BRENO AUGUSTO DIAS VITORINO

ARQUITETURA PARA CONSTRUÇÃO DE SOFTWARE ADAPTATIVO PARA REDES DE SENSORES

Belo Horizonte
25 de agosto de 2006

Livros Grátis

http://www.livrosgratis.com.br

Milhares de livros grátis para download.

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ARQUITETURA PARA CONSTRUÇÃO DE SOFTWARE ADAPTATIVO PARA REDES DE SENSORES

Dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

BRENO AUGUSTO DIAS VITORINO

Belo Horizonte
25 de agosto de 2006

UNIVERSIDADE FEDERAL DE MINAS GERAIS

FOLHA DE APROVAÇÃO

Arquitetura para construção de software adaptativo para redes de sensores

BRENO AUGUSTO DIAS VITORINO

Dissertação defendida e aprovada pela banca examinadora constituída por:

Dr. ANTÔNIO OTÁVIO FERNANDES – Orientador
Universidade Federal de Minas Gerais

Dr. CLAUDIONOR JOSÉ NUNES COELHO JÚNIOR
Universidade Federal de Minas Gerais

Dr^{a}. LINNYER BEATRYS RUIZ
Universidade Federal de Minas Gerais

Dr. WILTON SPEZIALI CALDAS
Universidade Federal de Minas Gerais

Belo Horizonte, 25 de agosto de 2006

Resumo

Este trabalho propõe uma abordagem sistemática sobre o uso da adaptação e redundância em redes de sensores sem fio, através de uma nova arquitetura para aplicações nessa plataforma. A rede de sensores surge como um novo paradigma à computação. Ela incorporar dados sobre o mundo físico, como temperatura, pressão e luminosidade, à infra-estrutura de *software* já existente. Essas novas oportunidades de explorar o mundo também trazem interessantes desafios. Entre eles, destaca-se a operação autonômica da rede, ou seja, de forma totalmente automática, sem intervenção humana. Para que isso se torne realidade, a aplicação deve ser adaptativa, no sentido em que o *software* altera seu fluxo de execução de forma a manter a qualidade do serviços prestados. A mudança de execução implica na utilização de um novo conjunto de dados, com maior qualidade, obtido através dos recursos redundantes disponíveis.

A arquitetura propõe a modelagem de uma aplicação para redes de sensores em planos de abstração, os quais correspondem a refinamentos sucessivos no processo de entendimento e organização dessa aplicação. Em cada um dos planos há uma estratégia de adaptação, a saber: assinalamento de papéis, adoção de alternativas à implementação de papéis, especificação de fluxos de execução e inclusão de algoritmos distribuídos. Especifica-se exatamente quando adotar cada estratégia durante o processo de modelagem, trazendo uma metodologia que inibe as técnicas *ad hoc* que vêm sendo geralmente empregadas no projeto e implementação de redes de sensores. Para fins de inspeção da qualidade dos serviços, há uma camada de dados de adaptação compartilhada entre os planos. Ela contém um conjunto de valores que permitem a avaliação da eficácia da adaptação corrente e das alternativas de

adaptação disponíveis. A viabilidade da arquitetura foi analisada em um estudo de caso, o qual envolveu a modelagem de uma aplicação típica para RSSFs.

Abstract

This work proposes a systematic approach for adaptation and redundancy employment in wireless sensor networks through a novel architecture. The sensor networks appear as a new computation paradigm. It allows the incorporation of physical data, like temperature, pressure, and light level, to existing software infrastructure. These new oportunities to explore the world also brings interesting challenges. One distinguishing challenge is the autonomic network operation. It means that the network has to work in a totally automatic way, with little or no human intervention. For such situation to become a reality, the application must be adaptive, in the sense that the software must change the execution flow to maintain its services' quality. The change in execution implies in the usage of a new data set, with greater quality, obtained from available redundant resources.

The architecture proposes the modeling of an application for sensor networks in abstraction layers, which correspond to successive refinements in the process of understading and organizing this application. In each layer there is a adaptation strategy, which are: role assignment, adoption of alternatives to the roles implementation, specification of execution flows, and inclusion of distributed algorithms. As we specify when and where to use each adaptation strategy during the modeling, we inhibit ad hoc techniques which are usually applied in the project and implementation of sensor networks. For the inspection of services' quality, there is a adaptation data layer shared with all other layers. It contains a set of values that allows the effectiveness' evaluation of both the current adaptation and its alternatives. The architecture viability was analyzed in a case study that involved the modeling of a typical sensor network application.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

Capítulo 1

Introdução

As redes de sensores sem fio (RSSFs) são um novo tipo de infra-estrutura computacional adequada a aplicações que requerem sensoriamento detalhado de um certo *campo de interesse* (CI). Campo de interesse, neste estudo, diz respeito às regiões geográficas nas quais os nós serão depositados, tais como florestas, prédios, linhas de manufatura. Os nós sensores, dispositivos que compõem essas redes, possuem tamanho reduzido (na escala de cm^3, atualmente) e utilizam-se de transceptores sem fio e baterias não-renováveis [Akyildiz et al., 2002].

Geralmente, as RSSFs devem operar sem a intervenção humana, de forma *autonômica*. Isso significa que elas devem se auto-configurar, assim que são implantadas, de acordo com os recursos de sensoriamento e comunicação disponíveis. Essa tarefa envolve, geralmente, a configuração automática da rede para a distribuição dos dados, no início de sua operação. Durante a operação, o objetivo da atividade autonômica é a manutenção da qualidade dos serviços da rede, de acordo com métricas, a exemplo de taxa de entrega, cobertura, precisão dos dados. Enfim, as redes continuamente monitoram os recursos disponíveis e executam alternativas, sempre que algum objetivo da aplicação não está sendo adequadamente cumprido. Essas políticas de adaptação são normalmente embutidas durante o desenvolvimento da aplicação, na qual o desenvolvedor já codifica os procedimentos de adaptação. Uma das principais causas da adaptação são as falhas, que devem ser contornadas para que a aplicação continue em funcionamento.

Nesse tipo de redes, *falhas* são uma certeza, e não exceção [Vieira et al., 2003]. Como as redes são de sensoriamento, estarão em contato direto com um ambiente não-controlado, e isso implica que esses nós sejam possivelmente afetados e, conseqüentemente, danificados. Condições atmosféricas, como chuva ou neblina, interferem na comunicação sem fio, altas temperaturas descalibram os sensores, obstáculos limitam a comunicação com nós vizinhos, e até mesmo nós podem ser destruídos completamente. Uma falha ainda intensamente investigada [Zhou e Roure, 2006] é o esgotamento da bateria do nó sensor, que o torna inútil. Assim, tanto o *hardware* como o *software* são projetados tendo em mente essa possibilidade de falha.

A adaptação necessária ao contorno de falhas e à operação autonômica é conseguida essencialmente através da utilização de *recursos redundantes*. Em uma RSSF, eles consistem tanto em *hardware* quanto em *software* redundante. No caso do *hardware*, isso significa que são dispostos na rede mais nós do que o mínimo necessário para sua operação. Esses nós, denominados nós reservas (*backup*), permanecem em estado inativo enquando a rede está funcionando normalmente. Eles são ativados para cobrir algum recurso de *hardware* que apresente perda de confiabilidade, como sensores imprecisos ou baixa conectividade de um nó sensor com a rede. Em termos de *software*, considera-se redundância quando há vários algoritmos que podem ser empregados para o mesmo fim. Eles se diferenciam no uso de energia, nos recursos de *hardware* utilizados e na confiabilidade que oferecem. Essa gama de algoritmos traz flexibilidade à rede para alterar o modo de execução, tornando-a, portanto, adaptável.

Os nós sensores, na perspectiva da rede, conferem à aplicação redundância temporal e espacial. Históricos de leituras dos sensores e do fluxo de mensagens ajudam a determinar o comportamento normal da rede e as ações mais adequadas frente a alguma mudança nesse modelo [Deshpande et al., 2004]. Em uma certa região do campo de interesse, um grupo de nós deve obter medidas similares do ambiente. Essas informações em conjunto são mais precisas e confiáveis do que as medidas de um único nó. Desse ponto de vista, uma estratégia comum é fornecer ao usuário da rede os dados de forma agregada por região.

Novos algoritmos adaptativos em contextos específicos [Heinzelman et al., 1999] e a reprogramação da RSSF em tempo de operação [Levis et al., 2004, Levis e Culler, 2002] são outras abordagens recorrentes para a adaptação. No primeiro caso, os algoritmos são projetados para que adaptem sua execução, utilizando algum recurso redundante. Entretanto, sua eficácia só é garantida em um cenário específico de operação da rede para o qual foram testados. Por outro lado, quando se trata da programação da RSSF, a literatura se volta à possibilidade de alterar o *software* da rede após sua implantação (vide seção 2.2). Com a rede já em funcionamento, esse é um recurso interessante, caso os requisitos da aplicação sejam alterados ou ocorra uma falha no *software*. Entretanto, ambas as abordagens não facilitam o projeto de uma rede autonômica. Neste trabalho, propomos uma abordagem diferente à adaptação. Nessa proposta, a aplicação pode conter vários algoritmos adaptativos e escolher entre a execução de um ou outro, cuja escolha estaria condicionada ao cenário corrente percebido pela rede.

Uma outra categoria de adaptações, também bastante estudada, são os algoritmos para otimização [Schurgers et al., 2002]. Devido ao imperativo da limitação de energia das RSSFs, todos os módulos de *hardware*, pilha de protocolos, modos de sensoriamento e, por fim, a aplicação como um todo, são desenhados para que tenham um consumo mínimo de energia.

Neste trabalho, consideradas raras exceções, os algoritmos de otimização serão tratados como adaptativos. Isso quer dizer que devem estar “auto-conscientes” do estado atual da rede para que tomem uma atitude, reativa ou pró-ativa, a fim de que os parâmetros determinados sejam, de fato, otimizados.

Diante do que foi exposto, a questão da adaptação deve ser tratada em pelo menos duas etapas: no projeto e na implementação da RSSF. No projeto, devem-se estabelecer: i) métricas de qualidade da aplicação; ii) adaptações necessárias; iii) recursos redundantes disponíveis; iv) possíveis cenários de operação da rede. Todos esses elementos se refletem na implementação do *software*, na qual o programador deve mapear essas necessidades na linguagem de programação aplicada.

No entanto, observa-se que são poucos os trabalhos que se ocupam da modelagem da adaptação no desenho e/ou na montagem de aplicação. Na prática, as aplicações para RSSFs são projetadas sem nenhuma metodologia específica para tratamento da redundância e da adaptação, ou seja, de forma *ad hoc*. Isso dificulta tanto o trabalho do projetista, que pode ter dificuldade em trazer esses requisitos não-funcionais à atividade de desenho, quanto o do desenvolvedor, que não dispõe de estruturas reutilizáveis de adaptação a sua disposição.

A figura 1.1, a seguir, apresenta uma visão ainda panorâmica do modelo e seus planos, os quais serão devidamente explicitados no capítulo 3.

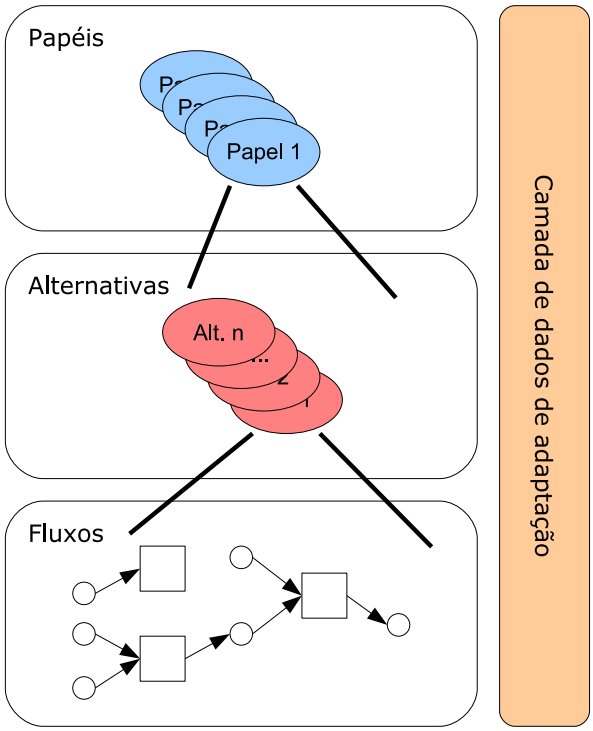

Figura 1.1: Estrutura geral do modelo proposto.

Na figura acima, que apresenta a estrutura geral do modelo exposto, uma aplicação deve ser entendida, em um primeiro momento, como um conjunto de papéis que cada nó exercerá na rede. Cada papel é constituído por um conjunto de alternativas, as quais equivalem a alternativas de execução dos papéis. Por fim, cada alternativa está associada a um conjunto de fluxos de execução, os quais detalham os recursos redundantes, algoritmos e adaptações necessárias à implementação da alternativa.

Este trabalho tem, como base, a metodologia para tolerância a falhas para robôs

proposta por Caldas [Caldas, 2004]. Nesse trabalho citado anteriormente, assim como neste estudo, a aplicação é modelada em planos de abstração. A cada plano, são utilizadas adaptações e recursos redundantes próprios, bem como há um grafo que determina o comportamento da aplicação. Quanto mais vértices o grafo contiver, mais alternativas a aplicação possuirá e, dessa forma, haverá mais opções de adaptação.

A perspectiva da metodologia proposta é em relação a um nó da rede. A construção do modelo explicita quais as interações do nó com a rede, quais as políticas de adaptação cada nó pode adotar, quais recursos redundantes serão utilizados em cada adaptação e quais os impactos da adaptação nos parâmetros de qualidade do nó. A metodologia favorece ao projetista a percepção, tanto dos recursos redundantes locais ao nó quanto dos recursos vizinhos a esse nó. Dessa forma, a metodologia contribui para a utilização de políticas e algoritmos distribuídos, naturalmente mais escaláveis do que aqueles que interagem com dados globais da rede.

No plano dos papéis, a aplicação é pensada de acordo com os papéis que um nó pode exercer na rede. O papel do nó determina como ele contribui, em um dado momento da operação da rede, no objetivo da aplicação como um todo. Por exemplo, nós reservas e nós ativos são dois papéis distintos, representados por dois vértices no grafo de papéis. As transições entre vértices condicionam a mudança de papéis de um nó de acordo com o estado da aplicação.

Cada papel possui alternativas que exibem quais as opções que o nó tem para cumprir um determinado papel na rede. Elas são escolhas de adaptação que o nó tem à sua disposição, caso a alternativa corrente não satisfaça às métricas de qualidade da aplicação. Cada alternativa apresenta dois parâmetros, confiabilidade e custo em energia, ambos utilizados na decisão para realizar a transição entre elas. No modelo, as alternativas são representadas por vértices em um grafo em um plano próprio. Cada aresta desse grafo indica quais transições entre alternativas são possíveis, e está associada a uma função de transição que realiza quaisquer operações necessárias à adaptação da aplicação entre uma e outra alternativa.

Para cada alternativa, há os fluxos de execução da aplicação. Eles contêm os recur-

sos de *hardware* e os dados utilizados (representados por círculos), bem como os módulos de *software* que atuam sobre esses dados (representados por quadrados). Os fluxos modelam efetivamente os recursos de *hardware* e *software* utilizados na execução de uma alternativa. É possível, nesse fluxo, associar um mesmo dado a mais de um módulo ou produzir um mesmo tipo de dado a partir de vários módulos. Associado a esse fluxo, existem algoritmos de adaptação e algoritmos de otimização, cuja função é garantir a confiabilidade e o consumo de energia propostos à alternativa. Para isso, eles podem interferir na execução dos nós vizinhos, no caso dos algoritmos de adaptação, ou no próprio fluxo de execução, no caso dos algoritmos de otimização.

Os planos, além de estarem associados a diferentes níveis de abstração da aplicação, exibem, também, uma hierarquia de adaptações. As transições entre papéis possuem prioridade máxima, e são efetivadas assim que se façam necessárias. As transições entre alternativas são avaliadas em seguida, e, por fim, são avaliados os algoritmos de adaptação e otimização da alternativa corrente. Dessa forma, as decisões implementadas nos níveis superiores de abstração possuem prioridade sobre as decisões dos níveis inferiores. Essa organização reforça que as adaptações nos níveis superiores correspondem a decisões mais gerais, que são tomadas para atingir os objetivos principais da aplicação, enquanto as decisões nos níveis inferiores se referem à manutenção da qualidade dos serviços da rede.

1.1 Objetivos

Este trabalho tem como objetivo propor uma metodologia para modelagem de aplicações para redes de sensores sem fio (RSSFs), que incluam a utilização dos recursos redundantes da rede e de algoritmos adaptativos. A partir desse modelo, será possível a avaliação desses algoritmos, quando utilizados em conjunto em uma aplicação não simulada em diversificados campos de interesse (CIs). Espera-se contribuir para que projetistas e desenvolvedores estejam aptos a gerenciar, com competência, a complexidade dessas aplicações, que necessariamente atuam de forma autonômica. Isso somente se dá no momento em que a adaptação é tratada de forma sistemática.

Até o final dos anos 90, as redes de computadores dependiam essencialmente de administradores que garantiam sua disponibilidade. Mas a tendência contemporânea nos sugere que a forma tradicional de administração de redes tende a se tornar obsoleta, na ocasião em que os elementos computacionais poderão estar em todos os lugares, na forma de nanorobôs, celulares, iPods, todos interconectados, formando uma imensa rede de informação. Segundo Horn [Horn, 2001], uma opção, nesse contexto, são as redes autonômicas, que gerenciam automaticamente os elementos de rede depositados em larga escala, em ambientes não-estruturados.

Ao utilizar a metodologia proposta, a experiência de arquiteturas aplicadas em projetos anteriores de sucesso poderá ser reaproveitada mais facilmente, visto que o uso de modelos formais registra as decisões de qualquer projeto.

1.2 Contribuições

A metodologia para aplicações adaptativas proposta neste trabalho oferece as seguintes contribuições:

- Uma abordagem sistemática sobre a questão da adaptação em tempo real das aplicações para RSSFs, facilitando a incorporação deste aspecto às aplicações.

- Auxílio na compreensão da implementação da aplicação, exibindo todas as interações entre as adaptações e explicitando os recursos redundantes alocados.

- Conhecimento do desempenho de diferentes algoritmos adaptativos em uma mesma aplicação, considerando-se que, em geral, os algoritmos são avaliados apenas isoladamente.

- Investigação da questão da adaptação em vários aspectos, desde o projeto à implementação das aplicações. Dessa forma, pode-se conhecer as abordagens existentes e como estas são aplicáveis em cada estágio do desenvolvimento da aplicação.

- Uma organização do uso de diferentes arcabouços de adaptação. Cada camada da arquitetura proposta aborda o uso da adaptação em um contexto específico. Essas camadas são amarradas na mesma arquitetura, e, por conseqüência, na mesma aplicação. Ao especificar como se dá a interação entre essas camadas, vislumbra-se uma maneira de abordar os diferentes tipos de adaptação em uma mesma aplicação.

1.3 Organização do texto

No capítulo 1, introdutório, são apresentados brevemente alguns conceitos-chave para a realização deste estudo, a saber: redes de sensores sem fio (RSSFs); campos de interesse, adaptação, redundância, tolerância a falhas. Destaco a seguir, apenas alguns desses conceitos, considerando-se que os demais serão abordados, mais cuidadosamente, nos capítulos seguintes. Sobre as RSSFs, pode-se afirmar que elas devem operar de forma *autonômica*. Quanto à operação autonômica, pode-se afirmar que é obtida, sobretudo, com a utilização de *recursos redundantes*. Considerados esses princípios, a adaptação deve ser tratada em pelo menos duas etapas: no projeto e na implementação da RSSF. Os objetivos deste trabalho e possíveis contribuições também foram abordados neste capítulo, nas seções 1.1 e 1.2, respectivamente.

No capítulo 2, são retomados e discutidos conceitos fundamentais já apontados no capítulo anterior. Em primeiro lugar, conceitos gerais sobre RSSFs são discutidos, sobretudo, quanto a suas aplicações típicas e às plataformas de *hardware* empregadas. Afirma-se, também, que as aplicações para RSSFs só serão empregadas de forma mais ampla, a partir do uso extensivo da adaptação, na verdade, uma ferramenta básica para construção da função autonômica. Levando-se em conta esse pressuposto, analisa-se a adaptação nas diversas formas que ela assume, desde o projeto até a operação da rede. Isso implica, por exemplo, emprego de algoritmos para o controle de topologia, otimização dos recursos consumidos da rede.

Já no capítulo 3, será explicitada a arquitetura de adaptação que se propõe, neste trabalho: os principais elementos, a organização, bem como o modo de interação para que

a aplicação seja, de fato, adaptativa. Destacam-se, em especial, reflexões sobre a camada de dados de adaptação, um detalhamento sobre cada um dos três planos de abstração oferecidos pela arquitetura. Finalmente, é apresentado o emprego da arquitetura na modelagem de uma aplicação real, como prova de conceito.

Por fim, no capítulo 4, de caráter conclusivo, são retomados os conceitos básicos discutidos ao longo do estudo. Avaliam-se, então, os ganhos reais da aplicação da arquitetura em aplicações autonômicas. Finalizando este capítulo, são apresentadas possíveis e novas frentes de pesquisa que se abrem, a partir do que se discutiu, ao longo de todo o texto.

Capítulo 2

Adaptação e redundância em redes de sensores

Neste capítulo, são discutidos conceitos-chave para a realização deste estudo, a saber: redes de sensores sem fio, adaptação, redundância, tolerância a falhas. Abordam-se, inicialmente, conceitos gerais sobre RSSFs, quanto às suas aplicações típicas e plataformas de *hardware* empregadas. Observa-se, então, que as aplicações para RSSFs só serão empregadas em larga escala quando houver o uso extensivo da adaptação, importante ferramental para construção da função autonômica. Considerado esse pressuposto, investiga-se a adaptação nas diversas formas que ela toma, desde o projeto até a operação da rede.

2.1 Redes de sensores sem fio

Uma rede de sensores sem fio (RSSF) vem sendo caracterizada por um conjunto de dispositivos em operação, denominados nós sensores, dispostos em um certo campo de interesse (CI). Conforme já anunciado no capítulo 1, ao se tratar de CIs, focalizam-se as regiões geográficas nas quais os nós serão dispostos, a fim de registrar informações sobre fenômenos relevantes que podem ocorrer nesse espaço. Essas informações, associadas ao instante e local em que ocorreram, são nomeadas, neste trabalho, como eventos de interesse (EIs).

RSSFs já vêm sendo empregadas, ainda que de forma prototípica, em diferentes tipos de aplicações: monitoramento ambiental [Mainwaring et al., 2002], rastreamento de animais [Juang et al., 2002], monitoração de ecossistemas em microescala [Culler e Mulder, 2004], vigilância de áreas de segurança [He et al., 2004], rastreamento de contêineres em linhas de manufatura [Clauberg, 2004], agricultura de precisão [Burrell et al., 2004], detecção de elementos químicos tóxicos [Ong et al., 2000], entre outras.

Em uma RSSF, como concebida atualmente, os nós sensores coletam dados sobre o CI e os enviam, em última instância, a um nó chamado ponto de acesso (PA). O PA tem o papel de interligar as redes de sensores às redes tradicionais, nas quais os dados podem ser armazenados e analisados. A figura 2.1, seguinte, ilustra esse cenário.

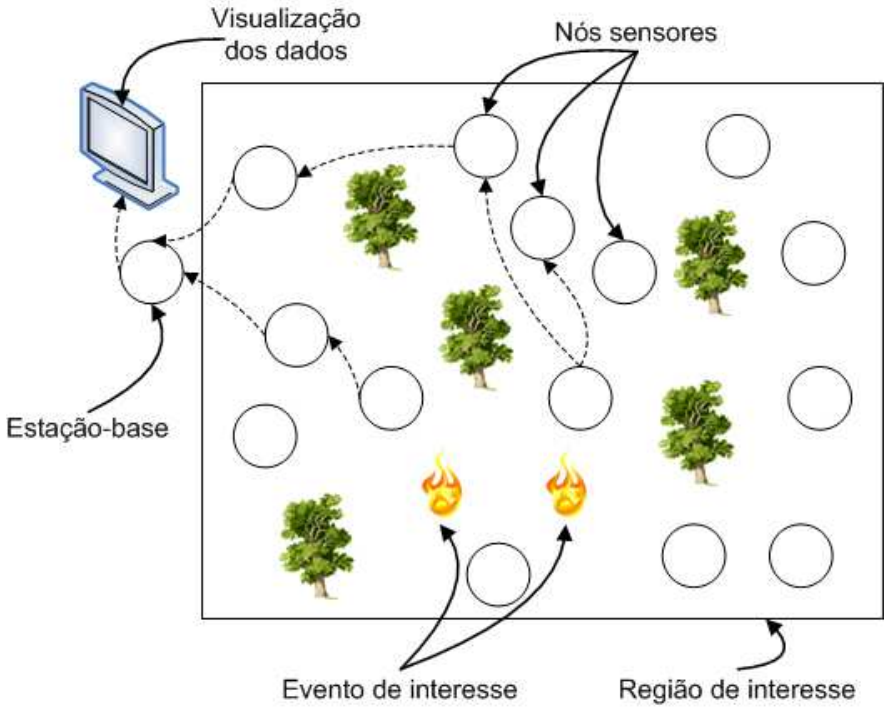

Figura 2.1: Esquema de uma rede de sensores típica, neste exemplo, usada no alerta a incêndios. As setas tracejadas mostram o fluxo de dados da rede no caso de detecção de um evento de interesse (EI).

Em uma aplicação de detecção de incêndios, conforme ilustrado acima, vários nós sensores munidos de termistores são distribuídos a fim de emitir um alerta em caso de um foco de incêndio. Na maior parte do tempo, espera-se que os nós sensores façam amostragens de temperatura em intervalos na ordem de minutos. Além disso, eles devem manter

informações sobre a qualidade dos enlaces de comunicação dos nós vizinhos e dos dados amostrados. Na detecção de um EI, neste caso, um foco de incêndio, os nós imediatamente transmitem essa informação ao PA, por meio de uma comunicação multi-saltos, tal como representado na figura 2.1 pelas linhas tracejadas. Quando o PA recebe o alerta dos nós, ele comunica esse fato a um sistema externo, conectado à Internet, que, por sua vez, dispara o alerta de incêndio, mostrando a localização e extensão desse EI.

De acordo com as características dos CIs, propomos agrupá-los em *controlados* e *não-controlados*. Controlados são aqueles CIs em que há uma infra-estrutura que mantém o ambiente em condições físicas constantes. Já os não-controlados são aqueles em que os nós são dispostos em um ambiente natural, em condições físicas imprevisíveis, portanto. Tradicionalmente, as redes são implantadas em ambientes controlados, que fornecem elementos favoráveis à sua correta operação. Nesse caso, os cenários não-controlados surgem como um importante desafio ao planejamento e à correta operação das aplicações. Outro fator a se ressaltar é que os nós sensores, assim como outros dispositivos que se utilizam de sensoriamento (robôs, por exemplo), possuem uma intrínseca relação com o ambiente no qual estão dispostos.

Nos CIs controlados, há uma série de elementos que auxiliam o planejamento da RSSF. No caso de uma linha de manufatura, a localização dos nós sensores pode ser cuidadosamente estudada para que a cobertura da rede seja otimizada. Da mesma forma, os nós sensores podem ser preparados para o CI em que estarão expostos, utilizando-se, por exemplo, de invólucros com a resistência adequada às condições físicas do local de deposição, de sensores que possuam sensibilidade mais apropriada ao EI. Eles, também, podem ser trocados e ter suas baterias renovadas, em caso de falha ou de esgotamento da fonte da energia.

Num CI não-controlado, como, por exemplo, na agricultura, os nós, obviamente, estarão sujeitos às intempéries, que, por sua natureza, são inesperadas quanto ao instante em que surgem, quanto ao espaço de abrangência, quanto à gravidade e duração. Isso dificulta o projeto da rede, visto que as possibilidades de interação dessas intempéries com a RSSF são

inúmeras. Geralmente, nessa classe de CIs, há ainda o problema da dificuldade de acesso ao ambiente pelo homem, onde não é possível fazer uma deposição manual dos nós. Esse contexto torna inviável a manutenção manual ou a reposição de nós falhos. Os cenários controlados e não-controlados, portanto, influenciam na forma de como atacar os desafios para a implantação das aplicações, sobretudo, a adaptação e a robustez.

Cada dispositivo de uma RSSF é chamado nó sensor. Um nó sensor, de acordo com Ruiz [Ruiz, 2003], possui cinco componentes ou módulos básicos, a saber: sensoriamento, processamento, comunicação, energia e memória, ilustrados na figura 2.2, que se segue.

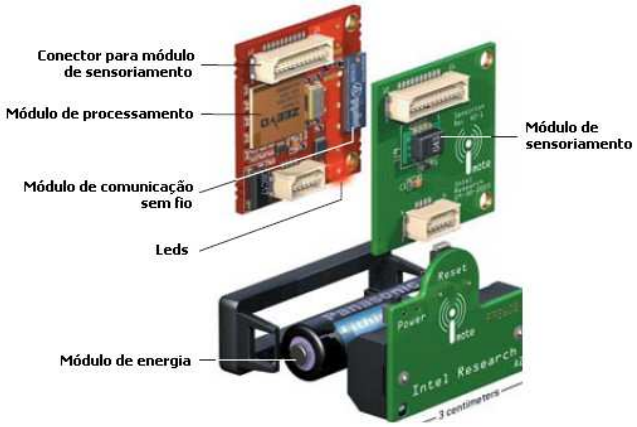

Figura 2.2: Módulos do nó sensor iMote, da família Motes de Berkeley. Fonte: [Culler e Mulder, 2004].

Na realidade, a organização do *hardware*, mostrada na figura 2.2, é o senso comum para a maioria dos nós sensores existentes, por exemplo, a família de nós sensores Mica Motes [Hill e Culler, 2001], EYES [van Hoesel et al., 2003], μAMPS [Finchelstein, 2005] e BEAN [Vieira, 2004]. Os módulos básicos de um nó sensor serão brevemente caracterizados a seguir. Mais detalhes sobre esses módulos podem ser encontrados no trabalho de Vieira [Vieira, 2004].

Sensoriamento São sensores com baixo consumo de energia e dimensões reduzidas, geralmente, agrupados a uma placa de sensores. Essa placa é conectada ao nó sensor através de um barramento específico. Exemplos de sensores típicos adotados são: termistor, acelerômetro, microfone, fotômetro, barômetro.

Processamento É um microcontrolador de baixo consumo de energia, como o MSP430F149 [Texas Instruments, 2000] ou o ATMEGA128L [ATMEL, 2003]. Ele dispõe de conversores analógicos-digitais, que interligam com os sensores. Entretanto, FPGAs e microprocessadores não projetados para baixo consumo de energia podem ser utilizados, no caso de nós com maior poder de processamento e energia [Bellis et al., 2005].

Comunicação São um ou mais transceptores que realizam a troca de mensagens pelo meio sem fio. Tipicamente, utiliza a freqüência na faixa de 900 MHz ou 2.4GHz, denominada banda ISM (*Industrial, scientifical and medical band*), não licenciada para equipamentos comerciais. Comumente, os transceptores empregam as tecnologias de radiofreqüência ou, mais recentemente, UWB (*UltraWide Band*) [Vieira et al., 2005].

Energia É obtida por meio de baterias não-renováveis do tipo linear simples ou de lítio de baixa capacidade. O fato de não serem renováveis impõe aos nós sensores um limite de tempo de operação, que pode ser ampliado por algoritmos mais eficientes no consumo de energia. Há estudos para a utilização de energia renovável, como a mecânica ou a solar [Finchelstein, 2005, Voigt et al., 2003].

Memória É um dispositivo (de memória) *flash* acoplado ao nó, servindo como um repositório de dados persistentes. Sua capacidade está na ordem de dezenas ou centenas de *kilobytes*.

Como os nós sensores possuem tamanho reduzido e não há fios, tanto para distribuição de energia quanto para comunicação, eles apresentam oportunidades de aplicações até então inviáveis. É possível distribuí-los em um ecossistema para que monitorem as condições climáticas (temperatura, pressão, umidade) por meses, sem interferir nesse ambiente. Isso é importante em ecossistemas frágeis nos quais qualquer presença humana pode até aumentar a taxa de mortalidade dos animais ali presentes [Mainwaring et al., 2002]. Em aplicações industriais, podemos implantá-los nos insumos recebidos pela fábrica e rastreá-los pela linha de produção. Em indústrias farmacêuticas, nas quais os insumos devem ser rastreados para que não sejam extraviados ou descartados incorretamente no ambiente, essa aplicação é bas-

tante promissora. No campo militar, as RSSFs podem ser usadas para rastrear inimigos ou detectar o espalhamento de elementos químicos tóxicos [Ong et al., 2000].

Ao mesmo tempo, as RSSFs trazem desafios que, em conjunto, são inéditos para os projetistas de aplicações. Os principais desafios são: a restrição de recursos de energia, processamento e memória; a comunicação não-confiável; a coordenação distribuída; a segurança dos dados; a tolerância a falhas. Os nós sensores possuem um *hardware* bastante limitado, pois estes devem ser de baixo custo e de tamanho reduzido. Há a necessidade da deposição de centenas ou milhares de nós em uma única aplicação. Ao mesmo tempo, a projeção da indústria é que o custo desses nós chegue a menos de um dólar americano por unidade. Esses nós devem ser pouco intrusivos, para que a interferência da RSSF nos EIs seja mínima. Por exemplo, um nó sensor típico, o MICA2, possui apenas $5.7 * 3.18 * 0.64$ cm de volume.

Uma importante limitação dos nós sensores se deve ao consumo de energia. Como utiliza uma fonte não-renovável de energia, seus componentes devem ser bastante econômicos quanto ao uso desse recurso. Voltando ao exemplo do Mica2, ele é alimentado por duas pilhas AA e todos os seus componentes consomem na ordem de miliamperes, quando ativos, e microamperes, quando em modo econômico. Se se mantiverem ativos o tempo todo, a RSSF pode operar apenas durante cinco dias, aproximadamente [Mainwaring et al., 2002]. Portanto, uma aplicação típica para RSSF deve manter os componentes em estado econômico na maior parte do tempo. Todos os *softwares* que executam nos nós sensores, como a pilha de protocolos, o controle da taxa de sensoriamento, os algoritmos distribuídos e a própria aplicação, devem levar em conta esse fator.

2.2 Adaptação: conceitos essenciais

Na maior parte das aplicações, um importante desafio é implantar um modelo de operação autonômico, ou seja, que possa operar corretamente sem intervenção humana. Esse conceito, apresentado, inicialmente, em [Horn, 2001], surge da crescente complexidade dos sistemas computacionais, cuja manutenção se torna inviável. Ele propõe o projeto de siste-

mas computacionais auto-gerenciáveis, que possam se ajustar e trabalhar de forma autônoma. Um exemplo clássico de sistema autonômico é o sistema nervoso do corpo humano, o qual é capaz de ajustar os batimentos cardíacos e o ritmo respiratório, sem a plena consciência do indivíduo. No caso das RSSFs, desenvolver aplicações autonômicas significa que estas devem interagir com os nós vizinhos, detectar e recuperar falhas e otimizar o uso dos recursos da rede automaticamente.

Uma característica essencial de uma aplicação autonômica, portanto, é que ela seja adaptável. Isso significa que ela deve continuar a funcionar, de acordo com métricas de qualidade pré-estabelecidas, mesmo em presença de falhas de *hardware* ou *software* ou alterações bruscas nos CIs. A adaptação deve ser pensada durante o planejamento da aplicação de RSSF, considerando-se a operação normal da rede, as situações imprevisíveis que poderão ocorrer e as limitações e suposições que hajam sobre a rede. Nesse momento, o projetista pode modificar *hardware* e/ou *software* livremente para atingir os objetivos da aplicação, incluindo os mecanismos de adaptação necessários.

Neste trabalho, a adaptação é abordada com base nos trabalhos de Avancha [Avancha, 2005] e Cerpa e Estrin [Cerpa e Estrin, 2004]. Nesse contexto, a adaptação é vista como a capacidade do *software* para alterar seu modo de execução, a fim de manter uma certa métrica de qualidade. Avancha [Avancha, 2005] define métricas de qualidade, de acordo com três aspectos do nó sensor: comunicação, sensoriamento, energia. Essas métricas, associadas a regras booleanas, avaliam a situação da rede e disparam ações adaptativas. Cerpa e Estrin [Cerpa e Estrin, 2004], por sua vez, propõem um algoritmo de controle de topologia que ativa nós reservas, sempre que o número de nós vizinhos está abaixo de um limiar ou a taxa de perda de mensagens está alta.

O conceito de adaptação apresentado neste trabalho, portanto, não contempla os trabalhos sobre *hardware* reprogramável [Weng et al., 2004] ou reprogramação da aplicação [Levis e Culler, 2002, Levis et al., 2004]. Ao adaptar o *hardware*, é possível ajustar o poder de processamento do nó, de acordo com a tarefa a ser desempenhada. Esse fato é relevante, principalmente para funções computacionalmente mais caras, tal como processamento de

imagem. Já a reprogramação do *software* é interessante para consertar um defeito de software ou reconfigurar a rede, em caso de mudanças de requisitos durante sua operação. Em ambos os casos, há um dispêndio em energia, recurso escasso, para a reconfiguração, que poderia ser melhor empregado nos objetivos da aplicação. De qualquer forma, esses tipos de reprogramações não solucionam o desafio da adaptação como definida neste trabalho, a qual se configura automaticamente, de acordo com métricas de qualidade obtidas em tempo real.

A adaptação, de acordo com Avancha [Avancha, 2005], possui três fases ou componentes, a saber: i) avaliação das métricas de qualidade (MQs); ii) escolha da ação adaptativa adequada; iii) execução da ação.

As MQs são valores obtidos sobre o nível de qualidade dos serviços prestados por um nó ou grupo de nós. As MQs dizem respeito ao estado do *hardware* e ao estado da aplicação em si. O estado do *hardware* refere-se às condições de operação dos módulos do nó sensor, enquanto o estado da aplicação refere-se a métricas definidas pela própria aplicação, portanto, particulares a cada aplicação. Como as MQs referem-se a informações obtidas pela inspeção do próprio nó ou um grupo limitado de nós, são escaláveis em relação ao tamanho total da rede.

O nó sensor é capaz de monitorar o uso e a eficiência dos seus módulos. Componentes que lidem com recursos restritos, tal como o de energia, são monitorados, a fim de se determinar a taxa de consumo e a quantidade restante do recurso. Os demais módulos, por sua vez, têm sua eficiência e precisão como métricas de qualidade. Isso vale, por exemplo, para os módulos de sensoriamento e comunicação. Avancha [Avancha, 2005], em seu trabalho taxonômico, define métricas essenciais, considerando-se o estado do *hardware*. A tabela 2.1, na página seguinte, exibe as MQs mais importantes propostas por esse autor.

Módulos	**Parâmetros**
Energia	Capacidade energética restante (CER), taxa de consumo de energia (TCE)
Comunicação (camada física)	Potência recebida do canal (PRC), Potência recebida de ruído (PRR), Taxa de perda da portadora (TPP), Taxa de violação do formato do quadro (TVQ), Taxa de falhas no cabeçalho do quadro (TFC)
Comunicação (camada de enlace)	Taxa de transmissões falhas (TTF), Taxa de múltiplas retransmissões (TMR), Taxa de colisões (TC), Taxa de falhas na seqüência de frames (TFS)
Comunicação (camada de roteamento)	Número de enlaces comprometidos (NEC), Número de roteadores favoráveis (NRF)
Sensoriamento	Precisão do sensor (PS), modo de operação (MO)

Tabela 2.1: Parâmetros de qualidade quanto ao estado do *hardware* [Avancha, 2005].

Vale ressaltar que a obtenção das MQs é um esforço conjunto entre *hardware* e *software*. O *hardware* deve fornecer meios para que seus componentes sejam inspecionados pelo *software*. Exemplos desse esforço conjunto são o nível de sinal recebido na transmissão sem fio (RSSI - *Received Signal Strength Indication*) e a voltagem corrente da bateria. Ambos são importantes para a determinação da qualidade do enlace de comunicação e da energia restante na bateria, respectivamente. Se o *hardware* não oferecer esses recursos de inspeção, ainda assim, algumas MQs podem ser inferidas. As inferências podem ser feitas por meio da inspeção dos nós vizinhos ou de modelos bayesianos, os quais podem determinar o comportamento esperado de um módulo.

As MQs a respeito do estado da aplicação são específicas para cada aplicação. Como é mostrado na seção 2.5, essas MQs são estabelecidas a partir dos requisitos levantados para a aplicação. Algumas, entretanto, são recorrentes em várias aplicações. Essas são descritas na tabela 2.2 que se segue.

Métrica de Qualidade	**Descrição**
Latência	Intervalo de tempo entre a ocorrência de um EI e a sua recepção no PA.
Cobertura	Porcentagem da área monitorada em relação à área total do CI.
Precisão dos dados	Quantidade de erro existente em uma medição de um EI.
EIs detectados	Porcentagem de EIs percebidos pela RSSF em relação ao total de EIs que de fato ocorreram. Na literatura, essa MQ dirige os algoritmos que tratam do *problema da exposição*.
Disponibilidade	Porcentagem do tempo total de operação em que a RSSF está operando corretamente.

Tabela 2.2: Métricas de qualidade quanto ao estado da aplicação.

É importante saber que todas as MQs listadas anteriormente são relativas à qualidade da aplicação percebida pelo usuário, enquanto que as MQs relativas ao estado do *hardware* são percebidas por um nó ou grupo de nós. Ambas as métricas são importantes, visto que a adaptação deve ser feita em todos os níveis, desde um único nó até à aplicação como um todo. Essas MQs são obtidas através de um conjunto combinado de técnicas de inferência e inspeção da rede, por meio de nós líderes responsáveis por um grupo de nós e/ou algoritmos distribuídos que trabalham com dados locais.

Conseguidas as medidas dos parâmetros de adaptação, o próximo passo é avaliá-las. A avaliação consiste em verificar se os valores estão de acordo com os limiares adequados, valendo-se de expressões booleanas. Os limiares e expressões são formulados graças aos requisitos da aplicação e a simulações. As expressões de avaliação são implantadas no *software*, que, em intervalos regulares, realiza a computação das avaliações. Ao verificar que uma avaliação é negativa, o *software* parte para a escolha da adaptação a ser tomada, a fim de que a avaliação torne-se novamente positiva.

A escolha da adaptação talvez seja a fase mais complexa, visto que há poucos trabalhos no sentido de se estabelecer uma metodologia nesse contexto. A escolha envolve o custo em realizar a adaptação e a possível eficiência desta sobre as MQs. Nesse sentido, Marrón et al. [Marrón et al., 2005a] propõem que essas escolhas sejam realizadas de acordo com três dimensões, dispostas em um cubo: características instantâneas da rede, objetivos da aplicação e parâmetros de otimização. Caldas [Caldas, 2004] escolhe a adaptação

com maior índice de ganho, ou seja, aquela que será mais efetiva naquele instante. O índice de ganho é calculado a partir das previsões do nível de confiança e desempenho que a adaptação trará à aplicação. Geralmente, as escolhas são definidas em um contexto específico, como o de cobertura de redes, como mostra a máquina de estados finitos do ASCENT [Cerpa e Estrin, 2004], ilustrada na figura 2.3.

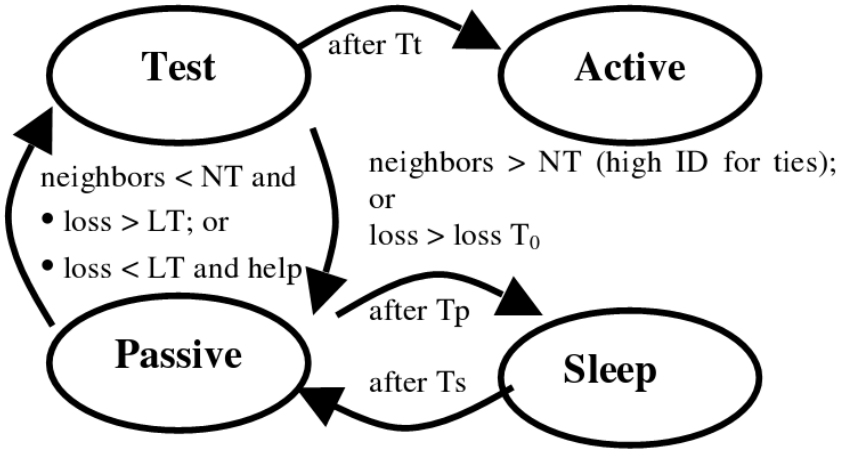

Figura 2.3: Grafo de transições do algoritmo ASCENT. Fonte: [Cerpa e Estrin, 2004].

Na figura acima, as avaliações, escolhas e ações são totalmente especificadas através de uma máquina de estados finitos. Cada ação é especificada por um estado, e as transições entre estados são as escolhas possíveis. Cada escolha é ativada pela avaliação de duas MQs: número de nós vizinhos (NT - *Neighborhood Threshold*) e perda de mensagens (LT - *Loss Threshold*. Os limiares para esses parâmetros, neste trabalho, foram estabelecidos de forma arbitrária. A escolha das ações, nesse caso, não envolve um cálculo de ganho da adaptação, ela está apenas condicionada à violação de alguma das avaliações.

Por fim, há a execução da ação de adaptação escolhida. Isso pode envolver o disparo de ações nos nós vizinhos, e a adequação dos demais módulos de *software* ativos à ação. Nesse sentido, é importante que um módulo de *software* seja reposto por outro com um impacto mínimo. Isso só é alcançado através de interfaces bem definidas entre os módulos. Para o aspecto de roteamento da RSSF, He et al. [He et al., 2003] organiza-o em três módulos: encaminhamento de pacotes, coleta de estado e informação de estado. A informação de estado são as variáveis sobre a vizinhança, o roteamento e o próprio nó, que são arma-

zenadas em cada nó e devem ser escaláveis com o tamanho da rede. O encaminhamento de pacotes e a coleta de estado são módulos programáveis, encapsulando as políticas de decisão de roteamento e a transmissão das mensagens de controle que constróem e mantêm as rotas. Uma ação de adaptação, nesse caso, significa a mudança da estratégia de implementação de qualquer um dos módulos programáveis. Como os módulos possuem funcionalidade e interface bem definidas, o impacto dessas adaptações é mínimo, considerando-se que há o reaproveitamento do estado da rede. Essa, também, é a técnica adotada no projeto TinyCubus [Marrón et al., 2005a].

2.3 Adaptação e redundância

Ao analisar a implementação das adaptações, observa-se o uso obrigatório da redundância. Por redundância, entende-se como sendo a capacidade de se obter a mesma informação por diferentes meios. Cada um desses meios possui sua confiabilidade dos dados e seu custo para obtê-los. A adaptação, portanto, escolhe uma dessas fontes de dados, de acordo com seu objetivo. Por exemplo, se o objetivo corrente for aumentar a confiabilidade da informação, será escolhido o meio mais confiável para se obter uma dada informação, mesmo a um maior custo em energia.

Existem três formas de redundância: temporal, espacial e de informação. Em RSSFs, a redundância temporal se refere ao histórico dos dados coletados e do comportamento da rede durante um certo período. A redundância espacial, por sua vez, refere-se aos nós sensores vizinhos. Denominam-se nós sensores vizinhos aqueles que apresentam uma interseção na área de sensoriamento ou na área de comunicação com um nó sensor de referência. Ao contrário dos sistemas computacionais tradicionais, nos quais a redundância espacial implica replicação de *hardware* e aumento indesejado do peso e custo do dispositivo, nas RSSFs os nós sensores têm baixo custo e são dispostos, redundantemente, em uma certa região. Logo, a redundância espacial nas RSSFs é um caso comum e deve ser explorada. Por fim, a redundância de informação ocorre, por exemplo, quando um nó sensor agregador recebe os dados coletados por vários outros nós. A redundância de informação ocorre, também, na pilha de

protocolos, com o uso de códigos de erro para recuperação de uma mensagem corrompida.

Um exemplo de redundância temporal é exibido na figura 2.4, abaixo.

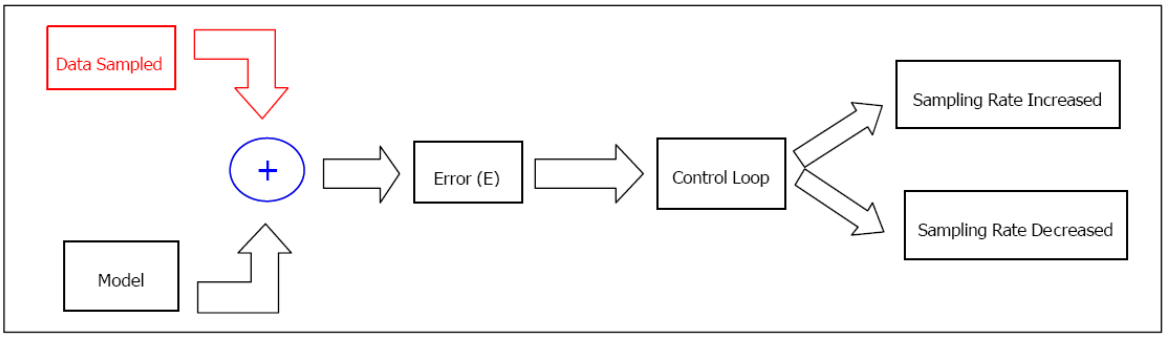

Figura 2.4: Redundância do sensor, considerando os dados obtidos e um modelo do ambiente. Fonte: [Jain e Chang, 2004].

Essa figura representa o esquema de obtenção dos dados de sensoriamento. Observe que esses dados são obtidos através do sensores em tempo real e, ao mesmo tempo, através de um modelo construído a partir da observação temporal do comportamento do ambiente. Como há duas fontes de dados distintas para a mesma informação, o algoritmo adaptativo determina a confiabilidade de cada fonte e envia a informação que seja mais confiável. A redundância temporal, por traçar uma estimativa do dados reais nos momentos atual e futuro da rede, serve de comparação com os dados obtidos pelos sensores, verificando se estes não estão muito discrepantes daqueles. Desse modo, ela contribui para o disparo de uma ação adaptativa, por exemplo, a recalibração do sensor ou a verificação do modelo diante do comportamento do ambiente percebido pelos nós sensores vizinhos. Modelos analíticos podem ser empregados em outras situações, por exemplo, para a localização geográfica de um EI [Koushanfar et al., 2002]. Utiliza-se, então, as leis de Newton, o espaço euclidiano e as leis trigonométricas para estabelecer relações com as coordenadas trianguladas pelos sensores.

Quanto à redundância espacial, o que se observa na literatura são trabalhos no sentido de controle de topologia [Quintão et al., 2004, Liu e Li, 2003, Chen et al., 2001] e cobertura da rede [Carbunar et al., 2004, Tian e Georganas, 2002]. Em todos eles, nós redundantes são desligados para manter um mínimo de nós ativos que possam garantir o sensoria-

mento e a comunicação dos dados. Isso porque, se todos os nós da rede se mantiverem ativos, haverá uma implosão na quantidade de dados e mensagens que serão transmitidas, levando a um baixo desempenho e consumo excessivo de energia. Portanto, em RSSFs, é importante controlar a redundância espacial, de forma que ela auxilie em ações adaptativas, provendo rotas alternativas de comunicação e redundância de sensoriamento e, ao mesmo, não provoque problemas como descrito anteriormente [Coman et al., 2005]. Gao et al. [Gao et al., 2003] mostraram analiticamente que, para que haja 90% de redundância de sensoriamento, são suficientes cinco nós sensores vizinhos.

A redundância de informação, por sua vez, é abordada na literatura por técnicas avançadas de agregação de dados. A agregação consiste em realizar a fusão de dois ou mais fluxos de dados recebidos por nós sensores. Essa fusão é realizada usando desde algoritmos simples como a média ou mediana das leituras, ou técnicas mais sofisticadas que levam em conta, por exemplo, a suposição que um número máximo f de sensores podem falhar [Marzullo, 1989] apud [Jacob e Mathai, 2004]. Neste caso, espera-se como entrada os intervalos máximo e mínimo das leituras de cada sensor, incluindo aí a incerteza de cada sensor. Em seguida, o algoritmo destaca as interseções de intervalos que contêm pelo menos $n - f$ intervalos, no qual n é o número total de intervalos. Dessas interseções, determina-se o intervalo final (agregado) como o menor intervalo que contenha todos os intervalos destacados no passo anterior. Para melhorar o intervalo final, Iyengar et al. [Iyengar et al., 1994] apud [Jacob e Mathai, 2004] propõem atribuir um valor de confiabilidade a cada um dos intervalos destacados, e depois aplicar uma regra customizável para selecionar apenas aqueles intervalos com maior confiabilidade.

2.4 Adaptação e tolerância a falhas

Neste trabalho, a tolerância a falhas é considerada como uma das causas de uma adaptação. Define-se tolerância a falhas como a capacidade de um sistema fornecer as suas saídas corretas, mesmo na presença de defeitos e falhas [Caldas, 2004]. Em RSSFs, a tolerância a falhas é aplicada principalmente nos módulos que interagem diretamente com o

ambiente, tal como os sensores e os transceptores. Isso porque esse tipo de interação geralmente introduz erros consideráveis e até os danifica por completo.

Quanto ao sensoriamento, o método mais comum para tolerância a falhas é a correlação dos dados de um nó sensor com o de seus vizinhos. Essa correlação, que comumente se traduz na agregação de dados, visa trazer mais confiança às leituras dos sensores em uma região [Deshpande et al., 2005], dado que os nós sensores nessa região farão aproximadamente a mesma leitura.

Krishnamachari e Iyengar [Krishnamachari e Iyengar, 2003] realizaram uma investigação analítica sobre o nível de confiança proporcionado pela redundância espacial dos sensores. Seu trabalho concluiu que uma leitura de um sensor está correta se pelo menos metade das leituras dos nós vizinhos sejam similares à sua. Para encontrar esse resultados, eles supuseram que a probabilidade de falhas p dos sensores é simétrica e não-correlacionada e que os eventos de interesse são determinados por um simples limiar definido pelo usuário da aplicação. Uma falha no sensor pode ser detectada se os demais sensores também estão falhando . Por exemplo, na aplicação de monitoração de aves *Petrel* na ilha *Great Duck* [Szewczyk et al., 2004], uma baixa leitura de umidade estava altamente correlacionada com a falha de toda a placa de sensores de um nó.

Para a agregação de dados, é necessário realizar o agrupamento de nós por região. Cada região contém um nó lider, o qual recebe os dados dos demais nós e realiza a agregação dos dados vindouros. Há várias funções de agregação, de acordo com a seção 2.3. Após agregar os dados, o nó lider envia o dado agregado ao PA, por transmissão direta ou repassando por outros nós líderes [Habib et al., 2004]. Observa-se, na figura 2.5 da página seguinte, uma configuração de rede como descrita anteriormente.

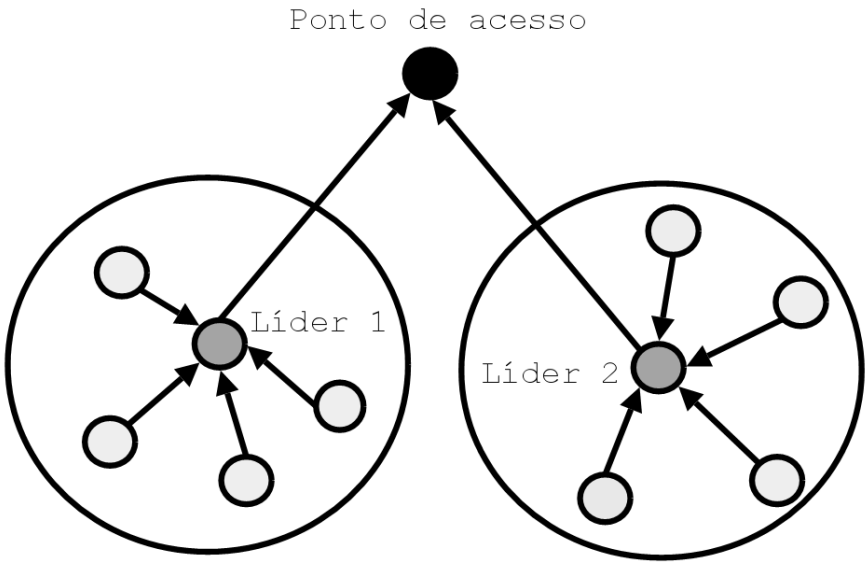

Figura 2.5: Exemplo de coleta hierárquica de dados por uma RSSF.

A topologia da rede explicitada na figura anterior, em forma de árvore, é a tradicional para redes hierárquicas. No entanto, é importante ressaltar que, se o nó líder falhar enquanto os nós enviam seus dados para este, haverá perda de informações, que não alcancarão o PA, e consumo desnecessário de energia. A partir dessa observação, Nath et al. [Nath et al., 2004] desacopla a topologia à agregação, aliando um algoritmo de roteamento por múltiplas rotas a um framework que inibe a contagem duplicada de leituras na agregação. Esse trabalho alia, portanto, a tolerância a falhas na comunicação e no sensoriamento.

O aspecto de tolerância a falhas na comunicação envolve todas as camadas da pilha de protocolos da RSSF. Nas camadas física e de enlace, são usadas as técnicas tradicionais: tratamento de colisão, códigos CRC e retransmissões, aliando-se a um desenho para eficiência em energia [Sadler et al., 2005]. Já na camada de roteamento, como descrito anteriormente, são usadas múltiplas rotas para a entrega da mesma mensagem. Esse procedimento pode ainda ser melhorando, adicionando-se a escolha de rotas que possuam maior qualidade dos enlaces [Woo et al., 2003] apud [Macedo, 2006]. Desse modo, consegue-se tanto evitar o custo excessivo em energia ao transmitir por vários caminhos, enquanto evita falhas provocadas por desconexão de rotas. Entretanto, pode haver um custo extra em energia, necessário para aumentar a confiabilidade das rotas. Essa situação ocorre quando muitos enlaces estão

ruins, o que leva ao uso de um maior número de rotas adicionais. A reconstrução periódica de rotas e a monitoração ativa dos enlaces também são usadas na tolerância a falhas, como empregado no protocolo PROC [Macedo, 2006].

2.5 Adaptação e projeto de aplicações para RSSFs

Como as RSSFs são muito recentes, ainda há poucas aplicações em campo e muitas delas ainda são protótipos experimentais. Dessa forma, grupos de pesquisa por todo o mundo têm estudado questões específicas dessas redes: segurança, roteamento de dados, design de *hardware*, modelos de programação, entre outros. Porém, o desafio atual é reunir as soluções propostas nos trabalhos literários da área na implantação de aplicações, tarefa dificultada pela falta de integração dos algoritmos e métodos propostos até então. Esta seção apresenta, de forma breve, uma metodologia proposta por Ruiz [Ruiz, 2003] para especificação de uma RSSF na qual essa integração é fundamental.

Para cada fase da metodologia, discutem-se algumas possibilidades de adaptação da RSSF frente às informações coletadas durante a especificação. Essas são apenas sugestões de adaptações, para exemplificar qual deve ser o pensamento do projetista em cada fase. Elas devem ser incrementadas com informações específicas da aplicação e novos estudos que estão sendo publicados.

Ruiz [Ruiz, 2003] sugere a concepção de uma aplicação para RSSFs, segundo um conjunto de sete fases: levantamento de requisitos, planejamento, programação dos nós, deposição dos nós, configuração dos nós (*nodes setup*), início do funcionamento da rede (*network bootup*) e operação da rede.

No levantamento de requisitos, o projetista deve levantar todas as características da aplicação, sem, no entanto, fazer escolhas quanto ao *software* e/ou *hardware* que serão utilizados. Esse é o momento de descrever completamente o problema a ser resolvido, estabelecer o(s) objetivo(s) da aplicação, comparar com soluções alternativas às RSSFs, analisar o comportamento previsto para o ambiente e para o(s) evento(s) a serem monitorado(s), determinar a topografia do ambiente, e até investigar se há restrições legais e ambientais quanto

à instalação da RSSF, em caso de ambientes naturais, por exemplo. Essas questões definem a especificação de requisitos da aplicação. Essa especificação irá guiar todas as decisões quanto ao número de nós e o *hardware* e *software* utilizados em cada nó e, em conseqüência, ao custo da rede em termos financeiros.

Essa documentação da aplicação, por si só, já restringe e/ou mostra oportunidades de adaptação da rede. Se o evento de interesse provavelmente ocorrer em certo período do dia, a aplicação deve, nesse período, se empenhar mais na detecção do evento, enquanto em outros momentos deve trabalhar em um modo econômico de energia. O tipo de evento determina quais sensores devem ser usados, e o nível de precisão da medição dirige as escolhas quanto ao nível de redundância necessário. É importante determinar se há algum tipo de correlação temporal ou espacial no evento. No caso de detecção de incêndios, no qual o evento se alastra a partir do foco inicial, todos os nós sensores vizinhos ao evento devem estar prontos para repassar os dados do incêndio e determinar a velocidade em que o incêndio está crescendo, antes que os nós sejam destruídos pelo fogo. Quanto mais elaborada for essa fase, mais otimizações e oportunidades de adaptação surgirão nas próximas fases.

No planejamento da rede, definem-se os serviços da aplicação, número e tipos de nós necessários e a pilha de protocolos, os quais irão compreender as fases seguintes. Nesta fase, o projetista deve decidir sobre os aspectos de implementação da rede, de acordo com o que foi levantado na especificação de requisitos. Já existem propostas de soluções, tanto em *hardware* como em *software*, para a maioria dos serviços necessários aos diversos tipos de RSSFs. Cabe ao projetista escolher a avaliar essas soluções em vista das especificidades da sua aplicação.

Alguns serviços são importantes na maioria das RSSFs, entre eles: configuração de parâmetros, controle de densidade, definição da área de cobertura e monitoramento da energia residual da rede. Configuração de parâmetros se refere à redefinição de parâmetros da rede e dos sensores durante a operação da rede. Esses parâmetros podem regular os temporizadores dos nós, o tamanho de agrupamentos (*clusters*) de colaboração entre nós, os limiares para detecção de eventos de interesse, entre outros. O controle de densidade refere-

se ao acionamento e/ou desligamento de nós, de forma a evitar um congestionamento no canal de comunicação ou excesso de redundância de sensoriamento, mantendo a cobertura da área de interesse. Por cobertura, entende-se, tanto a capacidade de monitoramento de toda a região de interesse quanto a manutenção de enlaces de comunicação com toda a rede. Portanto, a definição da área de cobertura é relevante ao informar se a cobertura está sendo mantida durante toda a operação da rede. Alguns serviços, como o monitoramento da energia residual da rede, também auxiliam na detecção de futuras falhas na cobertura. Ao mostrar quanto resta de energia em cada nó, o mapa de energia residual permite à aplicação se adaptar a futuras falhas na cobertura, ativando nós reservas e delegando menos tarefas aos nós com pouca energia [Mini et al., 2005].

No final do planejamento, antes de partir para a implementação da aplicação, é interessante construir um protótipo ou uma simulação. Devido à natureza de sistema embutido dos nós sensores e à natureza da própria aplicação, por ser distribuída, torna-se complicada a depuração da aplicação em campo. Ferramentas, como o TOSSIM [Levis et al., 2003], proporcionam simulações nas quais é possível analisar o comportamento da aplicação diante de um ambiente controlado. Isso significa determinar o número de nós da simulação, a conectividade entre eles, o posicionamento dos nós, a probabilidade de erro na transmissão de mensagens, o número de conversores analógico-digitais (ADCs) de cada nó (simulando os sensores). Usando o ambiente de scripting TYTHON [Demmer et al., 2005] aliado ao TOSSIM, é possível controlar variáveis do ambiente, ou seja, o número, o instante e o tipo de EIs que ocorrerão durante o tempo de simulação. Quanto ao tipo, o TYTHON permite eventos pontuais ou graduais. Eventos pontuais são aqueles detectados em um única localização do ambiente. Já os eventos graduais são aqueles detectados por um conjunto de sensores em uma região e sua intensidade percebida é inversamente proporcional à distância entre o nó e o evento. O TOSSIM executa simulações do mesmo código executável que estará em operação em campo, possibilitando, assim, uma depuração mais precisa da aplicação final.

O uso de simulações permite avaliar a correção dos algoritmos distribuídos, o comportamento dos nós diante dos EIs e a adaptabilidade da aplicação em vista de falhas nos nós

sensores. Como o ambiente e os nós são totalmente controlados, é possível determinar em qual instante a aplicação falha, deixando de perceber um evento de interesse ou atingindo estados indesejáveis, como um *deadlock*. Falhas injetadas controladamente nos nós tornam possível a análise de falhas em diferentes cenários, e, por conseqüência, uma visão analítica do nível de adaptabilidade da aplicação. Enfim, os estudos das simulações permitem identificar dificuldades que devem ser repensadas e contornadas, geralmente exigindo um replanejamento da aplicação, antes que esta possa ser viabilizada para ir a campo.

Ao decidir sobre o número de nós sensores a serem utilizados, deve-se levar em conta o nível de redundância e adaptação desejados à aplicação. Em uma RSSF, a redundância se baseia principalmente na quantidade de nós na própria rede. Portanto, deve ser incluídos mais nós do que o mínimo necessário para a operação normal da rede, os quais assumiram o papel de nós reservas. Os serviços presentes na RSSF irão acionar ou desligar esses nós automaticamente, de acordo com as métricas de qualidade de operação.

Na fase de programação de nós, supõe-se que os serviços já tenham sido codificados com o suporte de um sistema operacional específico para os nós sensores, tais como TinyOS [Hill et al., 2000] ou MANTIS [Abrach et al., 2003]. Esse código será transferido manualmente para os nós sensores por uma placa de programação, que se conectará a um PC comum por uma porta serial, paralela, USB ou Ethernet. Outra alternativa é reprogramar os nós, apesar das desvantagens já discutidas na seção 2.2. Após essa fase, os nós devem ser dispostos no ambiente-alvo, de forma aleatória ou de acordo com diretrizes adotadas na fase de planejamento [Nakamura, 2003].

As próximas fases, configuração, início de funcionamento e operação da rede, irão transcorrer conforme os serviços determinados na fase de planejamento. Na fase de configuração, os nós verificam o funcionamento de sensores e transceptores, executando funções de auto-teste. O serviço de auto-localização [de Oliveira, 2005], se houver, também é executado para determinar as posições geográficas de cada nó, em relação ao ambiente ou em relação aos nós vizinhos.

No ínicio de funcionamento da rede, os nós devem formar as rotas de comunicação.

Se a topologia da rede for hierárquica com agrupamentos, estes serão formados. Também, são ligados os sensores adequados, iniciando-se a coleta dos dados. Na operação da rede, ocorre o funcionamento rotineiro da RSSF. Há o sensoriamento, processamento e disseminação dos dados. Os serviços da aplicação, que gerenciam a densidade, a energia, a área de cobertura, a topologia, atuam paralelamente às atividades normais da rede para que os parâmetros de qualidade de serviço, confiabilidade e disponibilidade sejam mantidos durante essa fase.

Tendo em vista essa metodologia, conclui-se que a adaptação da RSSF é uma característica a ser definida e alcançada durante o planejamento da aplicação. Ela permeia todo o *software* e *hardware* utilizados, devido à interação contínua da aplicação com o ambiente, geralmente inóspito, a qual expõe a rede às intempéries. Ainda não há técnicas padronizadas para se alcançar a adaptação, que é totalmente dependente dos objetivos da RSSF e de seus requisitos. Há propostas de algoritmos adaptativos que são adequados em situações específicas, e expõem ao projetista soluções adaptativas possíveis nessas situações.

2.6 Adaptação e modelos de programação

O problema da adaptação se extende à implementação da aplicação, na medida em que há uma lacuna (*gap*) semântica entre os requisitos não-funcionais levantados durante o projeto das RSSFs (vide seção 2.5) e as linguagens de programação, *frameworks* e módulos de *software* disponíveis para RSSF. Como será abordado nesta seção, são bem recentes (cerca de cinco anos) os trabalhos que se ocupam de discutir metodologias que incluam o aspecto de adaptação, tão essencial às RSSFs, para o entender do desenvolvimento de *software*.

No que se refere a tolerância a falhas em sistemas robóticos, que se assemelham às RSSFs por sua interação com o ambiente, há uma importante abordagem chamada ALLIANCE [Parker, 1998]. Ela se refere a uma arquitetura de *software* para facilitar a incorporação de adaptação e tolerância a falhas em times de robôs cooperativos. Cada robô possui um conjunto de tarefas e sabe o desempenho esperado de cada uma. Durante sua operação, cada robô monitora a atividade realizada pelos vizinhos, por meio dos sensores ou de mensa-

gens trocadas entre eles. Há dois parâmetros essenciais para o disparo de ações adaptativas: impaciência e aquiescência. A impaciência diz respeito às falhas externas percebidas pelos robôs, enquanto a aquiescência é um parâmetro relativo às falhas do próprio robô. O fator impaciência aumenta quando um outro robô está cumprindo sua tarefa com baixo desempenho. Quando esse fator ultrapassa um limite pré-estabelecido, o robô passa a executar a tarefa do outro. Já a aquiescência aumenta quando o robô percebe que sua própria tarefa está com baixo desempenho, fator este que o leva a deixar essa tarefa e escolher outra.

Avancha [Avancha, 2005] define parâmetros de qualidade dos nós sensores, individualmente e em grupos (*clusters*), a fim de reconhecer situações de falha na aquisição dos dados, de comunicação e de segurança da rede como um todo. A proposta é que cada nó monitore seus componentes de energia, comunicação e sensoriamento continuamente. Caso haja alguma recuperação ou deterioração do serviço desses módulos, o nó deve reportar-se a um nó sensor rico em recursos (*resource-rich node*), que funciona como o líder de um agrupamento de nós sensores comuns. O líder, então, agrupa os dados de estado de cada nó, formando parâmetros de qualidade a respeito do agrupamento como um todo. Através de regras, envolvendo esses novos parâmetros, o líder ativa adaptações do agrupamento, enviando comandos para os nós sensores para que estes alterem seu estado. Por exemplo, se o agrupamento possuir um nível de energia normal e um elemento químico nocivo for detectado pela maioria dos nós, o líder pára de agregar os dados sensoriados com o objetivo de fornecer à estação-base informações mais precisas sobre o evento de interesse detectado. A adaptação pode ser feita, também, por cada nó, considerando apenas as informações locais. Desse modo, o nó sensor pode, por exemplo, aumentar o raio de alcance do rádio, caso a desconexão do nó seja iminente e o nível de energia seja normal [Habib et al., 2004].

O projeto TinyCubus [Marrón et al., 2005a] propõe um arcabouço para desenvolvimento de aplicações adaptáveis para RSSFs. O *framework* divide a aplicação em três partes: o arcabouço inter-camadas (*Tiny Cross-Layer Framework*), a *engine* de configuração (Tiny Configuration Engine) e o arcabouço de gerenciamento de dados (*Tiny Data Management Framework*). O arcabouço inter-camadas provê um repositório de dados compartilhado por

todas as camadas da aplicação. Dessa forma, as interfaces de cada módulo são ligadas aos dados que eles manipulam, e não à implementação das mesmas, permitindo que haja reconfiguração, em tempo de execução, do módulo que implementa a interface. A *engine* de configuração, por sua vez, permite a distribuição de novo código na rede, em tempo de operação, similar às abordagens de reprogramação dos nós. Por fim, o arcabouço de gerenciamento de dados determina quais os melhores módulos a serem executados em um dado momento, considerando-se as características instantâneas da rede, os objetivos da aplicação e os parâmetros de otimização.

Capítulo 3

Uma arquitetura de adaptação para redes de sensores

Neste capítulo, será explicitada a arquitetura de adaptação proposta nesta dissertação. Na seção 3.1, serão vistos os principais elementos, como são organizados e como eles interagem para que a aplicação seja efetivamente adaptativa. A seção 3.2 aborda a camada de dados de adaptação, na qual estão contidos os dados que dirigem as adaptações em toda a arquitetura. Na seção 3.3, detalha-se cada um dos três planos de abstração oferecidos pela arquitetura. Por fim, a seção 3.4 apresenta o emprego da arquitetura na modelagem de uma aplicação real, como prova de conceito.

3.1 Visão geral

A arquitetura de adaptação, desenvolvida neste trabalho, tem por objetivo trazer uma nova metodologia para o tratamento da adaptação em RSSFs. Baseando-se nos trabalhos de Caldas [Caldas, 2004], Marrón et al. [Marrón et al., 2005a] e Avancha [Avancha, 2005], a arquitetura propõe a modelagem da aplicação em camadas de abstração. Cada camada contém um grafo, o qual modela todas as adaptações planejadas. As adaptações compreendem ações que um nó executa individualmente, de acordo com variáveis monitoradas, a fim de adequar-se a falhas, otimizar o uso de recursos e/ou aplicar regras da própria aplicação. As

adaptações têm um foco tanto individual quanto coletivo, no sentido de que agem sobre os problemas detectados, tanto no nó sensor quanto na vizinhança desse nó. Essa arquitetura tem como entrada as adaptações planejadas à aplicação e, como saída, um modelo da aplicação adaptativa, tal como ilustra a figura 3.1.

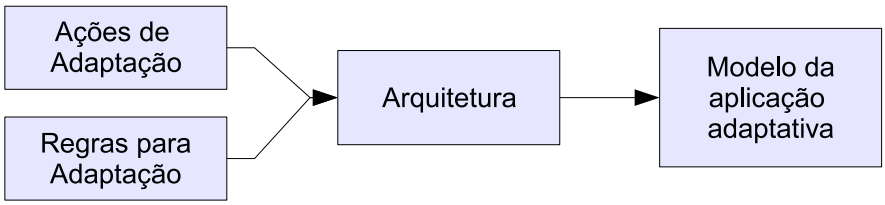

Figura 3.1: Insumos e produtos da arquitetura proposta

Segundo a figura acima, existem dois insumos da arquitetura: as regras para adaptação (RpAs) e as ações de adaptação (AdAs). As RpAs instruem à aplicação quais condições, do ambiente e da própria rede, que necessitam de AdAs. Essas, por sua vez, compreendem as modificações no comportamento do nó para que este efetue a adaptação. A AdA, portanto, especifica quais módulos de *software* estarão ativos em um nó, durante e após a adaptação. As RpAs e AdAs são distribuídas na arquitetura entre os planos de abstração, que, por fim, tomam a forma de vértices, arestas e atributos de um grafo que compõe a saída da arquitetura.

Os grafos obtidos na saída revelam o modelo de execução da aplicação. Em um dado instante da execução da aplicação, é possível associar todo o conjunto de *softwares* em execução a um vértice (ou a um subgrafo, quando se tratando dos fluxos de execução). Esse vértice, ou subgrafo, indica quais políticas de adaptação estão sendo tomadas naquele instante. Ao mesmo tempo, as RpAs estão sendo continuamente avaliadas. Se a avaliação sugerir a necessidade de adaptação, uma RpA levará a uma AdA que esteja conectada ao vértice corrente, nos planos de abstração superiores, e à seleção de um novo subgrafo, no plano inferior. O mecanismo de avaliação das RpAs se encontra embutido em cada plano de abstração.

A arquitetura de adaptação, como brevemente explanada no capítulo 1, possui três planos de abstração: papéis, alternativas e fluxos. Cada plano corresponde a um nível de abstração da aplicação, na qual se implantará o aspecto de adaptação. Dessa forma, espera-se

que a modelagem da aplicação na arquitetura faça com que o projetista incorpore a abstração de forma homogênea, ou seja, durante todo o processo de modelagem. Os planos de abstração, também, permitem que as políticas de adaptação sejam refinadas no mesmo ritmo em que se refina a própria aplicação. Mais ainda, o projetista poderá dividir as preocupações e responsabilidades sobre adaptação entre os três planos de abstração existentes.

Como premissa, assumiu-se que a aplicação não fará uso de algoritmos e/ou módulos que possuam informação global da rede. Apesar desse tipo de informação proporcionar o uso de algoritmos mais otimizados, ela nem sempre estará disponível. Geralmente, para se obter informação global, é necessário fazer uma requisição a esse dado a todos os nós da rede, o que ocasiona uma explosão de mensagens na rede. Mais ainda, os dados recebidos devem estar sincronizados, para que possam ser considerados válidos. A requisição e a sincronização são características que algumas RSSFs podem não suportar, e incorrem em um alto dispêndio de energia. Considera-se, então, que cada nó fará suas decisões, a partir de dados lidos localmente ou de um conjunto restrito de vizinhos, a um passo de comunicação, na maioria dos casos.

O primeiro plano de abstração, no nível mais alto de hierarquia, é o **plano de papéis**. Nesse plano, o projetista irá modelar como vértices de um grafo todos os papéis que os nós sensores podem ter durante a operação. Entende-se por perfil um certo comportamento do nó na rede. Por exemplo, em se tratando de agregação de dados, há três papéis que os nós podem assumir: fonte, agregador e sorvedouro. O nó sensor fonte é responsável por coletar os dados do ambiente e enviá-los a um nó com perfil agregador. Esse nó, por sua vez, faz a fusão dos dados de todos os nós fontes e os envia a um nó com perfil de sorvedouro, o qual é responsável por repassar todos os fluxos de dados ao PA. É importante observar que cada perfil pode ser assumido por mais de um nó em um dado instante de tempo, mas cada nó assume apenas um desses papéis. Ao longo do tempo de operação, os nós podem trocar de papéis, segundo certas regras.

Justamente a troca de papéis é o foco da modelagem da arquitetura no plano de papéis. Em primeiro lugar, a troca é somente disparada sob condições específicas. Essas

condições são modeladas como expressões booleanas associadas a arestas do grafo deste plano. As condições se referem às capacidades de *hardware* do nó sensor, seu nível de energia, à qualidade e número de enlaces disponíveis, a regras da aplicação e às condições dos nós vizinhos. As arestas indicam quais mudanças são possíveis de ocorrer, e em quais circunstâncias.

Para um dado papel, caso haja mais de uma alternativa para se realizá-lo, instancia-se o segundo plano de abstração, o **plano de alternativas**. Uma alternativa é um conjunto de fluxos, adaptações e otimizações que implementam um dado papel. Dadas as categorias de algoritmos para RSSFs - roteamento, cobertura, sincronização, localização, agregação, rastreamento - várias são as estratégias de implementação, dadas as condições da rede e dos nós. Portanto, no plano das alternativas, deseja-se mapear essa variedade de algoritmos e a possível adaptação da rede para utilizar os algoritmos adequados, de acordo com o estado da rede e do ambiente.

Da mesma forma que no plano de papéis, as alternativas são mapeadas em um grafo. Nesse grafo, os vértices são as alternativas em si e as arestas as possíveis transições (adaptações) entre alternativas. As condições de transição são expressadas por condições booleanas, as quais seguem as mesmas regras estipuladas no plano de papéis. Essas condições, devido à hierarquia de planos adotada, são avaliadas somente após a análise das condições no plano de papéis. Dessa forma, nota-se que as adaptações propostas no plano de papéis possuem prioridade sobre aquelas especificadas no plano de alternativas, reforçando-se que as decisões tomadas em um nível superior de abstração têm prioridade sobre aquelas nos níveis inferiores.

Por fim, o último plano de abstração, o **plano de fluxos**, implementa os fluxos de execução possíveis em uma alternativa. Cada fluxo de execução é expressado por um grafo, contendo o encadeamento de recursos de *hardware* e *software* necessários à sua realização. O projetista deve incluir nesse plano as adaptações do fluxo, criando, no grafo, vértices redundantes que sirvam de entrada a um módulo de *software* (também representado por um vértice), ou, de modo análogo, vários módulos de *software* que gerem saídas equivalentes. A

escolha do fluxo que será executado em um dado momento será guiada por dois parâmetros: confiabilidade dos dados e custo em energia. Esses fluxos redundantes se referem às decisões de adaptação que o nó pode realizar sem a interação com os demais nós vizinhos. Há módulos especiais, chamados módulos adaptativos, que existem especialmente para adaptações que exijam o uso de algoritmos distribuídos. Eles especificam em que condições são disparados, também, de forma similar aos papéis e alternativas. São módulos que interagem sobre os nós vizinhos, trazendo mais confiabilidade à coleta ou à difusão de dados. Podem, ainda, otimizar a energia consumida.

Ao analisar os três planos em conjunto, abrangem-se as seguintes adaptações:

Papéis - Adaptações relativas à troca de papéis (comportamentos) dos nós na rede. Por exemplo, no controle da cobertura, os nós trocam continuamente entre os papéis ligado e desligado.

Alternativas - Adaptações relativas a algoritmos equivalentes. He et al. [He et al., 2003] mostram que, para cada situação da RSSF, há uma pilha de protocolos mais adequada. Assim, não convém utilizar os mesmos protocolos durante todo o tempo de vida da rede.

Fluxos - Adaptações relativas a tolerância a falhas e otimização. A redundância explicitada pelos fluxos de execução permite adaptações locais, enquanto os módulos de adaptação lidam com a adaptação, considerando-se a vizinhança.

As adaptações, presentes nos três planos, obtêm as informações necessárias à avaliação da necessidade de adaptação através de uma única camada, a **camada de dados de adaptação**. Inspirada no conceito de *cross layer* [Marrón et al., 2005a], optou-se por manter essas informações, comuns aos três planos de abstração, em um único local para evitar a duplicação de informação e reduzir custos na geração e atualização dos dados.

3.2 Camada de dados de adaptação

A camada de dados de adaptação permeia todos os três planos de abstração. Essa camada é responsável por disponibilizar seus dados a todos os planos de abstração, obtidos através da interação com os diversos módulos de *software* da aplicação. Os dados de adaptação (DdAs) são os elementos essenciais na composição de RpAs. Esta seção tem por objetivo explicar quais são os DdAs disponíveis, como são obtidos e atualizados e, por fim, como esses dados são manipulados a fim de se criar as RpAs. Alguns exemplos de RpAs serão mostrados nessa e nas próximas seções, ao se aplicá-las nos grafos de adaptação.

Os DdAs correspondem a: i) capacidades do *hardware* subjacente; ii) métricas de qualidade (MQs) relativas ao estado do *hardware* e *software*; iii) estado de adaptação corrente. Esses três tipos de DdAs estão incluídos na camada de dados de adaptação, conforme mostra a figura 3.2, que se segue.

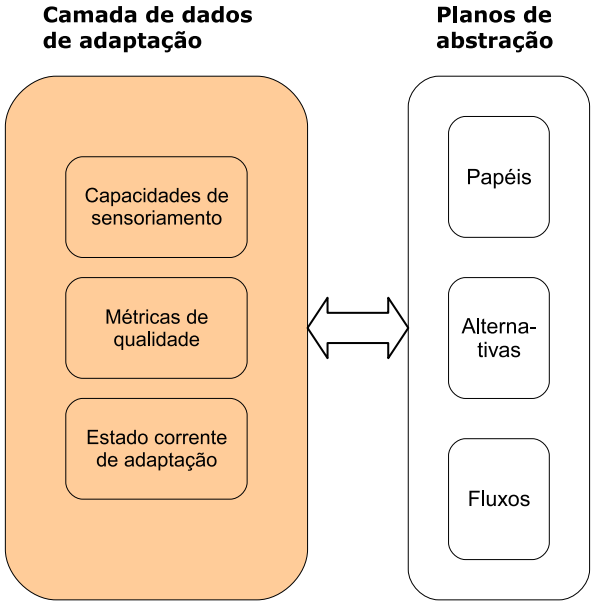

Figura 3.2: Elementos constituintes da camada de dados de aplicação e sua interação com os planos de abstração.

Como foi mostrado na seção 2.2, essas são MQs obtidas pelo próprio nó ou por troca de mensagens com os nós vizinhos. Isso significa que a camada de dados de adaptação e a aplicação devem interagir entre si, de forma que esta possa ser inspecionada em busca dos valores para os DdAs. Esses dados podem ser tanto valores lógicos (verdadeiro ou

falso) quanto enumerações ou valores inteiros. Um DdA pode ser indexado. Isso se dá, por exemplo, quando o projetista se refere ao número de nós vizinhos com um certo estado de adaptação corrente. Nessa situação, ele escreverá a expressão `regras_vizinhos[Fnt]`, a qual retornará o número de nós vizinhos com o estado de adaptação Fnt.

A tabela 2.1, no segundo capítulo, mostrou um conjunto de MQs sobre os módulos de sensoriamento, comunicação e energia, o qual é incorporado à arquitetura como dados de adaptação. Os valores desses dados de adaptação, segundo Avancha [Avancha, 2005], páginas 19–23, são delimitados pela seguinte enumeração: SUPERIOR (*ABNORMAL_HIGH*), ALTO (*HIGH*), NORMAL, BAIXO (*LOW*) e INFERIOR (*ABNORMAL_LOW*). Alguns deles não são capazes de assumir todos os cinco valores; por exemplo, a Capacidade energética restante (CER) é descrita apenas pelos valores NORMAL e INFERIOR. As regras para a determinação desses cinco valores, a cada dado de adaptação, são detalhadas em [Avancha, 2005]. Basicamente, elas consistem em obter uma medida real do módulo e aplicar limiares. Os valores dos limiares são os seguintes: i) constantes inicializadas na configuração da rede; ii) parâmetros μ e ρ da função gaussiana que determinam o comportamento normal do módulo.

Além dos dados da tabela 2.1, a camada de dados de adaptação pode incorporar outras MQs, tais como:

Capacidades de sensoriamento - Valores booleanos que descrevem a habilidade do nó sensor de capturar um certo fenômeno físico. Portanto, trata-se de uma auto-inspeção do nó por suas capacidades. Esses valores são associados diretamente a cada sensor presente no nó, ou a fenômenos físicos que podem ser deduzidos, a partir dos sensores existentes. Por exemplo, se um nó sensor é capaz de sensoriar a temperatura ambiente, o DdA PST, presença do sensor de temperatura, será verdadeiro.

Latência de transmissão - A latência, nesse caso, é medida pelo tempo percebido pela aplicação entre o envio da mensagem e a confirmação de recebimento desta. Para que esse parâmetro seja avaliado, o módulo de comunicação deve enviar um sinal à camada de dados de adaptação, sempre que receber uma requisição de envio e uma confirmação

de recebimento. A camada de dados de adaptação, por sua vez, computa o intervalo de tempo, o qual é armazenado diretamento no DdA `LT`, latência de transmissão, ou convertido em uma enumeração mais significante, como ALTO, NORMAL e BAIXO. Essa é mais uma MQ sobre o módulo de comunicação, a fim de determinar sua eficácia na transmissão de dados.

Número de rotas alternativas - O número de rotas alternativas diz respeito à redundância do módulo de sensoriamento quanto ao roteamento de mensagens. O estabelecimento de múltiplas rotas aumenta a confiabilidade na entrega das mensagens a um custo extra em energia, como discutido na seção 2.4. Essa informação é extraída do módulo de roteamento da aplicação, que deve possuir uma interface para atualizar esse parâmetro sempre que uma rota for construída ou removida. Ele é representado por um DdA simples, `NRA` (número de rotas alternativas), que armazena um valor inteiro sobre o número de rotas.

A camada de dados de adaptação pode ser expandida para incorporar outros dados, caso seja demandado por alguma AdA.

Existem duas fases para atribuição de valores aos DdAs: inicialização e atualização. A inicialização diz respeito aos valores iniciais dos DdAs, e se realiza na fase de configuração da rede (vide seção 2.5). Nessa fase, será necessária a inspeção do próprio nó e dos nós vizinhos para o preenchimento dos valores iniciais, para que seja feita a adoção das AdAs iniciais nos planos de abstração. Já a atualização é realizada durante a fase de operação da rede. Ambas as fases exigem participação ativa da aplicação, no sentido de fornecer os dados necessários à geração dos DdAs. A aplicação registra na camada de dados de adaptação os módulos de *software* que irão gerar as informações para construção dos DdAs. Se esses módulos forem desativados em caso de adaptação, os DdAs que dependem desses módulos, também, ficarão indisponíveis. Ao invés de um modelo de atualização periódica, sugere-se a implementação da atualização por meio de uma arquitetura *push*, nas quais os DdAs são atualizados, assim que os valores fornecidos pelos módulos se alterem. Dessa forma, os módulos devem informar à camada de dados sempre que seus valores forem modificados.

Essa é uma abordagem coerente ao modelo dirigido por eventos de RSSF, no qual a execução é disparada, somente quando há eventos como, por exemplo, a recepção de uma mensagem ou a leitura pronta de um sensor.

Os DdAs fornecidos pela camada de dados de adaptação são elementos essenciais para a formação de RdAs. Ao agregar um ou mais DdAs, valendo-se de operadores lógicos e de comparação, em uma expressão booleana, têm-se as RdAs. As regras são avaliadas de acordo com as políticas de cada plano de abstração.

3.3 Planos de abstração

Enquanto a camada de dados de adaptação fornece toda a infra-estrutura para a composição de RpAs, os planos de abstração representam a estratégia de AdAs a partir da análise das RpAs. Há três planos de abstração na arquitetura: papéis, alternativas e fluxos. Cada um deles será explanado nas três subseções seguintes.

3.3.1 Papéis

Conforme a arquitetura proposta neste trabalho, a aplicação será pensada, primeiramente, como um conjunto de papéis. Cada **papel** equivale a um comportamento de um nó durante um intervalo de operação da rede. O comportamento define o que o nó irá executar, em termos de sensoriamento, processamento e comunicação, além de indicar como se dá a interação com os demais nós vizinhos. Por exemplo, um nó com papel *roteador* tem a única função de repassar mensagens recebidas para outros nós, enquanto um nó com papel *rastreador* participa, junto com outros nós, na percepção de um EI que se move pelo ambiente monitorado. O papel não especifica diretamente quais são os módulos de *hardware*/*software* que implementam as funcionalidades do nó, mas sim a atitude do nó diante das características momentâneas do próprio nó, da rede ou do CI. A interação entre os nós, cada qual com seu papel, determina, em última instância, o comportamento geral da aplicação como um todo. O plano de papéis deverá, enfim, conter todos os possíveis comportamentos/papéis

que um nó possa adquirir, portanto, deve especificar toda a aplicação através dessa abstração.

Esse plano de abstração, assim como os outros dois, dispõe de um grafo que determina as adaptações possíveis nesse plano. Seus vértices correspodem aos papéis que um nó sensor pode assumir, enquanto suas arestas conectam papéis em que seja possível a adaptação, ou seja, a troca entre os dois papéis. Além disso, cada papel tem, associado a si, uma RpA que determina em que condições aquele papel é adequado de ser desempenhado pelo nó. Vejamos um exemplo na figura 3.3, abaixo.

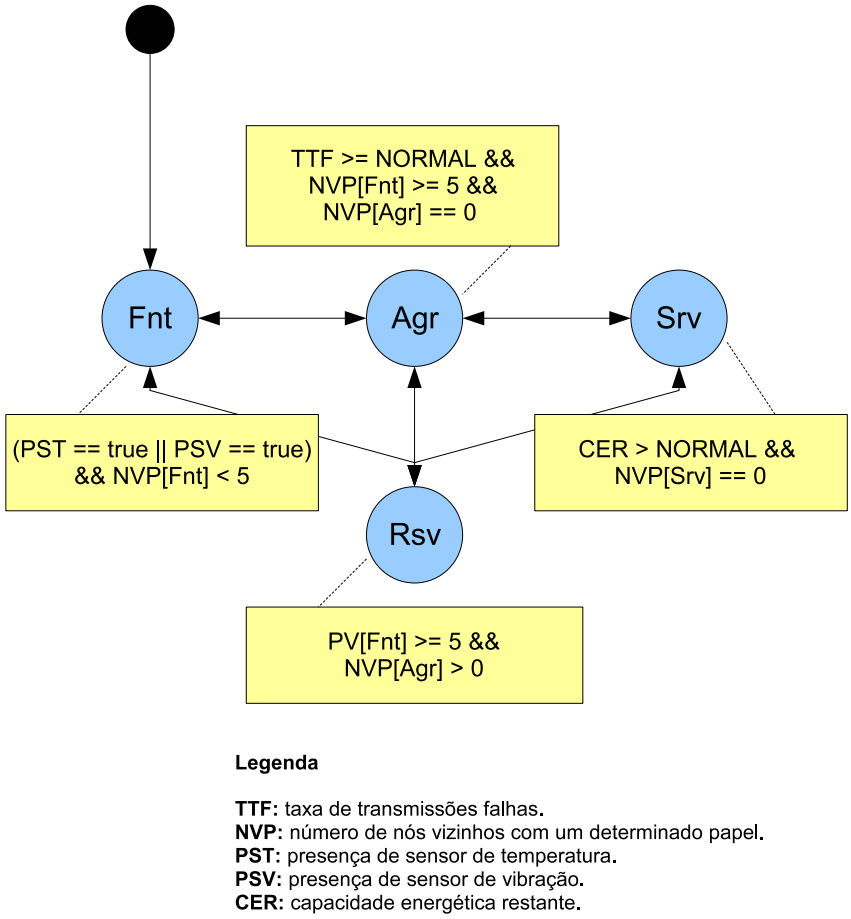

Figura 3.3: Exemplo de um grafo no plano de papéis.

O grafo acima contém cinco papéis, representados pelos vértices: fonte (`Fnt`), agregador (`Agr`), sorvedouro (`Srv`) e reserva (`Rsv`). Esses papéis, na verdade, representam uma aplicação de monitoração em que há agregação de dados, como foi esquematizado na figura 2.1. Acrescenta-se a isso um gerenciamento de topologia, proporcionado pelo papel `Rsv`. Nesse último papel, o nó entrará em modo econômico, desligando os sensores e utilizando

o transceptor em uma freqüência mínima. Existe um vértice especial, em que não há arestas entrantes, representando o estado inicial dos nós. A partir dele, inicia-se a avaliação das RpAs para assinalamento de papéis. Nesse caso, todos os nós, a princípio, terão o papel `Fnt` e, após avaliar as RpAs, transitarão para outros papéis ou se manterão neste.

Os DdAs usados são: `PST`, `TTF` e `RV`. Esses três DdAs são usados para formar as RpAs de cada papel, essas últimas representadas pelos retângulos associados aos vértices. O DdA `PST`, presença de sensor de temperatura, possui um valor booleano que indica a presença ou ausência de termistor no nó sensor. `TTF`, taxa de transmissões falhas, inclui uma enumeração quanto à taxa de falhas no envio de mensagens. `NVP`, número de nós vizinhos com um papel, armazena o número de nós vizinhos que possui um determinado papel. Esse último DdA é indexado pelo papel, como no exemplo `PV[Fnt] < 5`, em que se verifica se existem menos de cinco nós vizinhos com o papel `Fnt`. Esse dado é obtido por meio dos próprios nós vizinhos, que enviam uma mensagem, sempre que eles alteram seu papel. Essa forma de atualização do `NVP` está de acordo com a arquitetura *push* descrita na seção 3.2.

A regra para avaliação dos RpAs é a seguinte: avalia-se, inicialmente, a RpA do papel corrente. Se ela for avaliada em falso, então, avaliam-se as RpAs dos papéis adjacentes ao papel corrente. Escolhe-se, após, uma delas que avalie em verdadeiro, aleatoriamente.Na realidade, a escolha aleatória é apenas uma simplificação: papéis equivalentes podem ter um critério de desempate mais sofisticado. Por exemplo, cada papel pode ser associado a um número, o qual indica a prioridade desse papel em relação a outros, quanto à métrica principal de otimização da aplicação. Se a métrica de otimização for o tempo de vida da rede, os papéis que consumirem menos energia serão priorizados.

Ao se criar as RpAs de cada papel, deve-se levar em conta a dinâmica de adaptação, para que se alcance o resultado esperado. Regras mal elaboradas podem levar a uma situação indesejada em que todos os nós permanecem no mesmo papel, ou que alguns papéis nunca sejam atribuídos a algum nó. Essa deve ser uma preocupação do projetista, que deve construir cuidadosamente as RpAs para que não ocorram esses problemas. Uma estratégia básica para isso é verificar se as RpAs adjacentes possuem regras análogas. Por exemplo, enquanto o

papel `Fnt` contém a RpA com a expressão `PV[Fnt] < 5`, o papel `Agr`, adjacente a `Fnt`, contém a Rpa com a expressão `PV[Fnt] >= 5`. Como essas duas expressões não podem ser verdadeiras ao mesmo tempo, isso garante a seleção de um ou outro papel. É importante ressaltar que isso é feito para respeitar a restrição de que um nó, em um dado momento da operação, deve ter apenas um papel.

A troca de papéis, efetivada pela aplicação, implica mudança do conjunto de alternativas e fluxos em execução. Cada papel é associado a uma ou mais alternativas, e estas, por sua vez, associadas a um ou mais fluxos. Em se tratando do modelo de execução, quando um papel se altera, o conjunto de alternativas disponíveis passa a ser aquele associado ao novo papel. Por conseqüência, o conjunto de fluxos disponíveis também se modifica. Esses novos fluxos são responsáveis pela alteração no comportamento do nó designada pelo rótulo do papel. Isso porque os fluxos, em última instância, determinam quais mensagens de aplicação são transmitidas e quais sensores são requisitados.

O plano de papéis, conforme Frank e Römer [Frank e Römer, 2005], é uma abstração para o nível de sistemas em aplicações para RSSF. Ao invés do projetista elaborar complicados módulos de *software* para gerenciar cobertura, topologia ou agrupamento (*clustering*), ele utiliza essa abstração na qual o problema passa a ser a elaboração correta das regras, tarefa mais simples do que a depuração dos algoritmos distribuídos embutidos nos módulos. Portanto, é uma ferramenta eficaz, simples de usar e poderosa para a definição da interação entre comportamentos complexos em RSSFs. Ao mesmo tempo, o plano de papéis não pode cobrir todos os casos imagináveis. Haverá situações, por exemplo, em que o uso de um módulo de *software* customizado poderá trazer ganhos consideráveis, ou mesmo não seja uma abstração adequada para uma certa interação entre os nós sensores. Nesses casos, o projetista que recorre a essa arquitetura deve implementar essa interação no plano de fluxos. Mesmo assim, o plano de papéis é uma abstração que o projetista deve considerar em primeiro lugar, devido a sua simplicidade, antes de partir para soluções customizadas.

3.3.2 Alternativas

O plano de alternativas diz respeito a alternativas de implementação para cada um dos papéis definidos. A criação de alternativas se faz justamente quando há a necessidade de se implantar mais de um forma de realização de um mesmo papel.

Para fixar um exemplo, observe o caso do papel de agregação (Agr), mostrado na figura 3.3. Esse papel especifica que o nó deve receber os dados dos demais nós sensores com o papel fonte (Fnt), fazer a fusão desses dados e reenviá-los a um nó sorvedouro (Srv). Entretanto, não fica claro qual o método desejado para realizar a fusão, por exemplo. Como já foi discutido na seção 2.3, as técnicas de média, mediana ou intervalos de confiança são apenas o início de uma série de algoritmos que já foram propostos e ainda estão por serem desenvolvidos. Em se tratando da recepção e entrega de mensagens, também não é explicitado qual o algoritmo de roteamento deve ser aplicado. He et al. [He et al., 2003] analisam essa classe de algoritmos e concluem que o desempenho de um ou outro está condicionado às condições da rede. Por exemplo, quando o número de nós sorvedouros for grande, a difusão direcionada [Intanagonwiwat et al., 2003] poderá ser mais eficiente. Da mesma forma, se houver disponível na rede a informação de localização do nó, algoritmos como o GPSR [Karp e Kung, 2000], que dependem dessa informação, podem se destacar frente aos outros.

Mantendo o exemplo anterior, a idéia é criar um grafo associado ao papel de agregação, no qual seus vértices representem cada método de fusão que possa ser empregado. As arestas conectam alternativas intercambiáveis, determinando quais alternativas são alcançáveis, a partir da alternativa adotada em um certo momento. A cada vértice, associam-se duas funções: objetivos e restrições. Essas funções são processadas a fim de que se possa escolher a melhor alternativa, dadas as características da rede no momento. A figura 3.4, na página seguinte, mostra o exemplo de um grafo de alternativas para o papel agregador (Agr), definido na seção anterior.

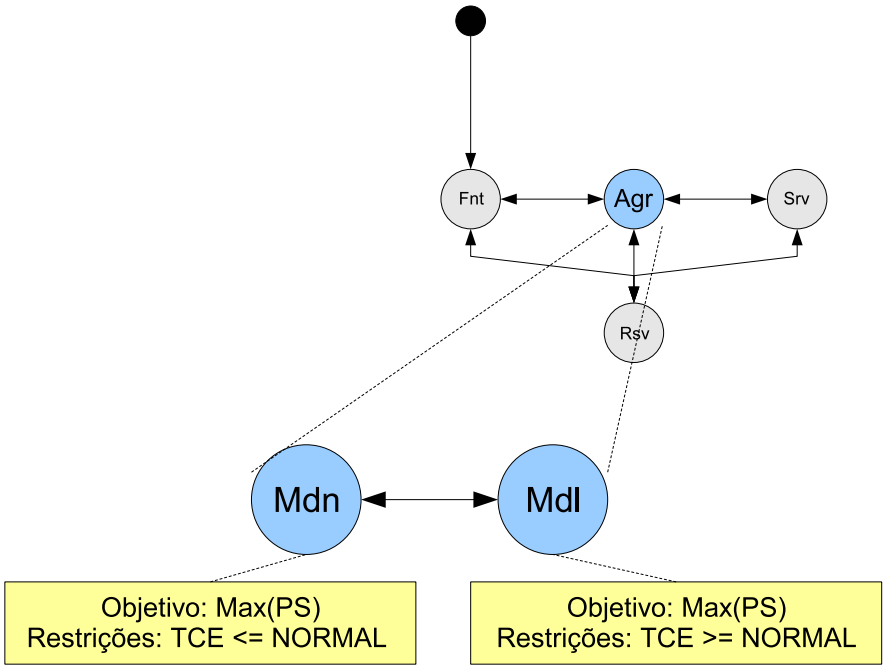

Figura 3.4: Exemplo de alternativas associadas ao papel Agr.

No grafo acima, existem duas alternativas à implementação do papel Agr: mediana (Mdn) e modelo (Mdl). Na primeira alternativa, Mdn, será empregada uma simples mediana frente a um conjunto de dados sobre a mesma região em um certo intervalo de tempo. Já na segunda alternativa, Mdl, emprega-se um modelo do ambiente, baseando-se no trabalho de Jain e Chang [Jain e Chang, 2004]. Nesse caso, os dados de sensoriamento são fornecidos como entrada para um estimador com filtro de Kalman. O estimador consegue predizer os futuros valores de sensoriamento, de acordo com aqueles vistos até então. Por um lado, com o estimador, é possível enviar apenas os dados sensoriados que estejam suficientemente fora da curva de predição. Por outro, nós, cujos dados sejam muito discrepantes dos demais, podem ser ignorados ou ter seus sensores desligados, obrigando-os a adotar outro papel na rede.

Observa-se, nesse contexto, a importância da camada de dados de adaptação em relação à comunicação entre os planos de abstração. Nesse caso, entre o plano de papéis e o plano de alternativas. Mostrou-se, assim, que uma mudança de alternativa pode provocar uma mudança de papel nos nós vizinhos. Isso porque uma alternativa pode alterar um DdA, local ou remoto, utilizado em uma RpA no plano de papéis. Aqui, tem-se o efeito esperado:

o papel de um nó fonte só faz sentido se seus sensores estiverem funcionando corretamente, e uma alternativa pode detectar justamente um sensor com falhas. Isso, por conseqüência, leva a uma mudança do papel desse nó na rede, de acordo com a falha detectada e não-resolvida.

A cada vértice do grafo de alternativas, aliam-se duas funções: objetivo e restrição. A função objetivo diz respeito a quais DdAs serão maximizados ou minimizados. Essa função traça, então, o porquê de se adotar uma alternativa em detrimento de outra. A função restrição, por outro lado, mostra qual o custo de se adotar aquela alternativa. Ambas as funções, em tempo de execução, são avaliadas nessa camada para efetuar a escolha de uma alternativa, aquela que será dotada pelo nó sensor, em determinado momento do tempo de vida da rede. Essas funções, em conjunto, compõem a RpA de uma dada alternativa.

A avaliação do grafo pelo plano de alternativas se dá de formas diferentes, dependendo da fase da RSSF. Quando na fase de configuração da rede, todas as alternativas são avaliadas. Então, uma delas é escolhida como a alternativa corrente. Isso é necessário, pois não há, nesse grafo, um vértice "inicial" que indique qual alternativa deve ser adotada por padrão. É preferível, então, que todas as alternativas sejam avaliadas e que apenas uma seja adotada, conforme as características do nó sensor e de seus vizinhos no início da rede. Quando na fase de operação da rede, são avaliadas as alternativas alcançáveis a partir da alternativa corrente.

Em ambas as avaliações, as RpAs das alternativas em questão são comparadas às políticas de adaptação descritas pelo projetista. Essas políticas, que são compartilhadas com o plano de fluxos, são regras que incluem a maximização ou minimização de MQs, tais como a função objetivo. Essas regras são criadas de acordo com o levantamentos de requisitos da aplicação, que identifica quais MQs relativas ao estado da aplicação (vide tabela 2.2) são importantes à aplicação, e em qual prioridade. Por exemplo, uma aplicação pode determinar que sua prioridade principal é maximizar os EIs detectados, e, como segunda prioridade, a maximização do tempo de vida da rede. Dessa forma, as adaptações relativas à detecção do máximo de EIs são priorizadas em detrimentos das adaptações que conservam energia da energia.

As MQs relativas ao estado da aplicação, como o próprio nome informa, referem-se a métricas percebidas pelo usuário, ao observar a qualidade dos serviços na perspectiva da rede. Dessa forma, não são métricas fáceis de serem medidas pelos próprios nós, os quais possuem uma visão parcial da rede. Logo, ao incorporá-las às políticas de adaptação, elas devem ser reduzidas às MQs sobre o estado do *hardware* e *software* (vide tabela 2.1), as quais são obtidas pela inspeção dessa visão parcial. No exemplo dado no parágrafo anterior, a maximização dos EIs detectados pode ser aproximada pela maximização da precisão do sensor (PS) e do número de roteadores favoráveis (NRF), enquanto a maximização do tempo de vida da rede pode ser dada pela maximização da capacidade energética restante (CER) e minimização da taxa de consumo de energia (TCE). Ao aplicar a prioridade imposta para essas duas MQs, a política de adaptação é descrita em duas expressões, na seguinte ordem: `Max(PS) && Max(NRF)` e `Max(CER) && Min(TCE)`.

Mostrou-se até então que as alternativas referem-se a uma estratégia de implementação de um papel. Entretanto, a definição concreta da estratégia, o que envolve a especificação dos módulos de *software* e dos recursos de sensoriamento e comunicação usados, é delegada ao plano subjacente - o plano de fluxos.

A criação do grafo de alternativas é interessante se os fluxos subjacentes forem, consideravelmente, distintos entre si. Como será visto na seção 3.3.3, seguinte, há maneiras, no plano de fluxos, de especificar redundância na estratégia de processamento de dados. Nesse caso, bastaria incluir dois módulos de *software* que produzissem o mesmo resultado. Essa forma é suficientemente adequada, quando se trata de uma redundância mais simples de processamento. Entretanto, quando essa redundância se torna mais complexa, exigindo a colaboração de vários módulos de software e dados distintos, sua incorporação em um mesmo fluxo traz complicações ao raciocínio do projetista sobre esse fluxo. Enfim, as alternativas isolam fluxos distintos para um mesmo papel em grafos distintos, favorecendo, assim, a compreensão da modelagem da aplicação.

Nem sempre esse grafo é aplicável. Não há a necessidade de criação de alternativas, quando não há alternativas na implementação de um papel, ou quando essas alternativas

teriam um custo alto para serem efetuadas durante a operação, como pode ser o caso da mudança de algoritmos de roteamento. Nessas situações, as adaptações serão abordadas diretamente no plano de abstração subjacente.

3.3.3 Fluxos

O plano de fluxos é o terceiro e último dos planos de abstração. Nesse plano, são especificados os módulos de *software* e os dados consumidos e produzidos por esses módulos. Dessa forma, esse plano está em um nível de abstração mais próximo da implementação da aplicação. Nesse plano, a adaptação se concretiza na redundância de caminhos nos fluxos e no emprego dos módulos de adaptação distribuídos. Esses recursos, em conjunto, trazem flexibilidade quanto à otimização da execução, conforme as políticas de adaptação instituídas pelo projetista.

Esse plano, enfim, trata-se do refinamento final, feito pelo projetista, à modelagem da sua aplicação, ao implementar a arquitetura de adaptação proposta neste trabalho. A figura 3.5 evidencia como é realizado esse refinamento, considerando-se todos os três planos.

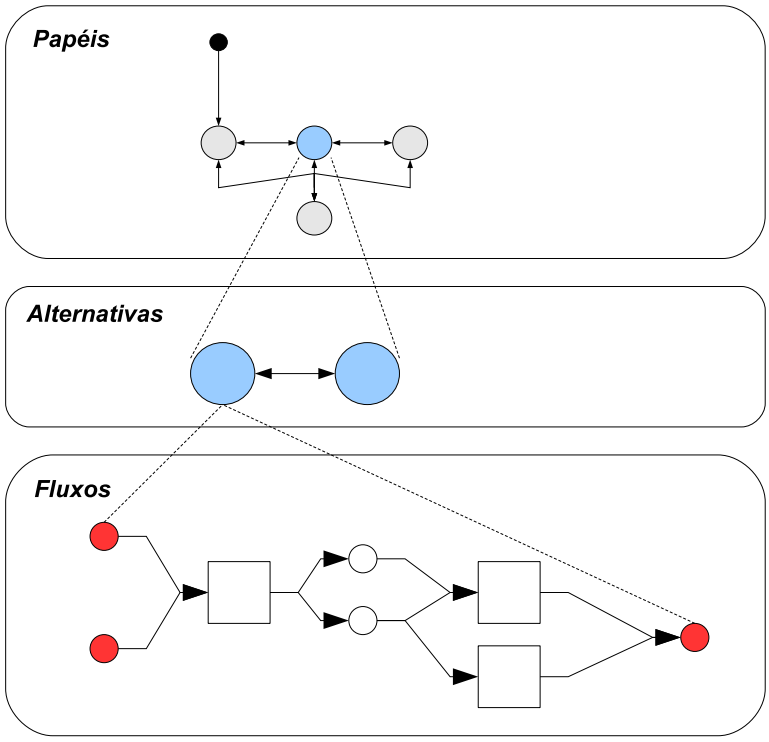

Figura 3.5: Refinamento da modelagem, do plano de papéis ao plano de fluxos.

A figura acima discorre, também, sobre o modelo de execução/interpretação da arquitetura. A aplicação, por meio dessa arquitetura, é descrita como um conjunto de papéis, alternativas e fluxos. Em um dado instante da execução da aplicação, a arquitetura determina que haverá um papel, uma alternativa e um fluxo atribuídos ao nó. Essas escolhas são realizadas na seguinte seqüência:

1. Um vértice no grafo contido no plano de papéis é escolhido como o papel corrente.

2. Analisa-se o grafo de alternativas, associado ao papel corrente. Um dos vértices desse grafo é escolhido como a alternativa corrente.

3. Analisa-se o grafo de fluxos associado à alternativa corrente. Escolhe-se um caminho nesse grafo como o fluxo corrente.

Como foi mostrado nas seções 3.3.1 e 3.3.2, os papéis e alternativas não estão ligados diretamente ao código que será executado pelos nós, mas determinam quais estratégias de adaptação estarão presentes no grafo de fluxos subjacente. Portanto, ao especificar o grafo de fluxos, o projetista deve ter em mente a qual alternativa e a qual papel esse fluxo se refere, a fim de incorporar as regras de adaptação sugeridas por esses ao grafo. Voltando ao exemplo do papel Agregador (Agr), ao especificar o grafo de fluxos correspondente à alternativa Mediana (Mdn), devem ser incorporados módulos de *software* que realizem a mediana e a transmissão de dados para o nó sorvedouro. Isso é responsabilidade do projetista, que deve ser levada em conta no momento da criação do grafo de fluxos. Se isso não for feito, as decisões tomadas para a escolha do papel e da alternativa não serão efetivas, visto que essas decisões justamente se valem do comportamento esperado do fluxo.

O grafo de fluxos é baseado no trabalho de Caldas [Caldas, 2004], o qual considera a modelagem de aplicações adaptativas para robôs através de um fluxo de dados. A modelagem de fluxos possui características interessantes: explicita dados e processamentos redundantes, os quais são as opções de reconfiguração da execução; abstrai a plataforma de *hardware* utilizada para a execução da aplicação; oferece um controle do escalonamento da obtenção de dados e da ordem de execução de cada trecho de *software*. Todos esses fatores permitem

a modelagem da redundância em um nível próximo ao da implementação, razão esta que confirmou a adoção desse tipo de abordagem também para o contexto das RSSFs. Ao obter uma representação gráfica do grafo, conforme mostrado na figura 3.6, o projetista facilmente visualiza se há redundância quanto à geração e processamento dos dados sensoriados.

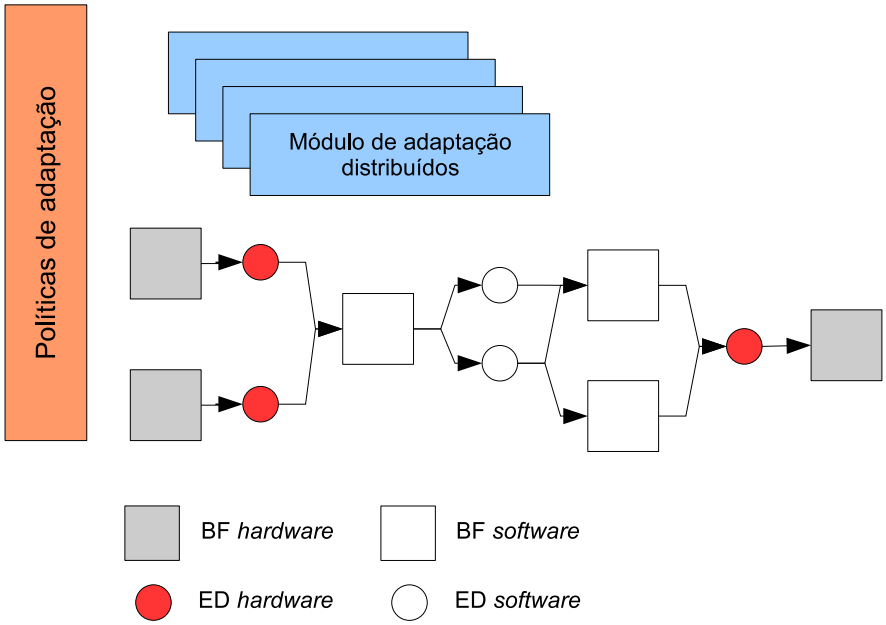

Figura 3.6: Elementos principais na definição do grafo de fluxos.

No grafo de fluxos, tal como apresentado na figura acima, há dois tipos de vértices: elementos de dados (EDs), mostrados como círculos, e blocos funcionais (BFs). Enquanto os blocos funcionais modelam os módulos de *software* que irão executar no nó, os elementos de dados indicam quais dados esses módulos produzem e consumem. Esses dados podem ter as seguintes naturezas: i) valores de entrada e/ou saída dos blocos funcionais; ii) dados de interação com o *hardware*, tais como amostragem dos sensores ou mensagens recebidas ou transmitidas pelo transceptor.

Os blocos funcionais abrangem todo o processamento dos dados realizado pelo nó sensor. Cada BF possui um conjunto de EDs de entrada e EDs de saída. Há três passos principais para a execução de um BF: i) leitura dos valores dos EDs de entrada; ii) processamento dos valores lidos; iii) escrita de valores nos EDs de saída. Quando um desses conjuntos de EDs refere-se ao *hardware*, os BFs associados têm a função de encapsular a interface da

aplicação com o *hardware*. Nesse contexto, os BFs trabalham em duas vias, recebendo e enviando dados ao *hardware*. A aplicação presencia o recebimento de dados do *hardware* ao amostrar algum sensor e/ou receber mensagens dos outros nós. Já a transmissão de mensagens aos outros nós ou a gravação de dados em memória persistente são exemplos típicos em que há o consumo de dados da aplicação pelo *hardware*. Dessa forma, sempre haverá BFs no início e no fim de um caminho no grafo, os quais lidam, respectivamente, com a entrada e saída de dados da aplicação com o *hardware*.

Os elementos de dados são o repositório de informações para os BFs. Separando-se os dados do processamento desses, torna-se possível compartilhar EDs entre BFs distintos, ou mesmo preencher um ED com o valor de dois ou mais processamentos distintos. Nessas duas situações, fica explícita a redundância no fluxo. Na primeira situação, o BF pode usar como entrada apenas aqueles EDs que possuam maior confiabilidade, ou de acordo com qualquer outro conjunto de MQs que se deseje maximizar/minimizar. Na segunda situação, na qual vários BFs geram valores para o mesmo ED, é possível implementar, pelo menos, uma destas duas estratégias para a atribuição do valor: escolha do melhor dado e fusão dos dados. Quem decide a estratégia de atribuição é o próprio ED, que possui uma função de análise dos dados recebidos. O objetivo da aplicação dessa estratégia é manter o nível de qualidade dos dados do fluxo, mesmo na presença de falhas ou em condições adversas da rede ou do ambiente.

Com o grafo de fluxos criado, resta discorrer sobre como esse grafo é tratado em tempo de execução. Esse tratamento se dá, basicamente, na escolha de um caminho (fluxo) no grafo e na execução desse caminho. Como o grafo apresenta fluxos de processamento redundantes, a escolha de um caminho trata-se da otimização das MQs relevantes à aplicação. Essas MQs são definidas como políticas de adaptação, políticas essas compartilhadas com o plano de alternativas. A idéia geral é que os fluxos redundantes sejam periodicamente comparados ao fluxo corrente. Caso seja constatado que um outro fluxo atenderá melhor às políticas de adaptação, esse deve ser adotado a partir de então. É importante ressaltar que o custo de alteração do fluxo corrente é mínimo, envolvendo, talvez, a ativação de uma ou

mais partes do *hardware* que estavam excluídas do fluxo corrente. Geralmente, a mudança de fluxo aborda apenas a alteração da estrutura de dados que mantém os dados sobre quais EDs e BFs estão envolvidos no fluxo corrente, sem necessidade de alterações no estado da aplicação, visto que este se encontra nos EDs que já estavam ativos antes da mudança.

A definição de uma heurística ideal na seleção do fluxo fica como um trabalho futuro. A estratégia adotada em [Caldas, 2004], quanto ao cálculo do índice de ganho dos fluxos redundantes, é perfeitamente viável, passando por modificações para atuar com um número variável de MQs ao invés de apenas dois parâmetros, desempenho e confiabilidade, utilizados correntemente.

Depois da escolha de um fluxo, sua execução estará condicionada a um certo escalonamento. O escalonamento refere-se à ordem em que os EDs são lidos e escritos e à ordem que os BFs são executados. O próprio fluxo determina, através de suas arestas, a dependência existente entre os BFs. Essa informação é utilizada no escalonamento para que essa ordem entre os BFs seja respeitada. Quanto aos EDs, existem as questões de redundância. Quando há um conjunto de EDs de entrada, o BF deve aguardar o preenchimento de valores a todo o conjunto antes de executar o processamento propriamente dito. Quanto aos EDs de saída, caso haja a geração de valores a esses EDs por mais de um BF, os dois valores são aguardados pelo ED, o qual, então, executa uma função que determina qual o valor final que ele irá carregar.

Apesar das vantagens da utilização do grafo de fluxos, constatei, neste estudo, que ele não é adequado para especificar todo o tipo de algoritmos que haverá na RSSF, em especial, algoritmos distribuídos. Essa classe de algoritmos é especificada por meio de uma complexa interação com os nós vizinhos através da troca de mensagens. Essa interação, em um fluxo, pode ser representada se o algoritmo for dividido em vários BFs, um para cada processamento executado em resposta à a uma mensagem de controle do algoritmo. Dessa forma, cada algoritmo empregado incitaria a construção de um complexo grafo de fluxo, o que traria ilegibilidade, sem nenhum ganho em termos de adaptação.

Consideradas essas dificuldades, foram introduzidos os módulos de adaptação dis-

tribuídos (MADs). Esses módulos implementam algoritmos distribuídos que se utilizam das informações dos nós vizinhos, e incluem todas as interações necessárias para a realização destes. Esses algoritmos podem lidar com diversos tipos de adaptações. Os exemplos típicos dessa classe de adaptação são o controle de topologia [Cerpa e Estrin, 2004] e o ajuste de confiabilidade no sensoriamento de um EI [Deshpande et al., 2005]. Os MADs são executados em paralelo ao fluxo corrente, ajudando no alcance das políticas de adaptação impostas. Eles, também, podem trocar informações com o fluxo, na medida em que podem ler ou escrever dados nos EDs do fluxo.

O plano de alternativas pode lidar com a situação em que um ou mais MADs devem ser desativados. A desativação é justificada por situações em que o benefício desses módulos se mantém inferior ao consumo de recursos dos mesmos. Para prever a ativação/desativação de MADs, um vértice pode ser criado no grafo de alternativas para cada um desses casos. Dessa forma, fica a cargo do plano de alternativas analisar se a alternativa corrente, com um determinado conjunto de MADs ativos, está atendendo adequadamente às políticas de adaptação e, se for o caso, mudar para uma alternativa diferente que possua um outro conjunto de MADs.

3.4 Estudo de caso: pontes auto-suficientes

O estudo de caso consiste em uma aplicação de monitoramento, muito comum no contexto ecológico, estrutural ou militar. Ela, basicamente, consiste na aquisição de dados sobre o ambiente - temperatura, pressão, som, vibração, intensidade de luz, entre outros - e no processamento desses dados, cujo resultado é enviado à estação-base. Durante o processamento, os dados podem ser simplesmente repassados sem modificações, agregados de acordo com regiões de interesse, ou mesmo filtrados para que a estação-base seja notificada, apenas em caso de um EI, a exemplo de incêndios florestais, rachaduras na infra-estrutura, detecção de tanques inimigos.

A aplicação de monitoramento escolhida é a de pontes auto-suficientes (*sustainable bridges*) apresentada em [Marrón et al., 2005b]. O objetivo da aplicação é monitorar as con-

dições físicas de uma ponte a fim de se detectar defeitos estruturais, tais como rachaduras. Sensores de umidade, temperatura, vibração e som são utéis nesse sentido. Mecanismos de localização e sincronização, também, serão utéis para localizar os dados sensoriais no tempo e espaço. Um requisito importante é que a base instalada de nós deve funcionar por meses, sem necessidade de intervenção manual. A figura 3.7 apresenta um esquema de deposição desses nós na ponte.

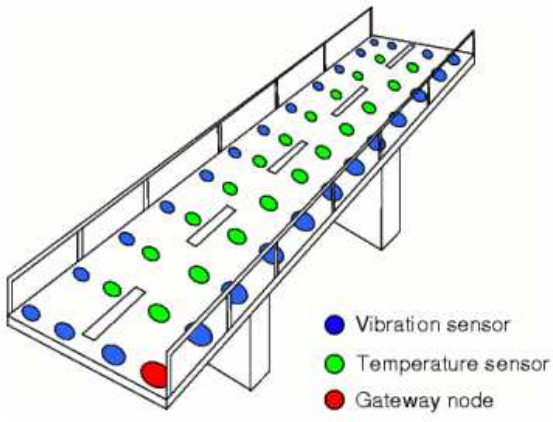

Figura 3.7: Modelo de deposição da aplicação de pontes auto-suficientes (*sustainable bridges*). Fonte: [Lachenmann, 2006].

A figura mostra que os nós com sensores de vibração estão posicionados ao longo das bordas da ponte, nas quais os movimentos da estrutura são percebidos com maior intensidade. Ao longo da ponte, observa-se nós com sensores de temperatura. Essa é a região da ponte com maior área de contato com a luz solar e os veículos, ambos fontes de calor que podem provocar rachaduras ao longo do tempo. Há também um nó sensor do tipo sorvedouro (*gateway*), cujo objetivo é transportar os dados da RSSF para a Internet, na qual são visualizados, armazenados e processados de forma conveniente por computadores comuns.

Antes de aplicar a arquitetura diretamente, é interessante, primeiramente, levantar quais adaptações serão necessárias. Há três aspectos de adaptação: aplicação, falhas e otimização. O primeiro aspecto, aplicação, diz respeito a adaptações impostas pelas características da aplicação em especial. O segundo aspecto, falhas, registra as adaptações que reagem a quaisquer tipos de falhas na RSSF, que comprometam a precisão ou a responsividade da aplicação. Por fim, o aspecto de otimização refere-se à maximização ou minimização de

MQs, de forma que a aplicação trabalhe da melhor forma possível. Essa primeira visão das adaptações serve como um conjunto inicial de idéias (*brainstorming*) de como o aspecto de adaptação irá influenciar a RSSF.

Durante o levantamento das adaptações, foram encontradas duas falhas críticas que devem ser necessariamente tratadas nesta aplicação:

- Falhas dos sensores: os sensores apresentam duas classes de erros: bizantinos e permanentes. Erros bizantinos dizem respeito ao desvio temporário e inadvertido do comportamento esperado. Nos sensores, ele se manifesta em leituras díspares dos dados verdadeiros ou daqueles reportados por, pelo menos, cinco nós sensores vizinhos [Gao et al., 2003]. Esses erros podem fazer com que seja reportado erroneamente o acontecimento de um EI. Quanto às falhas permanentes, são falhas em que o sensor reporta sempre um valor constante ou valores fora dos limiares daquilo que é esperado do ambiente. Esses sensores devem ser isolados para que seu erro não influencie na função de agregação.

- Baixa conectividade da rede: a comunicação de um nó com os demais pode ser interrompida inesperadamente durante qualquer momento da operação. Algum obstáculo pode surgir e retirar um nó da linha de visão dos demais ou o transceptor pode falhar por completo, deixando de enviar e receber mensagens da rede. Em uma RSSF, um nó sem comunicação torna-se praticamente inútil. À medida em que vários nós apresentem esse tipo de falha, a conectividade diminui, o que leva, por fim, ao particionamento da rede. Para contornar essa condição, mantêm-se alguns nós como reservas, a fim de que entrem em operação para cobrir aqueles que falharam.

Quanto às restrições impostas pela aplicação, encontraram-se dois tópicos essenciais:

- Defeitos estruturais não podem ser ignorados: A RSSF deve estar sempre alerta a um indício de rachaduras ou outros defeitos estruturais na ponte. Isso significa que nenhuma leitura anormal, à exceção de falhas, deve ser ignorada. Isso poderia ocorrer

caso uma região deixe de ser coberta por nós sensores, caso haja muitos nós no modo reserva ou caso haja uma súbita falha em um grande número de nós. A estratégia para garantir a detecção de EIs será explorar a redundância obtida por dois tipos de sensores aliada ao controle de topologia.

- Dados sobre o defeito estrutural devem ser completos: ao ocorrer um EI, os dados sobre esse evento devem ser enviados de forma completa, sem agregação ou outra forma de fusão, para que possam ser analisados com cuidado pelos computadores conectados ao PA. Como será mostrado ainda nessa seção, o envio dos dados completos será modelado como uma alternativa ao envio dos dados simplificados, adotado por outra alternativa.

Por fim, ao pensar sobre possíveis adaptações de otimização, foram levantadas duas funções indispensáveis, as quais são exibidas a seguir em ordem decrescente de prioridade:

- Maximizar a disponibilidade de rede: o primeiro propósito é a maximização da diponibilidade da rede, visto que ignorar um EI pode levar a uma quebra na estrutura e, conseqüentemente, a uma catástrofe. Assim, a disponibilidade da rede é vista como elemento essencial para evitar essa situação indesejável. Uma disponibilidade máxima significa que os nós sensores estão, praticamente, em 100% do tempo de operação, monitorando todo o CI, contornando as falhas que possam interromper esse serviço de monitoração.

- Maximizar o tempo de vida da rede: após conseguir a disponibilidade necessária, o segundo objetivo é maximizar o tempo em que os nós podem operar sem intervenção humana. Um tempo maior de independência gera redução nos custos de manutenção da rede, a qual pode ser complicada se envolver reposições de nós em locais de difícil acesso na estrutura. Para atingir esse objetivo, os nós devem estar, sempre que possível, consumindo o mínimo de energia ao desligar partes do *hardware* que não estiverem em uso.

Com a determinação das adaptações relevantes, parte-se para a aplicação da arquitetura proposta neste trabalho à modelagem da aplicação. A aplicação será refinada enquanto se atravessam os três planos de abstração: papéis, alternativas e fluxos. Ao alcançar o último plano, tem-se um modelo que mapeia os anseios por medidas adaptativas, levantadas anteriormente, em um plano de avaliação e execução dessas adaptações por cada nó sensor.

Em primeiro lugar, determina-se quais são os papéis existentes na aplicação e, assim, constrói-se o grafo de papéis. Como se trata de uma aplicação de monitoramento, há três papéis principais: fonte, agregador e sorvedouro. Como há também a necessidade de haver nós reservas, será adotado o grafo de papéis que foi apresentado na figura 3.3. Ela contém todos os três papéis descritos acima, mais o papel reserva (Rsv). Uma explicação mais detalhada desse grafo se encontra na seção 3.3.1.

Observa-se, na verdade, que, para cada classe de aplicações, há um conjunto de papéis pré-determinados. Uma investigação interessante, a qual é apresentada como proposta de trabalho futuro, é descobrir e avaliar conjuntos de papéis que aparecem comumente nas classes de aplicações para RSSFs. Dessa forma, seria possível criar um arcabouço de papéis reutilizável em várias aplicações, os quais teriam sua eficiência e aplicabilidade bem estudadas.

Definido o grafo de papéis, parte-se, então, para o plano de alternativas. Para simplificar esse estudo de caso, especifico duas alternativas para o papel Fonte (Fnt): detecção (Dtc) e completo (Cpl). Os demais papéis, Agr, Srv e Rsv, não terão alternativas especificadas, apesar de ser possível incorporá-las, como ilustrou a figura 3.4 para o papel Agr. Os nós sensores com a alternativa Dtc enviam dados sumarizados ao nó agregador, enquanto os nós com a alternativa Cpl enviam todos os dados sensoriados a uma freqüência elevada ao nó agregador. A alternativa Dtc ainda se utiliza de algoritmos mais agressivos para obtenção de confiança nos dados, se comparada à alternativa Cpl. Essas alternativas, enfim, modelam duas situações distintas: i) não há EIs e os nós sensores se mantêm em um modo razoavelmente ecônomico em consumo de energia, aplicando algoritmos para aumentar a confiança dos dados e evitar alarmes falsos; ii) já houve uma detecção com sucesso de um EI e, nesse

caso, há uma remoção superficial de erros e o envio dos dados de sensoriamento brutos. O grafo de alternativas descrito nesse parágrafo encontra-se ilustrado na figura 3.8 seguinte.

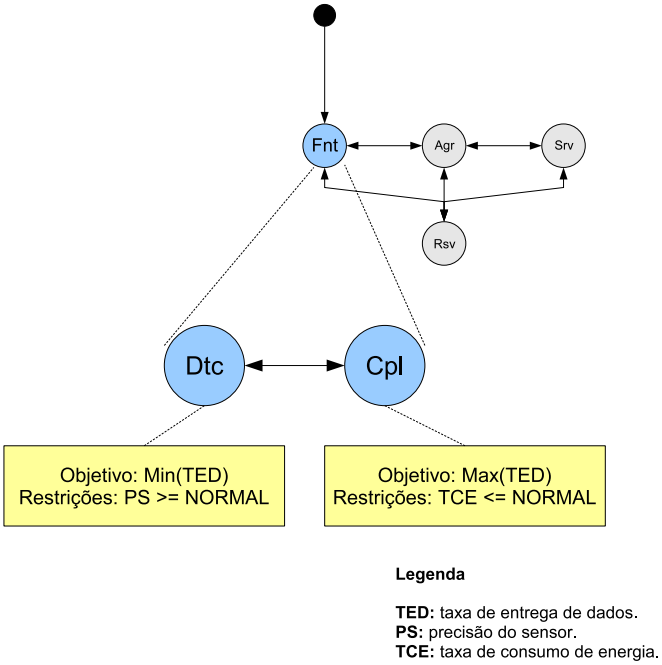

Figura 3.8: Grafo de alternativas referente ao papel fonte (Fnt).

No grafo exibido acima, visualizam-se, também, as RpAs associadas das alternativas Dtc e Cpl. A alternativa Dtc tem o objetivo de minimizar o consumo de energia, ao enviar dados sumarizados. Isso é representado pela função objetivo, a qual minimiza a MQ taxa de entrega de dados (TED) ao nó agregador. Ao contrário dessa alternativa, a Cpl envia os dados na sua forma bruta, maximizando, assim, a TED. As restrições da Dtc indicam que haverá um nível de confiança mínimo nos dados sensoriados, enquanto as restrições da Cpl proíbem o aumento indiscrimado do consumo de energia do nó, evitando que ele vá à exaustão rapidamente.

Para cada uma dessas alternativas, há um fluxo associado. O fluxo associado à alternativa Dtc garante uma minimização da TED, ao empregar técnicas de envio apenas dos dados que se diferem daqueles enviados anteriormente, enquanto aumenta a confiança dos dados, usando fontes redundantes para a detecção de eventos, a saber: i) sensor de temperatura; ii) sensor de vibração; iii) dados de detecção enviados por outros nós. O fluxo associado à alternativa Cpl, por sua vez, envia todos os dados sensoriados, não incluindo

nenhum BF para compressão ou fusão desses dados, maximizando, assim, a TED. A taxa de amostragem é cuidadosamente controlada pelos BFs que interfaceiam com os sensores, de forma a não haver uma sobrecarga de dados na rede e, conseqüentemente, o esgotamento prematuro dos recursos de energia da rede. Essas decisões estão ilustradas nas figuras 3.9 e 3.10 seguintes.

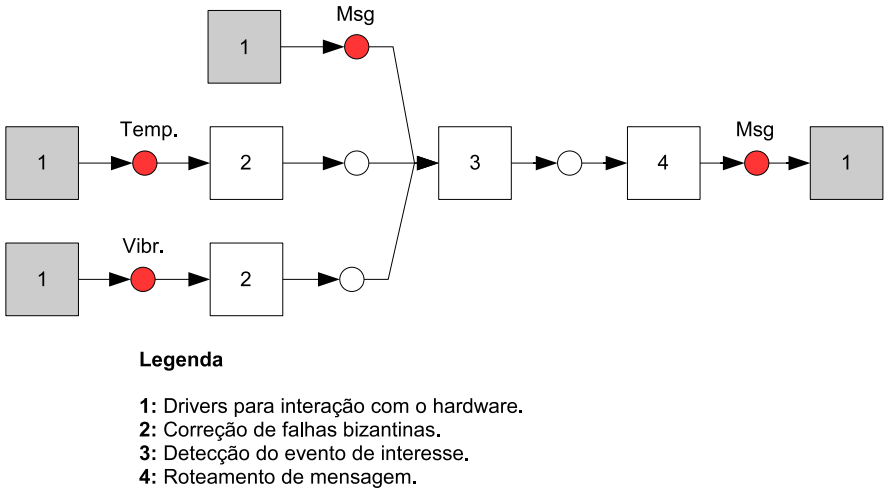

Figura 3.9: Fluxo referente à alternativa Detecção (Dtc).

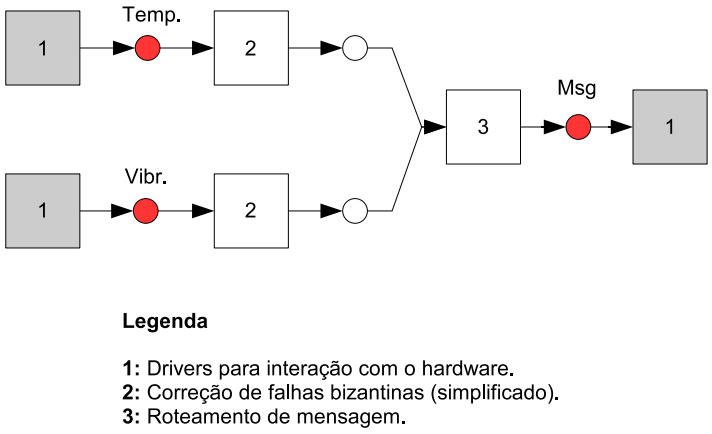

Figura 3.10: Fluxo referente à alternativa Completo (Cpl).

Capítulo 4

Conclusão

Nosso propósito foi o de desenvolver e apresentar uma arquitetura de adaptação, que se valha de uma nova metodologia para o tratamento da adaptação em RSSFs. Observou-se que, até então, a maioria das aplicações para RSSFs era projetada de forma *ad-hoc*, sem que houvesse uma estrutura que promovesse a organização do projeto e a reutilização de decisões arquiteturais no que tange à adaptação.

Durante o desenvolvimento deste trabalho, observaram-se vários pontos em que poderia haver um estudo mais aprofundado, a fim de se cobrir lacunas quanto à especificação completa de algoritmos, políticas e parâmetros que são utilizados recorrentemente na arquitetura. Eles serão listados e discutidos a seguir.

4.1 Trabalhos futuros

Aponto abaixo algumas questões que ficaram em aberto quanto à arquitetura e que merecem ser investigadas:

Escolha de vértices equivalentes Ao escolher um papel ou uma alternativa corrente no grafo correspondente, pode haver vários vértices alcançáveis para serem avaliados. Por simplicidade, a política sugerida é a escolha aleatória de um desses vértices, mas deve haver políticas de escolha mais inteligentes. Isso levaria a um trabalho comparativo, o qual poderia revelar quais políticas são viáveis e em quais situações/aplicações.

Avaliação da arquitetura nos nós sensores - Apesar de a arquitetura ser concebida levando-se em conta as restrições dos nós sensores, é interessante um trabalho de investigação sobre os impactos da adoção da arquitetura sobre o código-fonte e a execução deste código em uma RSSF não-simulada. Há pelo menos três aspectos essenciais a serem considerados: espaço ocupado em memória, principal e secundária; desempenho computacional; custo em energia.

Arcabouço de desenvolvimento - Da forma como foi proposto, fica a cargo do programador a transformação do modelo criado em código-fonte compilável para uma plataforma de *hardware* adequada às RSSFs. Entretanto, há regras para essa transformação que, se não forem seguidas, levam a um código que destoa da modelagem realizada. Ao aplicar essas regras em vários estudos de caso, é interessante incorporá-las em um programa que faça a geração automática do esqueleto do código-fonte a partir do modelo. Dessa forma, evita-se erros e facilita-se o trabalho do programador.

Avaliação de algoritmos de adaptação em conjunto Na literatura vigente, os trabalhos propostos raramente são analisados em conjunto com outros tipos de adaptação em uma aplicação. Portanto, não fica claro quais são as interações entre essas adaptações, que provavelmente ocorrem em toda aplicação para RSSF. A aplicação dessa arquitetura é um caminho para unir distintas adaptações em uma aplicação, e, assim, observar suas interações.

Referências Bibliográficas

[Abrach et al., 2003] Abrach, H.; Bhatti, S.; Carlson, J.; Dai, H.; Rose, J.; Sheth, A.; Shucker, B.; Deng, J. e Han, R. (2003). MANTIS: System support for multimodal networks of in-situ sensors. In *2nd ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, pp. 50–59, San Diego, California, EUA. ACM Press.

[Akyildiz et al., 2002] Akyildiz, I. F.; Su, W.; Sankarasubramaniam, Y. e Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422.

[ATMEL, 2003] ATMEL (2003). ATMEGA128L datasheet. `http://www.ortodoxism.ro/datasheets/atmel/2467S.pdf`.

[Avancha, 2005] Avancha, S. (2005). *A Holistic Approach to Secure Sensor Networks*. PhD thesis, University of Maryland.

[Bellis et al., 2005] Bellis, S. J.; Delaney, K.; O'Flynn, B.; Barton, J.; Razeeb, K. M. e O'Mathuna, S. C. (2005). Development of field programmable modular wireless sensor network nodes for ambient systems. *Computer Communications*, 28(13):1531–1544.

[Burrell et al., 2004] Burrell, J.; Brooke, T. e Beckwith, R. (2004). Vineyard computing: Sensor networks in agricultural production. *IEEE Pervasive Computing*, 03(1):38–45.

[Caldas, 2004] Caldas, W. S. (2004). *Tolerância a falhas adaptativa para robôs móveis com arquitetura híbrida*. PhD thesis, Departamento de Ciência da Computação da Universidade Federal de Minas Gerais, Belo Horizonte, MG.

[Carbunar et al., 2004] Carbunar, B.; Grama, A.; Vitek, J. e Carbunar, O. (2004). Redundancy and coverage detection in sensor networks. In *First IEEE International conference on Sensor and Ad Hoc Communications and Networks (SECON'04)*.

[Cerpa e Estrin, 2004] Cerpa, A. e Estrin, D. (2004). ASCENT: Adaptive self-configuring sensor networks topologies. *IEEE Transactions on Mobile Computing Special Issue on Mission-Oriented Sensor Networks*, 3(3).

[Chen et al., 2001] Chen, B.; Jamieson, K.; Balakrishnan, H. e Morris, R. (2001). Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *7th International Conference on Mobile Computing and Networking (MOBICOM'01)*, Rome, Italy.

[Clauberg, 2004] Clauberg, R. (2004). RFID and sensor networks: From sensor/actuator to business application. In *RFID Workshop, University of St. Gallen, Switzerland*.

[Coman et al., 2005] Coman, A.; Nascimento, M. A. e Sander, J. (2005). Exploiting redundancy in sensor networks for energy efficient processing of spatiotemporal region queries. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 187–194, New York, NY, USA. ACM Press.

[Culler e Mulder, 2004] Culler, D. E. e Mulder, H. (2004). Smart sensors to network the world. *Scientific American*, pp. 84–91.

[de Oliveira, 2005] de Oliveira, H. A. B. F. (2005). Um algoritmo recursivo de localização para redes de sensores sem fio. Master's thesis, Universidade Federal de Minas Gerais - Departamento de Ciência da Computação.

[Demmer et al., 2005] Demmer, M.; Levis, P.; Joki, A.; Brewer, E. e Culler, D. (2005). Tython: a dynamic simulation environment for sensor networks. Technical Report UCB/CSD-05-1372, EECS Department, University of California, Berkeley.

[Deshpande et al., 2005] Deshpande, A.; Guestrin, C. e Madden, S. (2005). Resource-aware wireless sensor-actuator networks. *IEEE Data Eng. Bull.*, 28(1):40–47.

[Deshpande et al., 2004] Deshpande, A.; Guestrin, C. e Madden, S. R. (2004). Model-driven data acquisition in sensor networks. In *In Proceedings of the 30th VLDB Conference*.

[Finchelstein, 2005] Finchelstein, D. F. (2005). Design considerations for ultra-low energy wireless microsensor nodes. *IEEE Transactions on Computers*, 54(6):727–740.

[Frank e Römer, 2005] Frank, C. e Römer, K. (2005). Algorithms for generic role assignment in wireless sensor networks. In *Proceedings of the 3rd international conference on Embedded networked sensor systems (SenSys'05)*, pp. 230–242, New York, NY, USA. ACM Press.

[Gao et al., 2003] Gao, Y.; Wu, K. e Li, F. (2003). Analysis on the redundancy of wireless sensor networks. In *WSNA '03: Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, pp. 108–114, New York, NY, USA. ACM Press.

[Habib et al., 2004] Habib, E.; Camara, D. e Loureiro, A. A. F. (2004). ICA: Um novo algoritmo de roteamento para redes de sensores. In *XXII Simpósio Brasileiro de Redes de Computadores*, pp. 147–160.

[He et al., 2004] He, T.; Krishnamurthy, S.; Stankovic, J. A.; Abdelzaher, T.; Luo, L.; Stoleru, R.; Yan, T. e Gu, L. (2004). Energy-efficient surveillance system using wireless sensor networks. In *2nd International Conference on Mobile Systems, Applications, and Services (MobiSys)*.

[He et al., 2003] He, Y.; Raghavendra, C. S.; Berson, S. e Braden, B. (2003). A programmable routing framework for autonomic sensor networks. *Autonomic Computing Workshop*, 00:60.

[Heinzelman et al., 1999] Heinzelman, W. R.; Kulik, J. e Balakrishnan, H. (1999). Adaptive protocols for information dissemination in wireless sensor networks. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pp. 174–185, New York, NY, USA. ACM Press.

[Hill e Culler, 2001] Hill, J. e Culler, D. (2001). A wireless-embedded architecture for system level optimization. Technical report, UC Berkeley, Berkeley, USA.

[Hill et al., 2000] Hill, J.; Szewczyk, R.; Woo, A.; Hollar, S.; Culler, D. e Pister, K. (2000). System architecture directions for networked sensors. In *Proceedings of the ninth international conference on Architectural support for programming languages and operating systems*, pp. 93–104, Cambridge, Massachusetts, Estados Unidos. ACM Press.

[Horn, 2001] Horn, P. (2001). Autonomic computing: IBM's perspective on the state of information technology. *IBM White Paper*.

[Intanagonwiwat et al., 2003] Intanagonwiwat, C.; Govindan, R.; Estrin, D.; Heidemann, J. e Silva, F. (2003). Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. Netw.*, 11(1):2–16.

[Iyengar et al., 1994] Iyengar, S. S.; Jayasimha, D. N. e Nadig, D. (1994). A versatile architecture for the distributed sensor integration problem. *IEEE Trans. Comput.*, 43(2):175–185.

[Jacob e Mathai, 2004] Jacob, C. e Mathai, R. J. (2004). Fault tolerance in sensor networks: A survey of fault tolerant sensor network algorithms and techniques. Manuscript at University of Wisconsin, Madison.

[Jain e Chang, 2004] Jain, A. e Chang, E. Y. (2004). Adaptive sampling for sensor networks. In *Proceedings of the 1st international workshop on Data management for sensor networks (DMSN '04)*, Toronto, Canada.

[Juang et al., 2002] Juang, P.; Oki, H.; Wang, Y.; Martonosi, M.; Peh, L.-S. e Rubenstein, D. (2002). Energy efficient computing for wildlife tracking: Design tradeoffs and early experiences with ZebraNet. In *Tenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X)*, San Jose, CA, USA.

[Karp e Kung, 2000] Karp, B. e Kung, H. T. (2000). Gpsr: greedy perimeter stateless routing for wireless networks. In *MobiCom '00: Proceedings of the 6th annual international*

conference on Mobile computing and networking, pp. 243–254, New York, NY, USA. ACM Press.

[Koushanfar et al., 2002] Koushanfar, F.; Potkonjak, M. e Sangiovanni-Vincentelli, A. (2002). Fault tolerance techniques for wireless ad hoc sensor networks. In *Proceedings of the First International Conference on Sensors*, pp. 1491–1496.

[Krishnamachari e Iyengar, 2003] Krishnamachari, B. e Iyengar, S. S. (2003). Efficient and fault-tolerant feature extraction in wireless sensor networks. In *IPSN*, pp. 488–501.

[Lachenmann, 2006] Lachenmann, A. (2006). TinyCubus: Applications. `http://www.ipvs.uni-stuttgart.de/abteilungen/vs/forschung/projekte/tinycubus/applications`.

[Levis e Culler, 2002] Levis, P. e Culler, D. (2002). Maté: a tiny virtual machine for sensor networks. In *ASPLOS-X: Proceedings of the 10th international conference on Architectural support for programming languages and operating systems*, pp. 85–95, New York, NY, USA. ACM Press.

[Levis et al., 2004] Levis, P.; Gay, D. e Culler, D. (2004). Bridging the gap: Programming sensor networks with application specific virtual machines. Technical Report UCB/CSD-04-1343, EECS Department, University of California, Berkeley.

[Levis et al., 2003] Levis, P.; Lee, N.; Welsh, M. e Culler, D. (2003). TOSSIM: accurate and scalable simulation of entire tinyos applications. In *Proceedings of the first international conference on Embedded networked sensor systems*, pp. 126–137, Los Angeles, California, Estados Unidos. ACM Press.

[Liu e Li, 2003] Liu, J. e Li, B. (2003). Distributed topology control in wireless sensor networks with asymmetric links. In *Proc. IEEE Globecom Wireless Communications Symp.*

[Macedo, 2006] Macedo, D. F. (2006). Um protocolo de roteamento para redes de sensores sem fio adaptável por regras de aplicação. Master's thesis, Universidade Federal de Minas Gerais - Departamento de Ciência da Computação, Belo Horizonte, Minas Gerais.

[Mainwaring et al., 2002] Mainwaring, A.; Polastre, J.; Szewczyk, R.; Culler, D. e Anderson, J. (2002). Wireless sensor networks for habitat monitoring. *First ACM Workshop on Wireless Sensor Networks and Applications.*

[Marrón et al., 2005a] Marrón, P. J.; Lachenmann, A.; Minder, D.; Gauger, M.; Saukh, O. e Rothermel, K. (2005a). Management and configuration issues for sensor networks. *International Journal of Network Management – Special Issue: Wireless Sensor Networks*, 15(4):235–253.

[Marrón et al., 2005b] Marrón, P. J.; Saukh, O.; Krüger, M. e Große, C. (2005b). Sensor network issues in the Sustainable Bridges project. In *European Projects Session of the Second European Workshop on Wireless Sensor Networks (EWSN 2005)*.

[Marzullo, 1989] Marzullo, K. (1989). Implementing fault tolerant sensors. Technical Report TR89-997, Department of Computer Science, Cornell University.

[Mini et al., 2005] Mini, R. A. F.; do Val Machado, M.; Loureiro, A. A. F. e Nath, B. (2005). Prediction-based energy map for wireless sensor networks. In *The Ad Hoc Networks Journal*, volume 3, pp. 235–253.

[Nakamura, 2003] Nakamura, F. G. (2003). Planejamento dinâmico para controle de cobertura e conectividade em redes de sensores sem fio planas. Master's thesis, Universidade Federal de Minas Gerais - Departamento de Ciência da Computação.

[Nath et al., 2004] Nath, S.; Gibbons, P. B.; Seshan, S. e Anderson, Z. R. (2004). Synopsis diffusion for robust aggregation in sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 250–262, New York, NY, USA. ACM Press.

[Ong et al., 2000] Ong, K. Y.; Longworth, T. L. e Barnhouse, J. L. (2000). Domestic preparedness program: Testing of APD200 chemical warfare agente detector against chemical warfare agents. Technical Report RT.DCC.013/2005, Engineering Department, University of Maryland.

[Parker, 1998] Parker, L. E. (1998). Alliance: an architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240.

[Quintão et al., 2004] Quintão, F.; Nakamura, F. G. e Mateus, G. R. (2004). Modelo exato e algoritmo híbrido para controle de topologia de redes de sensores. In *XXXVI Simpósio Brasileiro de Pesquisa Operacional*, São João del-Rei, Minas Gerais.

[Ruiz, 2003] Ruiz, L. B. (2003). *MANNA: A Management Architecture for Wireless Sensor Networks*. PhD thesis, Departamento de Ciência da Computação da Universidade Federal de Minas Gerais, Belo Horizonte, MG.

[Sadler et al., 2005] Sadler, C. M.; Kant, L. e Chen, W. (2005). Cross-layer self-healing mechanisms in wireless networks. In *Proceedings of the World Wireless Congress 2005*.

[Schurgers et al., 2002] Schurgers, C.; Tsiatsis, V.; Ganeriwal, S. e Srivastava, M. (2002). Optimizing sensor networks in the energy-latency-density design space. *IEEE Transactions on Mobile Computing*, 1(1):70–80.

[Szewczyk et al., 2004] Szewczyk, R.; Polastre, J.; Mainwaing, A. e Culler, D. (2004). Lessons from a sensor network expedition. *European Workshop on Sensor Networks (EWSN)*.

[Texas Instruments, 2000] Texas Instruments (2000). MSP430f149 datasheet. `http://www.ortodoxism.ro/datasheets/texasinstruments/msp430f149.pdf`.

[Tian e Georganas, 2002] Tian, D. e Georganas, N. D. (2002). A coverage-preserving node scheduling scheme for large wireless sensor networks. In *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pp. 32–41, New York, NY, USA. ACM Press.

[van Hoesel et al., 2003] van Hoesel, L. F.; Dulman, S. O.; Havinga, P. J. e Kip, H. J. (2003). Design of a low-power testbed for wireless sensor networks and verification. Technical Report TR-CTIT-03-45, University of Twente, Enschede, The Netherlands.

[Vieira et al., 2005] Vieira, L. F. M.; Vitorino, B. A. D.; Ruiz, L. B. e Fernandes, A. O. (2005). Desenvolvimento de aplicações para redes de sensores sem fio. Technical Report RT.DCC.013/2005, Departamento de Ciência da Computação, Universidade Federal de Minas Gerais.

[Vieira, 2004] Vieira, M. A. M. (2004). BEAN: Uma plataforma computacional para redes de sensores sem fio. Master's thesis, Universidade Federal de Minas Gerais - Departamento de Ciência da Computação, Belo Horizonte, MG.

[Vieira et al., 2003] Vieira, M. A. M.; Vieira, L. F. M.; Ruiz, L. B.; Loureiro, A. A. F.; Fernandes, A. O.; Nogueira, J. M. S. e da Silva Jr., D. C. (2003). Como obter o mapa de energia em rede de sensores sem fio? uma abordagem tolerante a falhas. In *WCSF'03: V Workshop de Comunicações sem Fio e Computação Móvel*.

[Voigt et al., 2003] Voigt, T.; Ritter, H. e Schiller, J. (2003). Utilizing solar power in wireless sensor networks. In *The 28th Annual IEEE Conference on Local Computer Networks (LCN 2003)*, Bonn/Königswinter, Germany.

[Weng et al., 2004] Weng, Y.; Kallakuri, S.; Liang, X.; Doboli, A.; Hong, S.; Robertazzi, T. e Doboli, S. (2004). Dynamic architecture adaptation to improve scalability of sensor networks: A case study for a smart sensor for face recognition. In *Proceedings of the Real Time Systems Symposium (RTSS'04)*, Lisboa, Portugal.

[Woo et al., 2003] Woo, A.; Tong, T. e Culler, D. (2003). Taming the underlying challenges of reliable multihop routing in sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pp. 14–27, New York, NY, USA. ACM Press.

[Zhou e Roure, 2006] Zhou, J. e Roure, D. D. (2006). Designing energy-aware adaptive routing for wireless sensor networks. In *Proceedings of The 6th International Conference on ITS Telecommunications*.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo