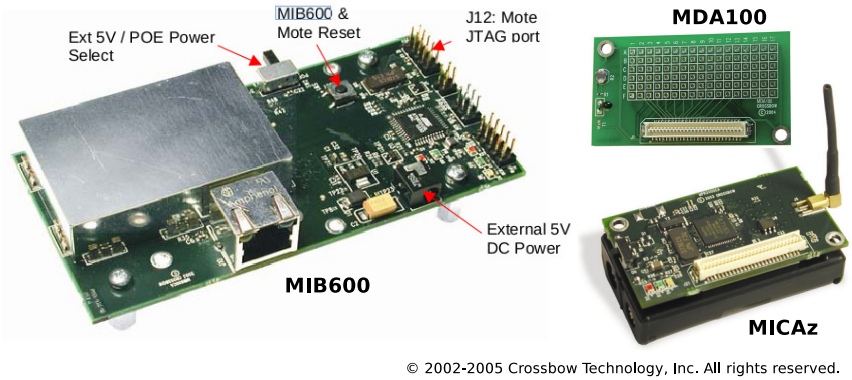

Figura 6. Componentes do Kit Nó MicaZ.

Na Figura 6 apresentamos as imagens dos três módulos do Kit Nó MicaZ.

O **Kit Nó TelosB** é formado por componentes de dois fabricantes: o mote TelosB, que também é baseado num mote padrão Berkeley, atualmente fabricado pela Memsic Corporation;e o adaptador SX-3000GB fabricado pela Silex Technology. O kit tem a seguinte configuração:

1. Módulo Processador + Rádio + Sensor + Interface Controle
   - Código fabricante - TPR2420CA
   - CPU - TI MSP430
   - ROM - 48K bytes
   - RAM - 10K bytes
   - Data - Flash memory 1024K bytes
   - Data - EEPROM 16K bytes
   - Rádio - Chipcon CC2420 - 2.4 GHz IEEE 802.15.4
   - Interface Controle - Padrão USB
   - Sensor de temperatura
   - Sensor de luminosidade
   - Sensor de umidade
2. Módulo Interface rede
   - Código fabricante - SX-3000GB
   - Conector RJ-45 10 / 100 / 1000 Base
   - Interface IEEE 802.3
   - Interface USB 2.0 Hi-Speed (A-Type)
   - Protocolo TCP/IP,
3. Módulo de alimentação
   - Disponibilizado em conjunto com o Módulo Interface de rede SX-3000GB.

Na Figura 7 apresentamos as imagens dos dois módulos do Kit Nó TelosB.

Os módulos que compõem cada tipo de kit deverão ser encapsulados em uma caixa plástica conforme indicação das Figuras 8 e 9.

Distribuição física prevista dos nós Os kits serão distribuídos pelos laboratórios no térreo do prédio do NCE na UFRJ. O projeto atual prevê a utilização de 35 Kits Nó MicaZ e 10 Kits Nó TelosB. Na Figura 10 apresentamos a proposta inicial de distribuição. Essa distribuição pode sofrer alterações conforme a disponibilidade de tomadas de rede e energia elétrica.

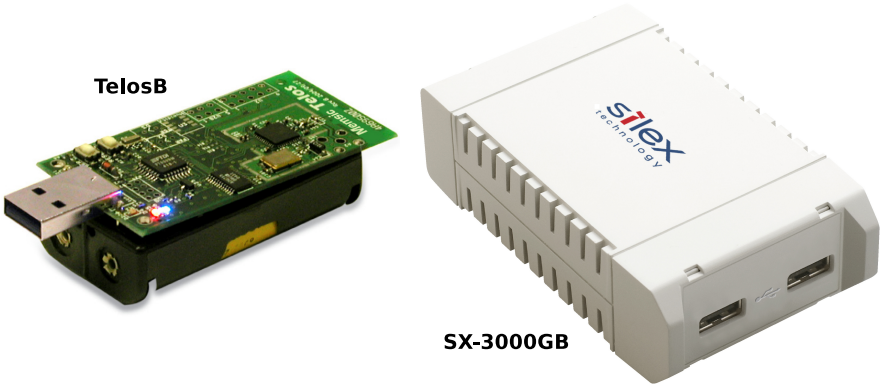

Figura 7. Componentes do Kit Nó TelosB.

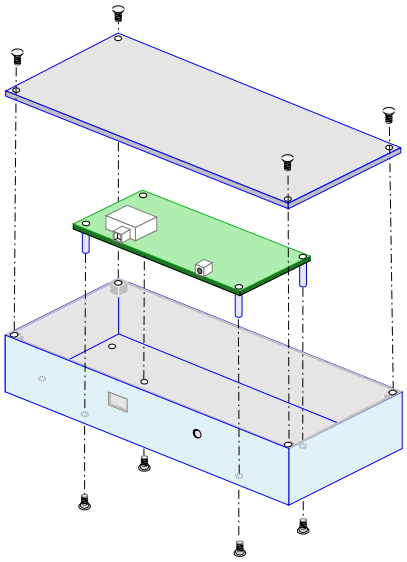

Figura 8. Montagem do Kit Nó MicaZ na caixa plástica.

2.1.1. Recursos de hardware e de software para instalação do testbed de sensores

Para a instalação do testbed de sensores, serão necessários os seguintes recursos de hardware:

- 35 plataformas **MicaZ** (adquiridas em projeto anterior);
- 35 placas de sensores **MDA100** (adquiridas em projeto anterior);
- 35 boards para carga de código **MIB600** (adquiridos em projeto anterior);
- 10 plataformas **TelosB** (adquiridas em projeto anterior);
- 10 **adaptadores SX-3000GB** (adquiridas em projeto anterior);
- 2 switches 24 portas (já adquiridos com recursos deste projeto);

2.2. Recursos de hardware e software para implantação do portal Web

Para a instalação e configuração do portal de acesso remoto e gerência do testbed serão necessários os seguintes recursos de hardware e software:

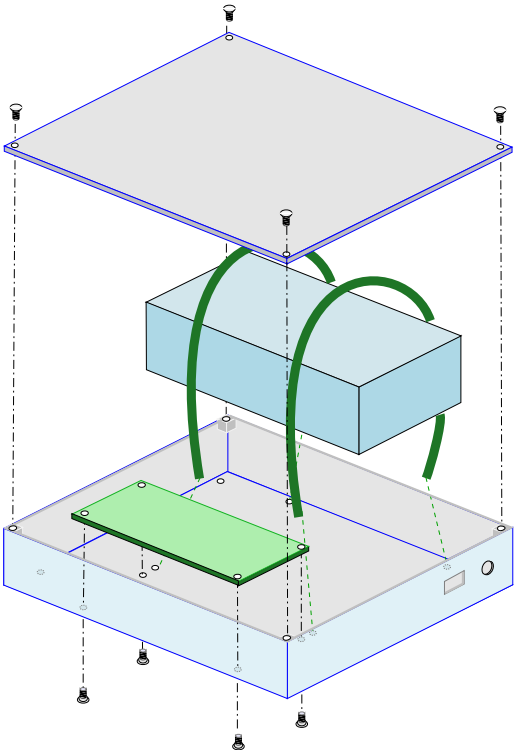

Figura 9. Montagem do Kit Nó TelosB na caixa plástica.

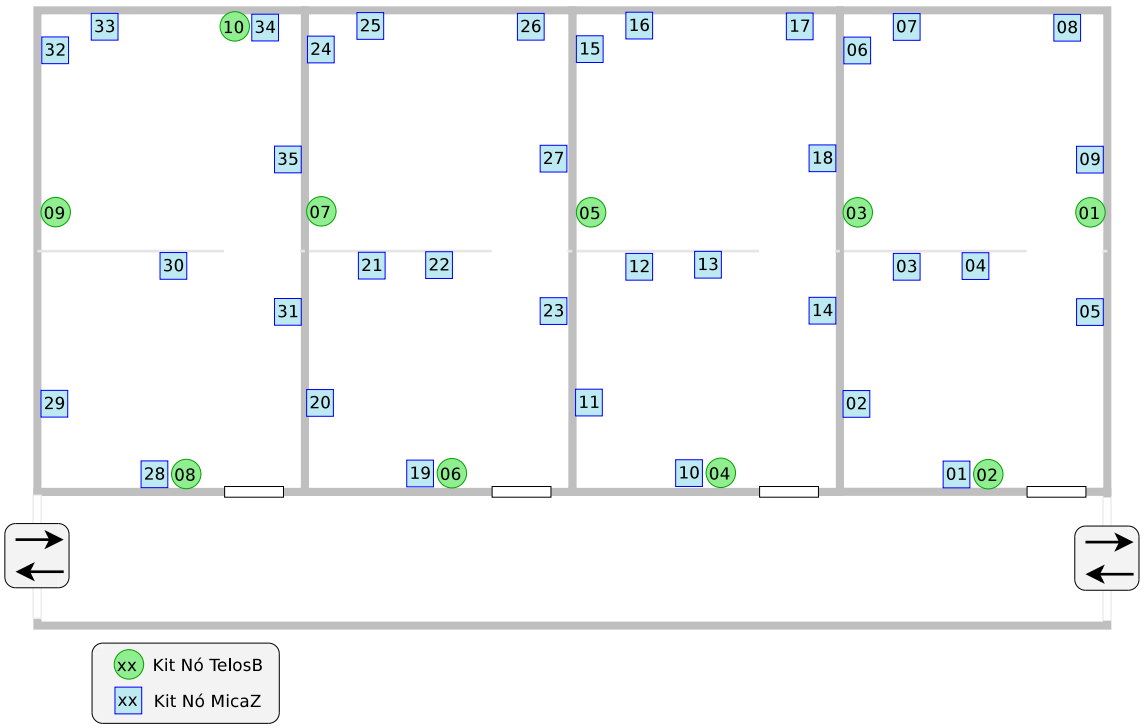

Figura 10. Distribuição física dos nós sensores.

- 1 Desktop com duas placas de redes e nobreak (*em aquisição com recursos deste projeto*).
- Pacotes e bibliotecas de software *open source* (sem necessidade de aquisição de

licenças).

3. Plano de testes

Realizaremos os testes para validação do protótipo em três etapas. Na primeira etapa de avaliação, um conjunto de aplicações básicas de redes de sensores sem fio, disponibilizadas nos ambientes TinyOS e Contiki, serão submetidas para execução no testbed. Essas aplicações de teste deverão contemplar o uso de diferentes topologias de rede e de recursos dos nós sensores (sensoreamento, processamento local e comunicação via rádio).

Na segunda etapa de avaliação, daremos ênfase a aplicações que requerem integração da aplicação executando na rede de sensores com a Internet. Essas aplicações usarão a pilha de protocolos IPv6 adaptada para redes LoWPAN.

Na terceira etapa de avaliação, o testbed será experimentado por desenvolvedores de aplicações para redes de sensores externos ao projeto. Nessa etapa, avaliaremos se o testbed é capaz de atender aos requisitos dos testes e avaliações projetados pelo desenvolvedor da aplicação. Todo o acesso ao testbed, incluindo a configuração dos experimentos, gerência da distribuição de tarefas entre os nós da rede, monitoramento e recebimento dos dados coletados deverão ser feitos via portal Web.

3.1. Aplicações básicas de redes de sensores sem fio

Na primeira etapa de testes, o objetivo será avaliar todas as funcionalidades do portal Web para acesso remoto ao testbed e recebimento dos logs de execução. Um conjunto de aplicações básicas para redes de sensores sem fio desenvolvidas nos ambientes TinyOS e Contiki serão selecionadas e executadas no testbed.

Avaliação Os sequintes aspectos serão avaliados nessa etapa de testes:

- Facilidade de uso da interface do portal;
- Funcionalidades providas pelo portal (controle de acesso, agendamento, configuração de experimentos, e execução);
- Operação dos nós do testbed (corretude, tempo para carga das aplicações e coleta dos resultados, interferências do meio).
- Interfaceamento do testbed com a máquina onde executa a aplicação do usuário, via interface serial.

3.2. Aplicações para demonstração de uso da pilha IPv6

Na segunda etapa do testes, o objetivo será demonstrar e avaliar a execução de aplicações para redes de sensores sem fio que fazem uso da pilha de protocolos IPv6. Duas aplicações alvo, baseadas em comunicação via IPv6, serão desenvolvidas para essa etapa de testes. Essas aplicações são descritas na subseções seguintes.

3.2.1. Aplicação *SmartHome*

No conceito de *casas inteligentes*, os equipamentos como lâmpada, TV, fogão, etc., podem se comunicar e com isso podemos remotamente controlar suas funções. Nossa

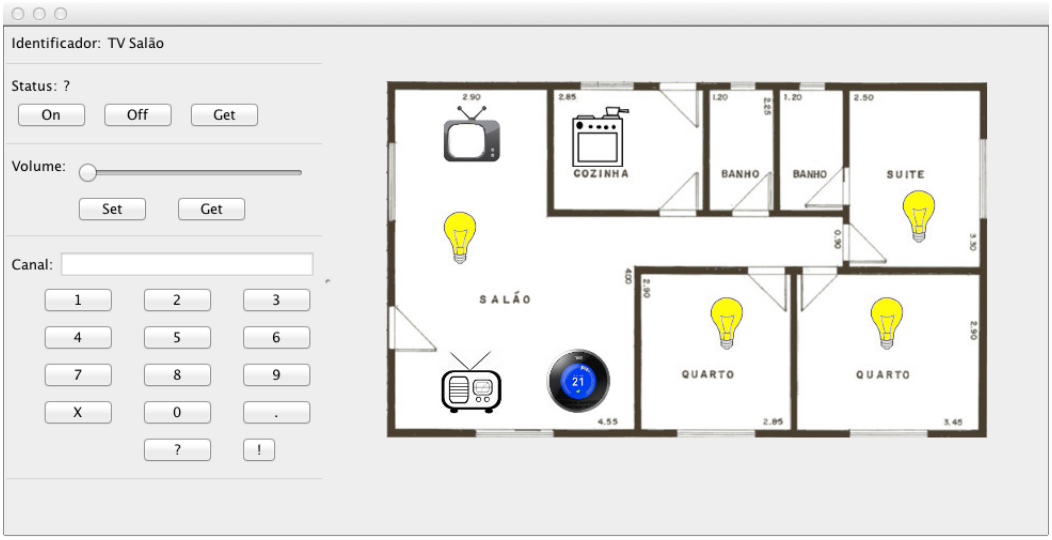

Figura 11. Interface Java para a interação com os equipamentos.

primeira aplicação considera cada um dos motes (nós da rede) como sendo o controlador de um desses equipamentos. De acordo com o tipo de dispositivo que o mote esteja controlando, ele será associado a um conjunto de operações que poderá realizar e um conjunto de informações que poderá prover.

Definimos cinco tipos de equipamentos que serão controlados. A seguir indicamos esses equipamentos e as informações recebidas e comandos que eles oferecem:

- Lâmpada
  - Informações: status (ligado/desligado).
  - Comandos: ligar/desligar.
- Termostato
  - Informações: status do sistema de climatização e a temperatura atual configurada.
  - Comandos: ligar/desligar, alterar da temperatura.
- Rádio
  - Informações: status do rádio, estação sintonizada e volume.
  - Comandos: ligar/desligar, alterar estação, alterar volume.
- TV
  - Informações: status do TV, canal sintonizado e volume.
  - Comandos: ligar/desligar, alterar canal, alterar volume.
- Fogão
  - Informações: status de cada boca e do forno, temperatura atual do forno.

Optamos por deixar o fogão sem atuação por questões de segurança.

Para interagir com cada um desses equipamentos, construiremos uma interface em Java, mostrada na Figura 11. Temos à direita da interface um mapa da casa, onde podemos encontrar a localização de cada um dos equipamentos. Ao clicar em um deles, o painel será modificado, apresentando uma interface de acordo com as operações que o equipamento selecionado poderá realizar.

A comunicação entre a interface e os equipamentos será feita por meio de pacotes UDP. Cada um dos equipamentos possuirá um endereço IPv6 conhecido pela interface e as mensagens serão roteadas entre si. Do ponto de vista da interface, não haverá nenhum tratamento especial, a comunicação será feita via UDP, como qualquer outra aplicação que envolveria PCs. O modelo de interação empregado será *Resquest/Response*, ou seja, os equipamentos sempre reagirão à interface, mas não iniciarão nenhuma comunicação.

Por questões de simplicidade, a parte final de atuação entre o mote e o equipamento controlado não será implementada. Ao invés disso, a realização dos comandos de atuação será simulada mantendo-se um estado interno para as informações (em vez de recuperá-las dos equipamentos). No entanto, isso não afeta o propósito da aplicação, que é de analisar aplicações IPv6 no testbed.

3.2.2. Aplicação *WaterMonitor*

Esta aplição tem o objetivo de realizar o monitoramento da água. Sensores seriam distribuídos ao longo de um rio, por exemplo, e algumas propriedades seriam medidas, como pH, oxigenação, condutividade, transparência, dentre outras.

A arquitetura desta aplicação difere da SmartHome. Neste caso estamos interessados no roteamento de mensagens. Dado que a extensão da área monitorada é bem maior que no caso anterior, os próprios motes deverão rotear a mensagem até a aplicação de coleta. Além disso, utilizaremos um modelo *Push*, onde os sensores enviarão periodicamente os dados captados para a aplicação de coleta em um PC.

Cada mote poderá fazer (ou simular) a leitura de diferentes propriedades da água e armazená-las internamente em um vetor. Depois de um determinado número de leituras, o mote enviará esse pacote para a aplicação de coleta. Como os dados serão enviados em pacotes UDP, implementaremos um esquema de ACK e retransmissão. Para isso, cada pacote possuirá um número serial que deverá ser reconhecido pela aplicação de coleta em um determinado tempo. Caso isso não ocorra, o mote retransmitirá o pacote. Isso deverá se repetir um número limitado de vezes, depois o mote assumirá que aquele pacote foi perdido e interromperá a sua retransmissão. Vale destacar que enquanto o processo de retransmissão ocorrer, o mote deverá continuar realizando leituras.

A aplicação de coleta está sendo implementada em Java (Figura 12), e o endereço IPv6 é conhecido por todos os motes da rede. Atualmente essa aplicação implementa apenas a lógica de enviar um ACK aos motes, e não realiza nenhum processamento com os dados recebidos. Isso nos permite realizar a depuração da aplicação e simular perda de pacotes. Como exemplo, essa aplicação poderia enviar os dados coletados para um banco de dados para posterior processamento.

Avaliação Os sequintes aspectos serão avaliados nessa etapa de testes:

- Interfaceamento do testbed com a máquina onde executa a aplicação do usuário, via protocolos IP;
- Tempo de resposta das aplicações (comunicação entre estação base e máquina do usuário);

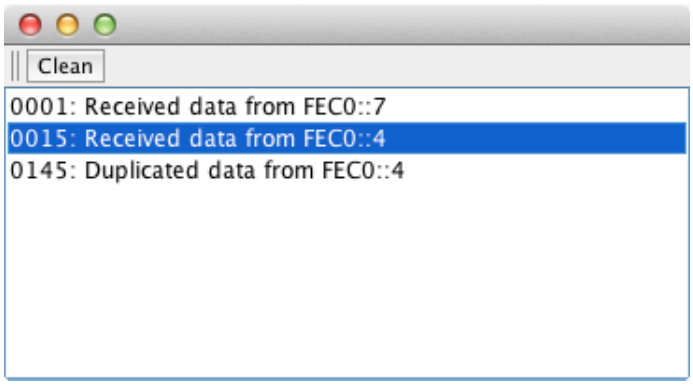

Figura 12. Interface Java para acompanhar a coleta de dados dos motes.

- Desempenho dos protocolos IPv6 dentro da rede de sensores: taxa de entrega dos pacotes, RTT, taxa de perda de pacotes, fragmentação de pacotes, auto-configuração das rotas, atribuição de endereços IPv6 aos nós sensores, escalabilidade e tolerância a falhas.

3.3. Acesso ao testbed por outros grupos de pesquisa

Na terceira etapa de testes, o objetivo será a avaliação geral do testbed por outros grupos de pesquisa no Brasil. Serão elaborados questionários para que os usuários externos avaliem o testbed.

Avaliação Os sequintes aspectos serão avaliados nessa etapa de testes:

- Facilidade de uso da interface do portal;
- Funcionalidades providas pelo portal (controle de acesso, agendamento, configuração de experimentos, e execução);
- Disponibilidade do portal e do testbed;
- Operação dos nós do testbed (tempo para carga das aplicações e coleta dos resultados);
- Interfaceamento do testbed com a máquina onde executa a aplicação do usuário, via interface serial e protocolos IP;
- Solução de controle de acesso federado ao testbed.

4. Cronograma de implantação e testes

1. Conclusão da implantação da segunda versão do testbed e portal de acesso (melhorias e correções da primeira versão): **3 a 28 de março**;
2. **Realização da primeira etapa de testes do protótipo: até 30 de abril**;
3. Projeto, implementação e avaliação de aplicações para demonstração de uso da pilha IPv6: **2 de abril a 30 de maio** (Atividade 8 - RP1);
4. **Realização da segunda etapa de testes do protótipo: até 30 de maio**;
5. Implementação e avaliação da solução de controle de acesso federado ao portal: **2 de abril a 30 de maio** (Atividade 9 - RP1);
6. Atualização da interface de operação do portal do testbed com melhorias e ajustes já identificados: **14 de abril a 30 de maio**;
7. **Realização da terceira etapa de testes do protótipo: até 30 de junho**;
8. Identificação e realização de correções e melhorias no protótipo antes da transferência de tecnologia: **de 16 de junho a 7 de julho**.

Referências

Werner-Allen, G., Swieskowski, P., and Welsh, M. (2005). Motelab: a wireless sensor network testbed. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, IPSN '05, Piscataway, NJ, USA. IEEE Press.

WISEBED (2008). Design of the hardware infrastructure, architecture of the software infrastructure & design of library of algorithms. Technical report, Collaborative project, Small and medium-scale focused research project (STREP).