ANDRÉ DE SOUZA BARBOSA
FREDERICO MOREIRA MACHADO
SYLVIO CEZAR KOURY MUSOLINO FILHO

IMPLEMENTAÇÃO DO PROTOCOLO ZIGBEE PARA REDES DE SENSORES SEM FIO

Projeto de Formatura apresentado à disciplina PCS 2502 – Laboratório de Projeto de Formatura II, da Escola Politécnica da Universidade de São Paulo.

São Paulo
2004

ANDRÉ DE SOUZA BARBOSA
FREDERICO MOREIRA MACHADO
SYLVIO CÉZAR KOURY MUSOLINO FILHO

IMPLEMENTAÇÃO DO PROTOCOLO ZIGBEE PARA REDES DE SENSORES SEM FIO

Projeto de Formatura apresentado à disciplina PCS 2502 – Laboratório de Projeto de Formatura II, da Escola Politécnica da Universidade de São Paulo.

Orientadora:
Profª Dra. Regina Melo Silveira

São Paulo
2004

LISTA DE ABREVIATURAS E SIGLAS

CAP	- Contention Access Period
CCA	- Clear Channel Assessment
CFP	- Contention Free Period
CSMA-CA	- Carrier Sense Multiple Access – Collision Avoidance
CSMA-CD	- Carrier Sense Multiple Access – Collision Detection
FCS	- Frame Check Sequence
GTS	- Guaranteed Time Slot
MAC	- Medium Access Control
MCPS	- MAC common part sublayer
MLME	- MAC sublayer management entity
PAN	- Personal Area Network

GLOSSÁRIO

Beacon-enabled PAN
Uma PAN caracterizada pela sincronização realizada através do envio de beacons pelo coordenador.

Coordenador de PAN
Nó responsável pela criação, associação, desassociação, sincronização e outras atividades de controle na PAN que ele coordena.

Superframe
Intervalo de tempo compreendido entre a transmissão de dois beacons consecutivos em uma PAN habilitada com beacons (beacon-enabled PAN).

Non beacon-enabled PAN
Uma PAN onde a sincronização entre os nós e coordenador não se dá através de beacons. Uma forma de sincronização para esse tipo de PAN é a técnica de "polling" do coordenador.

ÍNDICE

INTRODUÇÃO

Tema

O presente projeto tem como tema o padrão Zig Bee descrito como padrão aberto pelo IEEE através do protocolo 802.15.4 e suas aplicações em redes de sensores sem fio.

Objetivo

O objetivo deste projeto baseia-se no estudo de redes sem fio, especificamente redes de sensores de baixo consumo de energia, que utilizam o padrão 802.15.4. Este objetivo pode ser dividido em duas partes.

A primeira parte consiste em criar uma documentação sobre o padrão do IEEE descrito no protocolo 802.15.4 e aplicações no padrão Zig Bee.

A segunda parte está direcionada a compor um simulador que emule algumas funções do protocolo mencionado, possibilitando a ilustração de aplicações em rede sem fio sobre essa plataforma.

O Zig Bee é um padrão para redes sem fio de baixo consumo. Foi feito um consórcio entre várias empresas líderes de mercado (Philips, Motorola, Honeywell, entre outras) com o objetivo de definir um padrão aberto que será utilizado em diversas aplicações. As características presentes neste padrão, como baixo consumo de energia, fácil instalação, pequenas dimensões dos dispositivos de rede e principalmente baixo custo colocam este produto em posição privilegiada em alguns mercados características, as constatações destes fatos são também objetivo deste projeto. A idéia do nosso grupo neste projeto de formatura é utilizar e analisar um padrão novo que estará em franca utilização no momento de nossa entrada no mercado de trabalho.

A tecnologia Zig Bee aplicada a redes de sensores tem grande potencial no mercado que se configura de padrões de redes sem fio.

Motivação

A motivação inicial do grupo foi a de aplicar uma tecnologia de redes sem fio a uma das diversas possíveis aplicações adequadas a ela. Colocado este ponto como meta, o grupo analisou tecnologias e aplicações correlacionadas.

Diante das possibilidades, tomou-se conhecimento do novo padrão que surgia(Zig Bee) no momento e suas características inéditas no mercado, muito aderentes a aplicações de redes de sensores sem fio.

Aliando a intenção de aprofundarmo-nos em um novo padrão e a possibilidade de criar meios para construir aplicações com um mercado em formação e com um futuro promissor decidiu-se atuar sobre o padrão Zig Bee e sua aplicabilidade em redes de sensores.

Analisados os objetivos que o projeto alcança dentro da disciplina a que este coloca-se como tarefa, traçou-se os produtos possíveis a serem realizados a partir do projeto e seu possível interesse às pessoas que pudessem ter o projeto nas mão.

A primeira parte do projeto que compreende uma descrição teórica do padrão, tem motivação de tornar o padrão mais popular e fortalecê-lo dentro do meio acadêmico e como conseqüência fortalecendo o seu mercado em geral.

A segunda parte que consiste de um simulador do protocolo 802.15.4, tem como motivação ilustrar o funcionamento do padrão para algumas aplicações e ratificar a sua viabilidade.

Organização

O projeto foi organizado em fases complementares.e fases independentes que podem ser executadas paralelamente.

Portanto em um primeiro momento fez-se um estudo minucioso das características do protocolo. Esta fase foi percorrida em conjunto por todo o grupo, pois mostra-se imprescindível o conhecimento de todos os participantes sobre o objeto de suas tarefas.

O segundo momento fez-se uma modelagem inicial do simulador, discutindo-se seus escopo e objetivos e conseqüentes características implicadas pelos objetivos.

Em seguia a esta análise preliminar o grupo dividiu as tarefas conforme os módulos da modelagem do simulador. Paralelamente a isso designou-se uma parte da equipe para constituir a documentação a qual é objetivo da primeira parte do projeto e análises de

aplicação e de mercado do padrão, com relação a tecnologia e aspectos financeiros. Todas estas etapas foram desenvolvidas com periódicas verificações de congruência entre as diversas partes do projeto.

No encerramento reuniu-se toda a documentação que resultou dos diversos desenvolvimentos e as agrupou neste documento com as devidas aproximações necessárias.

Aspectos Conceituais

A metodologia escolhida para executar o projeto foi uma metodologia incremental, pois desta maneira os módulos do projeto foram sendo desenvolvidos e na seqüência já testados em sua funcionalidade e seu enquadramento no contexto do projeto. Este método foi adequado, pois este projeto foi separado em módulos que deveriam ser feitos em paralelo. Constatou-se então com o emprego deste método que o re-trabalho empregado foi menor, pois os resultados apresentavam-se logo ao fim de cada módulo e os ajustes necessários de modelagem logo eram percebidos.

DESCRIÇÃO DO PROTOCOLO 802.15.4

Características Gerais:

O protocolo 802.15.4 está definido pelo IEEE(Institute of Eletrical and Eletronics Engineers), descrevendo as camadas MAC (Medium Access Control) e PHY(Physical) para LR-WPAN(Low-Rate Wireless Personal Área Network).

Os principais objetivos a serem alcançados na constituição de uma LR-WPAN são:

- Facilidade de instalação.
- Confiabilidade na transferência de dados.
- Baixo custo.
- Baterias com longa vida.

Colocados estes objetivos, arquitetou-se um protocolo para atender aos requisitos. As principais características do protocolo são:

- Taxas de transmissão ao "ar livre" de 250 Kb/s, 40 Kb/s e 20 Kb/s.
- Operações com a rede em hierarquia estrela ou "peer-to-peer".
- Endereçamento de 16 ou 64 bits.
- Alocação de GTSs(guaranteed time slots).
- Canal de acesso com CSMA-CA
- Protocolo preparado para transmissão de dados com confiabilidade.
- Baixo consumo de energia.
- Detecção de energia dos canais.
- Indicação da qualidade da conexão no recebimento de pacotes.
- 16 canais na banda de 2450 MHz.
- 10 canais na banda de 915 MHz.
- 1 canal na banda de 868 MHz.

Composição da WPAN:

Uma WPAN pode ser composta por dois diferentes tipos de dispositivos:

- full-function device(FFD).
- Reduced-function device(RFD).

O FFD pode operar como coordenador da "personal area network"(PAN) ou como um dispositivo comum da PAN, enquanto que o RFD pode operar apenas como um dispositivo comum da rede.

Para compor uma WPAN são necessários pelo menos 2 dispositivos que estejam dentro de uma mesma área de comunicação, mas pelo menos 1 dos dispositivos deve ser um FFD para exercer a função de coordenador da rede.

Topologia das redes:

A LR-WPAN pode operar em duas topologias diferentes:

- Topologia estrela.
- Topologia "peer-to-peer".

Na topologia estrela a comunicação é estabelecida dentro da PAN entre dispositivos comuns(que podem ser FFD ou RFD) e o coordenador de PAN(que deve ser um FFD).
O coordenador de PAN exerce funções de coordenação de associação e dissociação da PAN, além de "rotear" os dados através da rede. Os participantes da PAN devem ter um endereço de 64 bits.Nesta topologia os dispositivos comuns só podem estabelecer comunicação com os seus coordenadores.
Na topologia "peer-to-peer" qualquer dispositivo pode comunicar-se com qualquer outro, desde que este esteja no seu raio de alcance.
Os FFDs podem comunicar-se com FFDs ou RFDs, enquanto que os RFDs podem comunicar-se apenas com FFDs.

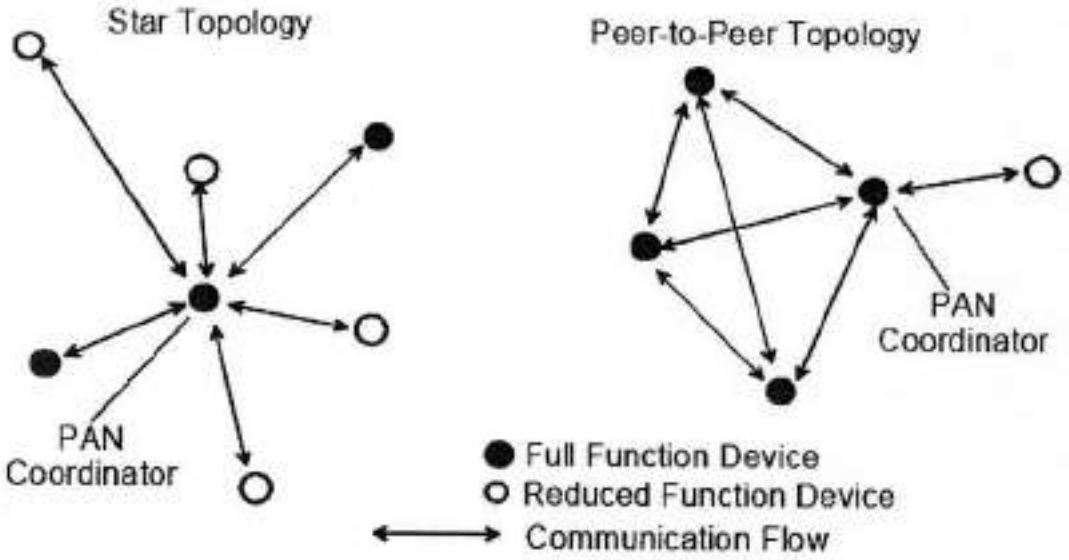


Figura1-Topologia das redes.

Formação das redes:

A formação das redes deve ser trabalhada na camada de rede, que não é especificada no protocolo 802.15.4, mas o documento sugere como poderiam ser essas associações.

Formação de uma rede estrela:

Depois de ativado um FFD, ele pode criar a sua própria PAN e ser o seu coordenador. Ele deve escolher um identificador de PAN, que é único entre as PANs que estão em contato por meio do radio. Isso possibilita que as redes estrela operem independentemente umas das outras. A partir deste momento, o coordenador da PAN pode autorizar que os FFDs ou RFDs que pedirem entrem em sua rede.

Formação de uma rede "peer-to-peer":

Na topologia "peer-to-peer" todos dispositivos podem comunicar-se com quaisquer dispositivos ao seu alcance. A definição do coordenador de PAN segue o critério do primeiro que tentar uma comunicação com os outros é aclamado o coordenador, e depois delegando funções de "cluster head" para outros dispositivos, criando "sub-grupos".

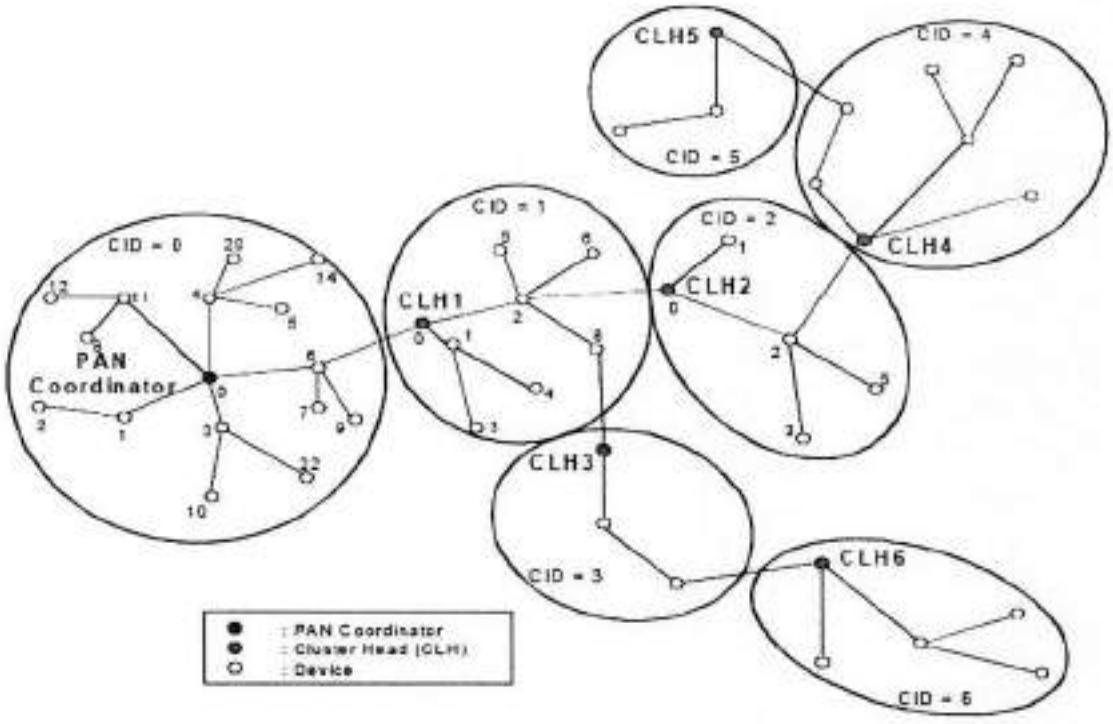


Figura2-Rede "peer-to-peer".

Camada Física(PHY):

A camada física definida no protocolo 802.15.4 exerce como funções principais:

- Ativar e desativar o dispositivo de radio.
- Detectar energia nos canais.
- Avaliar a qualidade de conexão no recebimento de pacotes.
- Detectar ociosidade do canal de acesso (CCA) para o CSMA-CA.
- Seleção da freqüência de atuação do canal.
- Transmissão e recepção de dados.

Freqüência de operação:

Os dispositivos que operarem de acordo com o protocolo 802.15.4, deverão operar em alguma das seguintes freqüências:

PHY (MHz)	Frequency band (MHz)	Spreading parameters		Data parameters		
		Chip rate (kchip/s)	Modulation	Bit rate (kb/s)	Symbol rate (ksymbol/s)	Symbols
868/915	868–868.6	300	BPSK	20	20	Binary
	902–928	600	BPSK	40	40	Binary
2450	2400–2483.5	2000	O-QPSK	250	62.5	16-ary Orthogonal

Figura3- Bandas de Freqüência e velocidade de transmissão de dados.

Freqüências estas que estão de acordo com as especificações dos mercados japonês, canadense, europeu e americano.

Especificações de serviços da camada física:

A camada física é modelada logicamente segundo a figura abaixo:

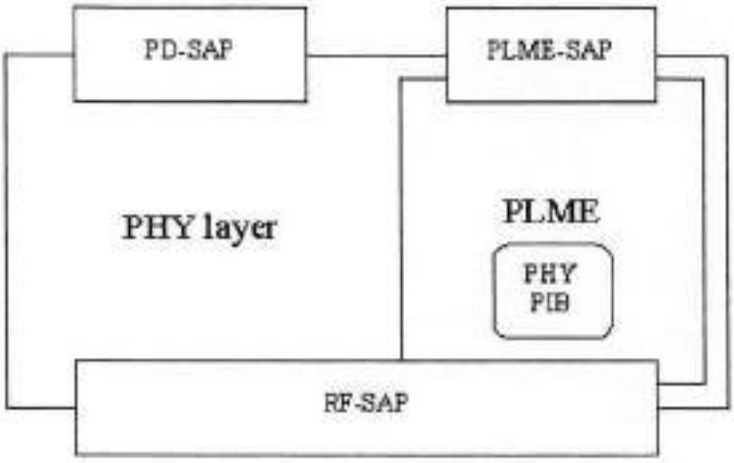


Figura4- Modelo da Camada Física.

Fornecendo a camada MAC dois tipos de serviço.

- Serviços de dados (Data Service).
- Serviços de gerenciamento (Managment Service).

Serviços de dados:

PD-DATA.request:

-primitiva utilizada para pedir a transferência de MPDU(PSDU) da camada MAC para a camada física local.

-essa primitiva tem os seguintes parâmetros:

Name	Type	Valid range	Description
psduLength	Unsigned Integer	≤ *aMaxPHYPacketSize*	The number of octets contained in the PSDU to be transmitted by the PHY entity.
psdu	Set of octets	—	The set of octets forming the PSDU to be transmitted by the PHY entity.

Figura 5- Parâmetros do PD-DATA.request.

-como conseqüência desse pedido o transmissor será ativado(TX_ON), a camada física irá compor um PPDU que envolve o PSDU e o transmitirá. Quando a transmissão for completada, esta entidade esperará uma primitiva PD-DATA.confirm com o estado de sucesso.

PD-DATA.confirm:

-confirma a transmissão de um MPDU da camada MAC de uma entidade para a camada MAC de outro ponto da rede.

-essa primitiva tem o seguinte parâmetro:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS, RX_ON, or TRX_OFF	The result of the request to transmit a packet.

Figura 6- Parâmetro do PD-DATA.confirm.

-essa primitiva é gerada como conseqüência de um envio de um MPDU e conforme o resultado da transmissão ela é gerada com o parâmetro status de SUCCESS, RX_ON(não pôde transmitir) ou TRX_OFF(não pôde transmitir).

PD-DATA.indication:

-indica a transmissão de um MPDU da camada MAC de uma entidade para a camada MAC de outro ponto da rede.

-essa primitiva tem os seguintes parâmetros:

Name	Type	Valid range	Description
psduLength	Unsigned Integer	≤ *aMaxPHYPacketSize*	The number of octets contained in the PSDU received by the PHY entity.
psdu	Set of octets	—	The set of octets forming the PSDU received by the PHY entity.
ppduLinkQuality	Integer	0 x 00–0 x ff	Link quality (LQ) value measured during reception of the PPDU (see 6.7.8).

Figura 7- Parâmetros do PD-DATA.indication.

-essa primitiva é gerada quando a camada física de uma entidade recebeu um PSDU e está pronto para passá-lo como MPDU a camada MAC da mesma entidade.

Serviços de gerenciamento:

-os comandos de gerenciamento são transportados entre o MLME e o PLME através do PLME-SAP.

PLME-CCA.request

-esta primitiva pede que o PLME execute um CCA (detecção de ociosidade de canal).

-não tem parâmetros.

-Esta primitiva ocorre quando o algoritmo de CSMA-CA no MLME pede avaliação do canal.

PLME-CCA.confirm

-esta primitiva é gerada como resposta a um PLME-CCA.request, e traz o resultado do teste de ociosidade do canal.

-os parâmetros são:

Name	Type	Valid range	Description
status	Enumeration	TRX_OFF, TX_ON, BUSY, or IDLE	The result of the request to perform a CCA.

Figura 8- Parâmetros do PLME-CCA.indication.

PLME-ED.request:

-essa primitiva é gerada a pedido do MLME para o PLME executar uma detecção de energia do canal.

-não tem parâmetros.

PLME-ED.confirm:

-essa primitiva traz o resultado da medição da energia do canal.

-os parâmetros são:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS, TRX_OFF, or TX_ON	The result of the request to perform an ED measurement.
EnergyLevel	Integer	0 x 00–0 x ff	ED level for the current channel.

Figura 9- Parâmetros do PLME-ED.confirm.

PLME-GET.request:

-essa primitiva pede informações sobre uma PHY PIB.

-é fruto de um pedido do MLME para o PLME sobre PHY PIB

-os parâmetros são:

Name	Type	Valid range	Description
PIBAttribute	Enumeration	See Table 19	The identifier of the PHY PIB attribute to get.

Figura 10-Parâmetros do PLME-GET.request.

PLME-GET.confirm:

-essa primitiva traz as informações pedidas pelo PLME-GET.request ao PLME.

-os parâmetros são:

Name	Type	Valid range	Description
Status	Enumeration	SUCCESS or UNSUPPORTED_ATTRIBUTE	The result of the request for PHY PIB attribute information.
PIBAttribute	Enumeration	See Table 19	The identifier of the PHY PIB attribute to get.
PIBAttributeValue	Various	Attribute specific	The value of the indicated PHY PIB attribute to get.

Figura 11-Parâmetros do PLME-GET.confirm.

PLME-SET-TRX-STATE.request:

-gerada quando o MLME pede ao PLME que mude o estado de operação do receptor.

-os parâmetros são:

Name	Type	Valid range	Description
state	Enumeration	RX_ON, TRX_OFF, FORCE_TRX_OFF, or TX_ON	The new state in which to configure the transceiver.

Figura 12- Parâmetros do PLME-SET-TRX-STATE.request

PLME-SET-TRX-STATE.confirm:

-traz o resultado do pedido de alteração do estado do receptor.

-os parâmetros:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS, RX_ON, TRX_OFF, TX_ON, BUSY_RX, or BUSY_TX	The result of the request to change the state of the transceiver.

Figura 13- Parâmetros do PLME-SET-TRX-STATE.confirm.

PLME-SET.request

-essa primitiva é gerada para fixar um PHY PIB para um valor dado.

-parâmetros:

Name	Type	Valid range	Description
PIBAttribute	Enumeration	See Table 19	The identifier of the PIB attribute to set.
PIBAttributeValue	Various	Attribute specific	The value of the indicated PIB attribute to set.

Figura 14- Parâmetros do PLME-SET.request.

PLME-SET.confirm

-essa primitiva é gerada como resultado e resposta à PLME-SET.request

-os parâmetros são:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS, UNSUPPORTED_ATTRIBUTE, or INVALID_PARAMETER	The status of the attempt to set the request PIB attribute.
PIBAttribute	Enumeration	See Table 19	The identifier of the PIB attribute being confirmed.

Figura 15- Parâmetros do PLME-SET.confirm.

Atributos da camada PHY:

Enumeration	Value	Description
BUSY	0 x 00	The CCA attempt has detected a busy channel.
BUSY_RX	0 x 01	The transceiver is asked to change its state while receiving.
BUSY_TX	0 x 02	The transceiver is asked to change its state while transmitting.
FORCE_TRX_OFF	0 x 03	The transceiver is to be switched off.
IDLE	0 x 04	The CCA attempt has detected an idle channel.
INVALID_PARAMETER	0 x 05	A SET/GET request was issued with a parameter in the primitive that is out of the valid range.
RX_ON	0 x 06	The transceiver is in or is to be configured into the receiver enabled state.
SUCCESS	0 x 07	A SET/GET, an ED operation, or a transceiver state change was successful.
TRX_OFF	0 x 08	The transceiver is in or is to be configured into the transceiver disabled state.
TX_ON	0 x 09	The transceiver is in or is to be configured into the transmitter enabled state.
UNSUPPORTED_ATTRIBUTE	0 x 0a	A SET/GET request was issued with the identifier of an attribute that is not supported.

Figura 16- Atributos da camada PHY.

Composição do pacote PPDU:

O pacote PPDU é formado por 3 componentes básicos:

- SHR, cabeçalho de sincronização.
- PHR, cabeçalho referente a camada PHY.
- PHY payload, dados que são transportados pela camada PHY, que serão significativos na camada MAC.

Octets: 4	1	1		variable
Preamble	SFD	Frame length (7 bits)	Reserved (1 bit)	PSDU
SHR		PHR		PHY payload

Figura 17- Fomato do PPDU.

O SHR contém:

- Preâmbulo, 32 bits destinados a sincronização com relação a uma mensagem que está por vir.
- SFD(start-of-frame delimiter), 8 bits destinados a indicar o fim do preâmbulo e começo do pacote de dados. Ver abaixo a composição.

Bits: 0	1	2	3	4	5	6	7
1	1	1	0	0	1	0	1

Figura 18 –Formato do campo SFD.

O PHR contém:

- Frame length, 7 bits que indicam o tamanho do PSDU em "octetos" que estará no PHY payload.

Frame length values	Payload
0–4	Reserved
5	MPDU (Acknowledgment)
6–7	Reserved
8 to *aMaxPHYPacketSize*	MPDU

Figura 19- Valores do Frame length.

- Reserved, 1 bit reservado.

O PHY payload contém:

- PSDU(PHY service data unit), dados do pacote PHY.

Constantes da camada PHY:

Esses dados são constantes residentes no hardware.

Constant	Description	Value
aMaxPHYPacketSize	The maximum PSDU size (in octets) the PHY shall be able to receive.	127
aTurnaroundTime	RX-to-TX or TX-to-RX maximum turnaround time (see 6.7.1 and 6.7.2)	12 symbol periods

Figura 20- Constantes da camada PHY.

Atributos do PIB(PAN information base):

Attribute	Identifier	Type	Range	Description
phyCurrentChannel	0 x 00	Integer	0–26	The RF channel to use for all following transmissions and receptions (see 6.1.2).
phyChannelsSupported	0 x 01	Bitmap	See description	The 5 most significant bits (MSBs) (b_{27}, ... b_{31}) of *phyChannelsSupported* shall be reserved and set to 0, and the 27 LSBs (b_0, b_1, ... b_{26}) shall indicate the status (1=available, 0=unavailable) for each of the 27 valid channels (b_k shall indicate the status of channel k as in 6.1.2).
phyTransmitPower	0 x 02	Bitmap	0 x 00–0xbf	The 2 MSBs represent the tolerance on the transmit power: 00 = ± 1 dB 01 = ± 3 dB 10 = ± 6 dB The 6 LSBs represent a signed integer in twos-complement format, corresponding to the nominal transmit power of the device in decibels relative to 1 mW. The lowest value of *phyTransmitPower* shall be interpreted as less than or equal to –32 dBm.
phyCCAMode	0 x 03	Integer	1–3	The CCA mode (see 6.7.9).

Figura 21- Atributos do PIB.

Camada MAC(medium access control):

A camada MAC definida no protocolo 802.15.4 exerce principalmente as seguintes funções:

- Gerar beacon para controle do fluxo de dados na rede se a entidade for um coordenador.
- Sincronizar a entidade aos beacons da rede.
- Associação e dissociação da PAN.
- Segurança.
- Emprego de mecanismo CSMA-CA para o canal de acesso.
- Emprego de mecanismo de GTS(guaranteed time slot).
- Viabilizar mecanismos de confiabilidade de transferência de pacotes.

Especificações de serviços da camada MAC:

A camada MAC é modelada logicamente segundo a figura abaixo:

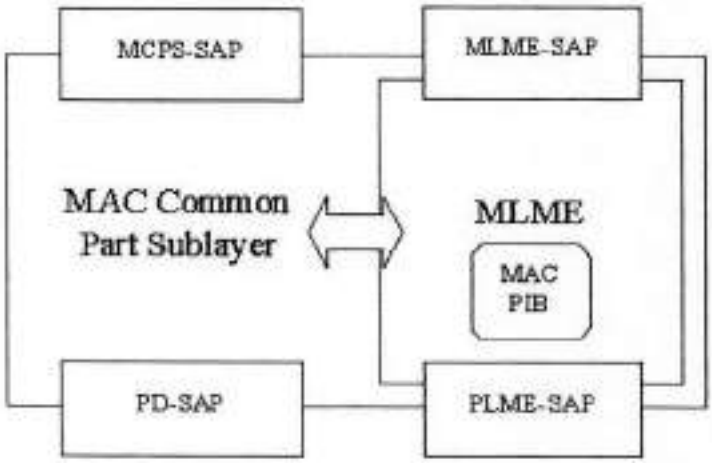


Figura 22- Modelo da camada MAC.

A camada MAC executa dois tipos de serviço:

- Serviços de dados
- Serviços de gerenciamento.

Os serviços de gerenciamento são executados pelo MLME (MAC sublayer management entity) e acessados pelo MLME-SAP.
Enquanto que os serviços de dados acontecem na MCPS (MAC common part sublayer) e acessados pelo MCPS-SAP.

Serviço de dados da camada MAC:

Os serviços de dados da camada MAC executa o transporte de SPDUs(SSCS(service specific convergence sublayer) protocol data units), que são as unidades de dados que serão enviados a camada de rede.
As primitivas dos serviços de dados da camada MAC são:

MCPS-DATA.request

-essa primitiva pede o envio de um SPDU da camada de rede local para uma camada de rede de outra entidade.

-os parâmetros são:

Name	Type	Valid range	Description
SrcAddrMode	Integer	0 x 00–0 x 03	The source addressing mode for this primitive and subsequent MPDU. This value can take one of the following values: 0 x 00 = no address (addressing fields omitted). 0 x 01 = reserved. 0 x 02 = 16 bit short address. 0 x 03 = 64 bit extended address.
SrcPANId	Integer	0 x 000–0 x ffff	The 16 bit PAN identifier of the entity from which the MSDU is being transferred.
SrcAddr	Device address	As specified by the SrcAddrMode parameter	The individual device address of the entity from which the MSDU is being transferred.
DstAddrMode	Integer	0 x 00–0 x 03	The destination addressing mode for this primitive and subsequent MPDU. This value can take one of the following values: 0 x 00 = no address (addressing fields omitted). 0 x 01 = reserved. 0 x 02 = 16 bit short address. 0 x 03 = 64 bit extended address.
DstPANId	Integer	0 x 0000–0 x ffff	The 16 bit PAN identifier of the entity to which the MSDU is being transferred.
DstAddr	Device address	As specified by the DstAddrMode parameter	The individual device address of the entity to which the MSDU is being transferred.
msduLength	Integer	≤ aMaxMACFrameSize	The number of octets contained in the MSDU to be transmitted by the MAC sublayer entity.
msdu	Set of octets	—	The set of octets forming the MSDU to be transmitted by the MAC sublayer entity.
msduHandle	Integer	0 x 00–0 x ff	The handle associated with the MSDU to be transmitted by the MAC sublayer entity.
TxOptions	Bitmap	0000 xxxx (where x can be 0 or 1)	The transmission options for this MSDU. These are a bitwise OR of one or more of the following: 0 x 01 = acknowledged transmission. 0 x 02 = GTS transmission. 0 x 04 = indirect transmission. 0 x 08 = security enabled transmission.

Figura 23- Parâmetros do MCPS-DATA.request.

MCPS-DATA.confirm:

-traz os resultados da tentativa de transmissão do SPDU.

-os parâmetros são:

Name	Type	Valid range	Description
msduHandle	Integer	0 x 00–0 x ff	The handle associated with the MSDU being confirmed.
status	Enumeration	SUCCESS, TRANSACTION_OVERFLOW, TRANSACTION_EXPIRED, CHANNEL_ACCESS_FAILURE, INVALID_GTS, NO_ACK, UNAVAILABLE_KEY, FRAME_TOO_LONG, FAILED_SECURITY_CHECK, or INVALID_PARAMETER	The status of the last MSDU transmission.

Figura 24- Parâmetros do MCPS-DATA.confirm.

MCPS-DATA.indication:

-essa primitiva indica a camada de rede que o SPDU foi transportado com sucesso à camada MAC.

-os parâmetros são:

Name	Type	Valid range	Description
SrcAddrMode	Integer	0 x 00–0 x 03	The source addressing mode for this primitive corresponding to the received MPDU. This value can take one of the following values: 0 x 00 = no address (addressing fields omitted). 0 x 01 = reserved. 0 x 02 = 16 bit short address. 0 x 03 = 64 bit extended address.
SrcPANId	Integer	0 x 0000–0 x ffff	The 16 bit PAN identifier of the entity from which the MSDU was received.
SrcAddr	Device address	As specified by the SrcAddrMode parameter	The individual device address of the entity from which the MSDU was received.
DstAddrMode	Integer	0 x 00–0 x 03	The destination addressing mode for this primitive corresponding to the received MPDU. This value can take one of the following values: 0 x 00 = no address (addressing fields omitted). 0 x 01 = reserved. 0 x 02 = 16 bit short device address. 0 x 03 = 64 bit extended device address.
DstPANId	Integer	0 x 0000–0 x ffff	The 16 bit PAN identifier of the entity to which the MSDU is being transferred.
DstAddr	Device address	As specified by the DstAddrMode parameter	The individual device address of the entity to which the MSDU is being transferred.
msduLength	Integer	≤ *aMaxMACFrame-Size*	The number of octets contained in the MSDU being indicated by the MAC sublayer entity.
msdu	Set of octets	—	The set of octets forming the MSDU being indicated by the MAC sublayer entity.
mpduLinkQuality	Integer	0 x 00–0 x ff	LQ value measured during reception of the MPDU. Lower values represent lower LQ (see 6.7.8).
SecurityUse	Boolean	TRUE or FALSE	An indication of whether the received data frame is using security. This value is set to TRUE if the security enable subfield was set to 1 or FALSE if the security enabled subfield was set to 0.
ACLEntry	Integer	0 x 00–0 x 08	The *macSecurityMode* parameter value from the ACL entry associated with the sender of the data frame. This value is set to 0 x 08 if the sender of the data frame was not found in the ACL.

Figura 25-Parâmetros do MCPS-DAT.indication.

MCPS-PURGE.request

-essa primitiva é utilizada para a camada de rede pedir à camada MAC que o MSDU indicado pelo "handle" que está na fila de transmissão, deve ser tirado desta.

-parâmetros:

Name	Type	Valid range	Description
msduHandle	Integer	0 x 00–0 x ff	The handle of the MSDU to be purged from the transaction queue.

Figura 26- Parâmetros do MCPS-PURGE.request.

MCPS-PURGE.confirm:

-essa primitiva é enviada da camada MAC para a camada de rede, indicando o resultado do pedido da primitiva MCPS-PURGE.request.

-parâmetros são:

Name	Type	Valid range	Description
msduHandle	Integer	0 x 00–0 x ff	The handle of the MSDU requested to be purge from the transaction queue.
status	Enumeration	SUCCESS or INVALID_HANDLE	The status of the request to be purged an MSDU from the transaction queue.

Figura 27- Parâmetros do MCPS-PURGE.confirm.

Seqüência de ações ilustrando a troca de mensagens de dados entre duas camadas MAC de dois pontos diferentes da rede:

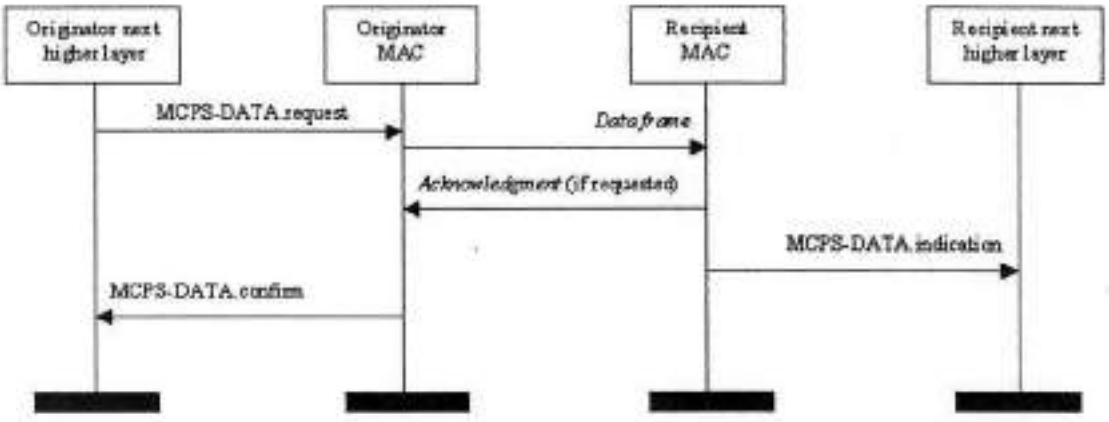


Figura 28- Troca de Mensagens de dado na camada MAC.

Serviços de Gerenciamento da camada MAC:

Primitivas de associação ao PAN:

Essas primitivas definem como um ponto se associa a um PAN.

MLME-ASSOCIATE.request:

-essa primitiva é usada para um ponto comum iniciar o processo de pedir a um coordenador de PAN a sua associação.Ela é gerada por uma camada de rede e enviada à camada MAC.

-os parâmetros são:

Name	Type	Valid range	Description
LogicalChannel	Integer	Selected from the available channels supported by the PHY	The logical channel on which to attempt association.
CoordAddrMode	Integer	0 x 02–0 x 03	The coordinator addressing mode for this primitive and subsequent MPDU. This value can take one of the following values: 2=16 bit short address. 3=64 bit extended address.
CoordPANId	Integer	0 x 0000–0 x ffff	The identifier of the PAN with which to associate.
CoordAddress	Device address	As specified by the CoordAddrMode parameter	The address of the coordinator with which to associate.
CapabilityInformation	Bitmap	See 7.3.1.1.2	Specifies the operational capabilities of the associating device.
SecurityEnable	Boolean	TRUE or FALSE	TRUE if security is enabled for this transfer or FALSE otherwise.

Figura 29- Parâmetros do MLME-ASSOCIATIVE.request.

MLME-ASSOCIATE.indication:

-essa primitiva é gerada pelo MLME de um coordenador de PAN e enviado para a sua camada de rede para indicar que recebeu o pedido de associação ao PAN de algum ponto não associado.

-os parâmetros são:

Name	Type	Valid range	Description
DeviceAddress	Device address	An extended 64 bit IEEE address	The address of the device requesting association.
CapabilityInformation	Bitmap	See 7.3.1.1.2	The operational capabilities of the device requesting association.
SecurityUse	Boolean	TRUE or FALSE	An indication of whether the received MAC command frame is using security. This value is set to TRUE if the security enable subfield was set to 1 or FALSE if the security enabled sub-field was set to 0.
ACLEntry	Integer	0 x 00– 0 x 08	The *macSecurityMode* parameter value from the ACL entry associated with the sender of the data frame. This value is set to 0 x 08 if the sender of the data frame was not found in the ACL.

Figura 30- Parâmetros do MLME-ASSOCIATIVE.indication.

MLME-ASSOCIATE.response:

-essa primitiva é gerada pela camada de rede do coordenador de PAN e enviada à sua camada MAC como resposta do MLME-ASSOCIATE.indication.

-os parâmetros são:

Name	Type	Valid range	Description
DeviceAddress	Device address	An extended 64 bit IEEE address	The address of the device requesting association.
AssocShortAddress	Integer	0 x 0000–0 x ffff	The short device address allocated by the coordinator on successful association. This parameter is set to 0xffff if the association was unsuccessful.
status	Enumeration	See 7.3.1.2.3	The status of the association attempt.
SecurityEnable	Boolean	TRUE or FALSE	TRUE if security is enabled for this transfer or FALSE otherwise.

Figura 31- Parâmetros do MLME-ASSOCIATIVE.response.

MLME-ASSOCIATE.confirm:

-essa primitiva é gerada pelo MLME do ponto não associado e enviada à sua camada de rede, como resposta ao seu pedido de associação a um PAN através do MLME-ASSOCIATE.request.

-os parâmetros são:

Name	Type	Valid range	Description
AssocShortAddress	Integer	0 x 0000–0 x ffff	The short device address allocated by the coordinator on successful association. This parameter will be equal to 0 x ffff if the association attempt was unsuccessful.
status	Enumeration	The value of the status field of the associate response command (see 7.3.1.2.3). SUCCESS, CHANNEL_ACCESS_FAILURE, NO_ACK, NO_DATA, UNAVAILABLE_KEY, FAILED_SECURITY_CHECK or INVALID_PARAMETER.	The status of the association attempt.

Figura 32- Parâmetros do MLME-ASSOCIATIVE.confirm.

A figura abaixo ilustra a seqüência de primitivas trocadas no processo de associação de um ponto comum a um PAN.

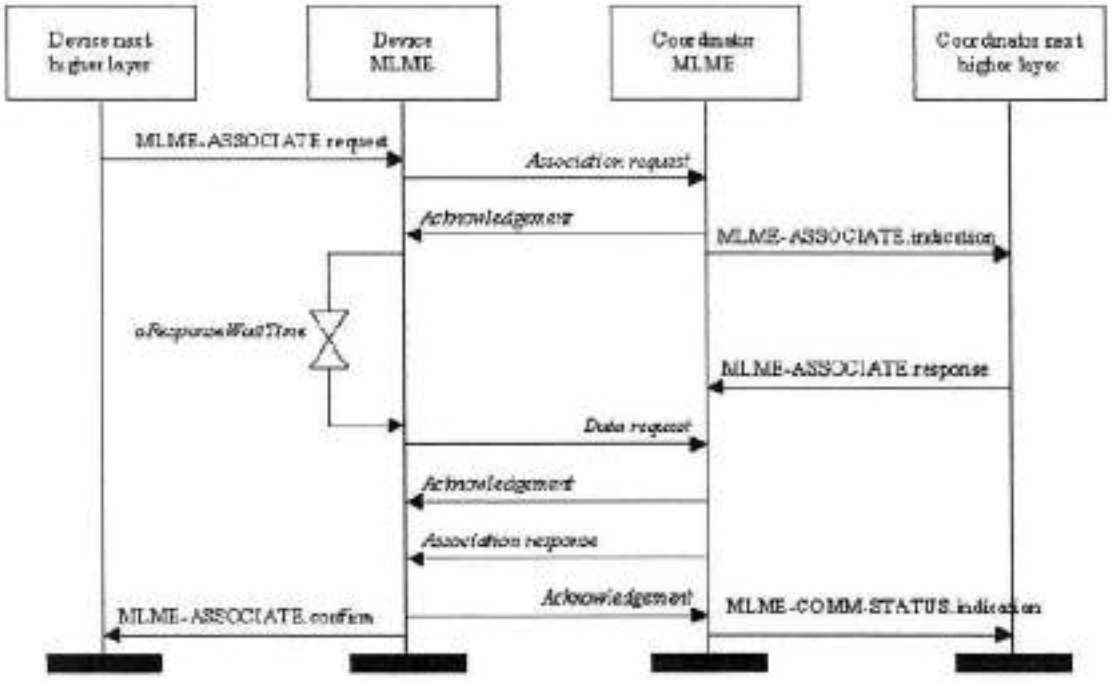


Figura 33- Troca de primitivas para associação.

Primitivas de dissociação do PAN:

Essas primitivas definem como um ponto associado a um PAN pode se dissociar deste.

MLME-DISASSOCIATE.request:

-essa primitiva é gerada por uma camada de rede de um ponto associado a um PAN, que pretende se dissociar deste, essa primitiva é enviada ao MLME do mesmo.

-essa primitiva pode ser gerada também pela camada de rede de um coordenador de PAN e enviada ao seu MLME com o intuito de excluir algum ponto associado ao PAN.

-os parâmetros são:

Name	Type	Valid range	Description
DeviceAddress	Device address	An extended 64 bit IEEE address.	The address of the device to which to send the disassociation notification command.
DisassociateReason	Integer	0 x 00–0 x ff	The reason for the disassociation (see 7.3.1.3.2).
SecurityEnable	Boolean	TRUE or FALSE	TRUE if security is enabled for this transfer or FALSE otherwise.

Figura 34- Parâmetros do MLME-DISASSOCIATIVE.request.

MLME-DISASSOCIATE.indication:

-essa primitiva é gerada pelo MLME ao receber uma notificação de dissociação de um ponto e é enviada à sua camada de rede.

-os parâmetros são:

Name	Type	Valid range	Description
DeviceAddress	Device address	An extended 64 bit IEEE address	The address of the device requesting disassociation.
DisassociateReason	Integer	0 x 00–0 x ff	The reason for the disassociation (see 7.3.1.3.2).
SecurityUse	Boolean	TRUE or FALSE	An indication of whether the received MAC command frame is using security. This value is set to TRUE if the security enable subfield was set to 1 or FALSE if the security enabled subfield was set to 0.
ACLEntry	Integer	0 x 00–0 x 08	The *macSecurityMode* parameter value from the ACL entry associated with the sender of the data frame. This value is set to 0x08 if the sender of the data frame was not found in the ACL.

Figura 35- Parâmetros do MLME-DISASSOCIATIVE.indication.

MLME-DISASSOCIATE.confirm:

-essa primitiva é gerada pelo MLME de um ponto que começou um processo de dissociação do PAN (pode ser o MLME do coordenador, quando este está pedindo a dissociação de um ponto do seu PAN ou pode ser o MLME do próprio ponto que está pedindo a sua dissociação do PAN ao qual está associado) e enviada à sua camada de rede como resposta a MLME-DISASSOCIATE.request que esta tinha enviado.

-os parâmetros são:

Name	Type	Valid range	Description
Status	Enumeration	SUCCESS, TRANSACTION_OVERFLOW, TRANSACTION_EXPIRED, NO_ACK, CHANNEL_ACCESS_FAILURE, UNAVAILABLE_KEY, FAILED_SECURITY_CHECK, or INVALID_PARAMETER	The status of the disassociation attempt.

Figura 36- Parâmetros do MLME-DISASSOCIATIVE.confirm.

A figura abaixo ilustra a seqüência de primitivas trocadas no processo de dissociação de um ponto comum a um PAN a que este pertence:

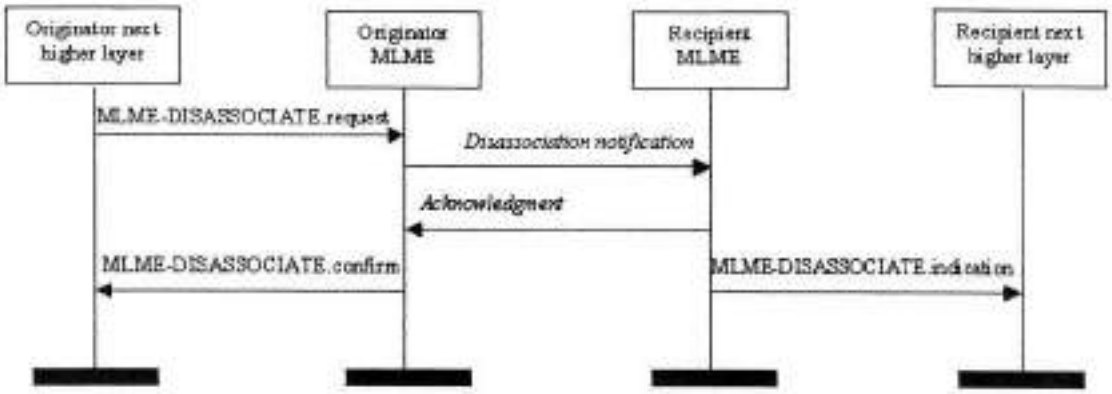


Figura 37- Seqüência de primitivas para a dissociação.

Primitivas de notificação de Beacon (Frames especiais que são enviados pelo coordenador de uma PAN):

Essas primitivas tem o objetivo de definir como um ponto comum pode ser notificado quando um beacon é recebido.

MLME-BEACON-NOTIFY.indication:

-essa primitiva é gerada no MLME e enviada à camada de rede do ponto, quando este recebe um beacon (gerado pelo coordenador do seu PAN).Nesta primitiva são passados parâmetros do beacon.

-os parâmetros são:

Name	Type	Valid range	Description
BSN	Integer	0 x 00–0 x ff	The beacon sequence number.
PANDescriptor	PANDescriptor value	See Table 41	The PANDescriptor for the received beacon.
PendAddrSpec	Bitmap	See 7.2.2.1.6	The beacon pending address specification.
AddrList	List of device addresses	—	The list of addresses of the devices for which the beacon source has data.
sduLength	Integer	0 – *aMaxBeaconPayloadLength*	The number of octets contained in the beacon payload of the beacon frame received by the MAC sublayer.
sdu	Set of octets	—	The set of octets comprising the beacon payload to be transferred from the MAC sublayer entity to the next higher layer.

Figura 38- Parâmetros do MLME-BEACON-NOTIFY.indication.

Primitivas para obter atributos de PIB:

MLME-GET.request:

-essa primitiva é gerada pela camada de rede de um ponto e enviada a camada MAC do mesmo para obter informações sobre o PIB MAC(seu PAN).

-os parâmetros são:

Name	Type	Valid range	Description
PIBAttribute	Integer	See Table 71 and Table 72	The identifier of the PIB attribute to read.

Figura 39- Parâmetros do MLME-GET.request.

MLME-GET.confirm:

-essa primitiva é gerada pela camada MAC como resposta a MLME-GET.request.

-os parâmetros são:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS or UNSUPPORTED_ATTRIBUTE	The result of the request for MAC PIB attribute information.
PIBAttribute	Integer	See Table 71 and Table 72	The identifier of the MAC PIB attribute that was read.
PIBAttributeValue	Various	Attribute specific: see Table 71 and Table 72	The value of the indicated MAC PIB attribute that was read.

Figura 40- Parâmetros do MLME-GET.confirm.

Primitivas de gerenciamento de GTS:

Essas primitivas definem como o GTS tem sua aplicação requerida. Os pontos que quiserem utilizar o mecanismo de GTS já devem estar acompanhando os beacon que são gerados e enviados pelo seu coordenador de Pan.

MLME-GTS.request:

-essa primitiva é gerada pela camada de rede de um ponto que quer ter um GTS alocado ou desalocado pelo seu coordenador de PAN, então esta primitiva é enviada ao seu MLME.

-os parâmetros são:

Name	Type	Valid range	Description
GTSCharacteristics	GTS characteristics	See 7.3.3.1.2	The characteristics of the GTS request.
SecurityEnable	Boolean	TRUE or FALSE	TRUE if security is enabled for this transfer or FALSE otherwise.

Figura 41- Parâmetros do MLME-GTS.request.

MLME-GTS.confirm:

-essa primitiva é gerada pelo MLME como resultado do processo iniciado pelo MLME-GTS.request, e oferece à camada de rede os resultados deste pedido.

-os parâmetros são:

Name	Type	Valid range	Description
GTSCharacteristics	GTS characteristics	See 7.3.3.1.2	The characteristics of the GTS.
status	Enumeration	SUCCESS, DENIED, NO_SHORT_ADDRESS, CHANNEL_ACCESS_FAILURE, NO_ACK, NO_DATA, UNAVAILABLE_KEY, FAILED_SECURITY_CHECK or INVALID_PARAMETER.	The status of the GTS request.

Figura 42- Parâmetros do MLME-GTS.confirm.

MLME-GTS.indication:

-essa primitiva é gerada pelo MLME do coordenador de PAN e enviada à sua camada de rede quando um GTS é alocado ou desalocado.

-os parâmetros são:

Name	Type	Valid range	Description
DevAddress	Device address	0 x 0000–0 x fffd	The short address of the device that has been allocated or deallocated a GTS.
GTSCharacteristics	GTS characteristics	See 7.3.3.1.2	The characteristics of the GTS.
SecurityUse	Boolean	TRUE or FALSE	An indication of whether the received frame is using security. This value is set to TRUE if the security enable subfield was set to 1 or FALSE if the security enabled subfield was set to 0.
ACLEntry	Integer	0 x 00–0 x 08	The *macSecurityMode* parameter value from the ACL entry associated with the sender of the data frame. This value is set to 0x08 if the sender of the data frame was not found in the ACL.

Figura 43- Parâmetros do MLME-GTS.indication.

A figura abaixo ilustra a seqüência de primitivas quando um ponto comum de um PAN pede a alocação de um GTS:

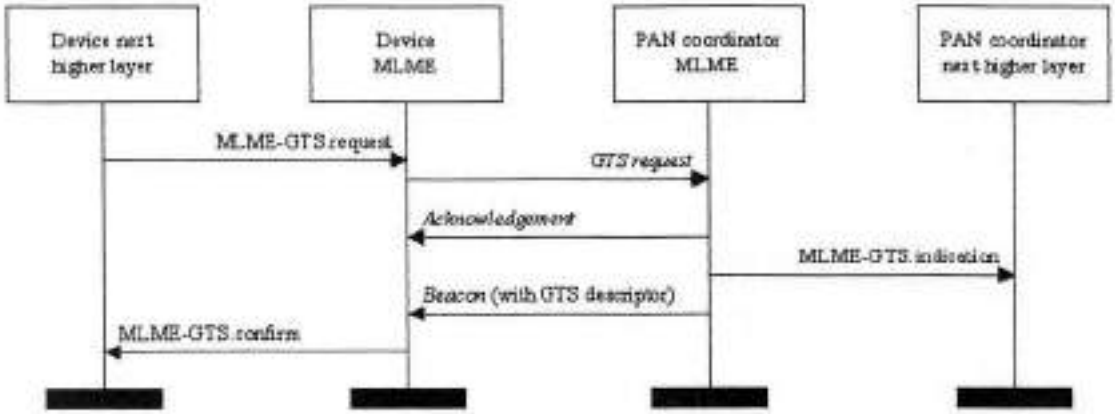


Figura 44- Seqüência de primitivas para alocação de GTS.

Abaixo, a figura ilustra a situação da desalocação de GTS, em a) pedido pelo ponto comum, em b) pedido pelo coordenador do PAN.

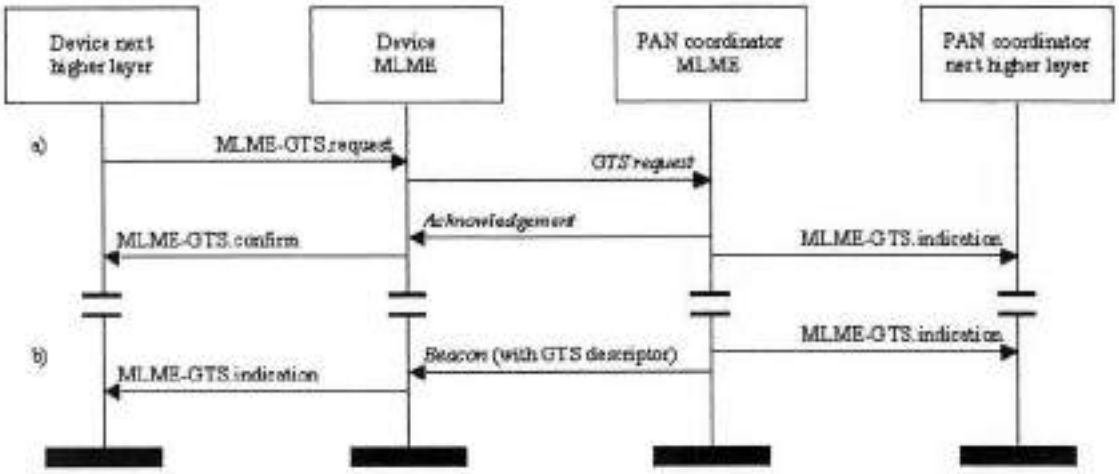


Figura 45- Seqüência de primitivas para desalocação GTS.

Primitivas para "recomeçar" a camada MAC:

Essas primitivas definem como retornar a camada MAC a seus valores default.

MLME-RESET.request:

-essa primitiva é gerada na camada de rede e enviada a camada MAC para restabelecer valores default às variáveis da camada MAC.

-os parâmetros são:

Name	Type	Valid range	Description
SetDefaultPIB	Boolean	TRUE or FALSE	If TRUE, the MAC sublayer is reset and all MAC PIB attributes are set to their default values. If FALSE, the MAC sublayer is reset but all MAC PIB attributes retain their values prior to the generation of the MLME-RESET.request primitive.

Figura 46- Parâmetros do MLME-RESET.request.

MLME-RESET.confirm:

-essa primitiva é gerada no MLME para responder á camada de rede os resultados da tentativa de MLME-RESET.request.

-os parâmetros são:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS or DISABLE_TRX_FAILURE	The result of the reset operation.

Figura 47- Parâmetros do MLME-RESET.confirm.

Primitivas para ativar o receptor por um tempo:

Essas primitivas definem como acionar o receptor por um tempo fixo.

MLME-RX-ENABLE.request:

-essa primitiva é gerada pela camada de rede como o objetivo de solicitar o acionamento do receptor por um período de tempo fixo. Esse período será acionado no início de um superframe em uma PAN funcionando no sistema de beacon ou imediatamente em uma PAN sem o sistema de beacon.

-os parâmetros são:

Name	Type	Valid range	Description
DeferPermit	Boolean	TRUE or FALSE	TRUE if the receiver enable can be deferred until during the next superframe if the requested time has already passed. FALSE if the receiver enable is only to be attempted in the current superframe. This parameter is ignored for nonbeacon-enabled PANs.
RxOnTime	Integer	0 x 000000–0 x ffffff	The number of symbols from the start of the superframe before the receiver is to be enabled. The precision of this value is a minimum of 20 bits, with the lowest 4 bits being the least significant. This parameter is ignored for nonbeacon-enabled PANs.
RxOnDuration	Integer	0 x 000000–0 x ffffff	The number of symbols for which the receiver is to be enabled.

Figura 48- Parâmetros do MLME-RX-ENABLE.request.

MLME-RX-ENABLE.confirm:

-essa primitiva é gerada pelo MLME como resposta ao pedido da camada de rede através do MLME-RX-ENABLE.request.

-os parâmetros são:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS, TX_ACTIVE, OUT_OF_CAP, or INVALID_PARAMETER	The result of the receiver enable request.

Figura 49- Parâmetros do MLME-RX-ENABLE.confirm.

A figura abaixo ilustra a troca de primitivas necessárias para a ativação do receptor:

Em a) em uma PAN com sistema de beacon funcionando.

Em b) em uma PAN com o sistema de beacon desligado.

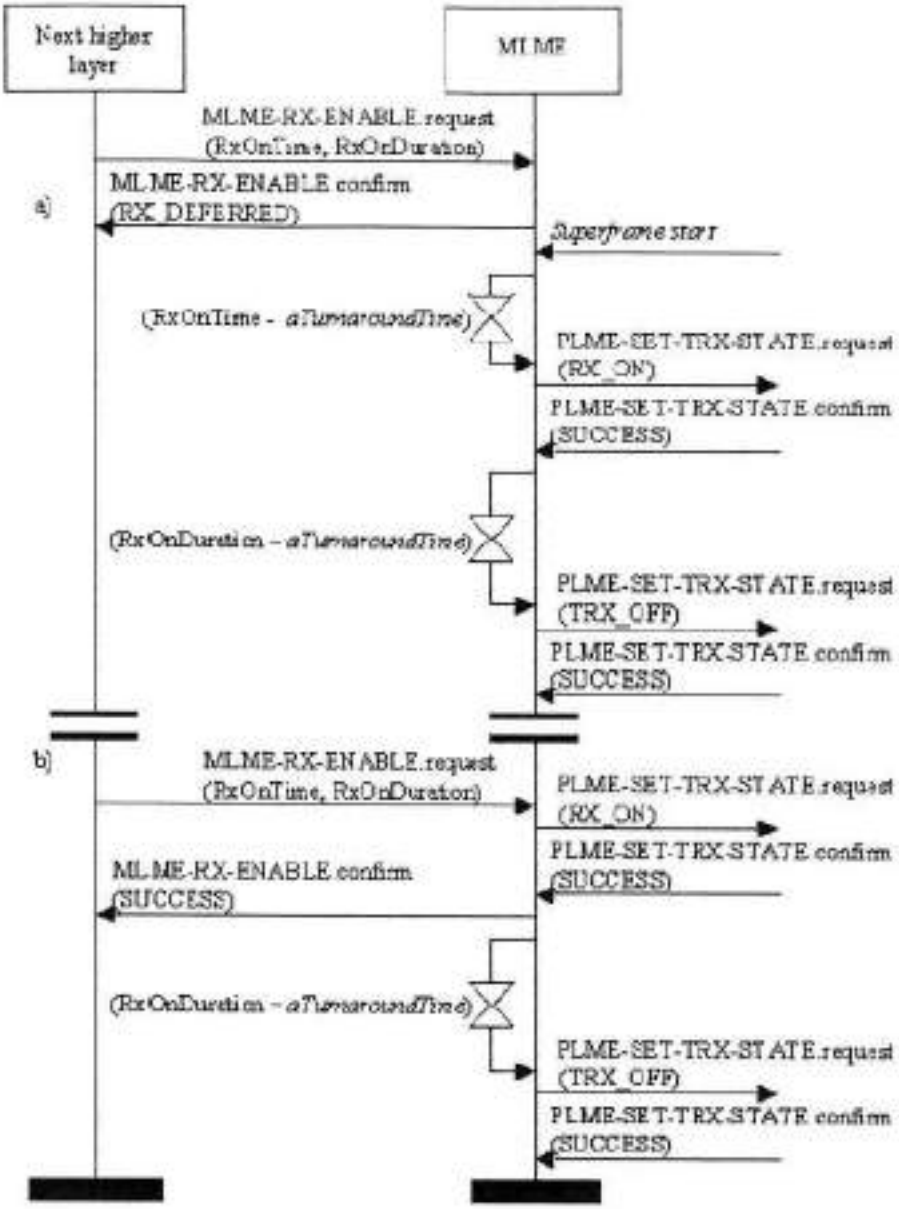


Figura 50- Seqüência de primitivas para ativação de um receptor.

Primitivas para o rastreamento de um canal:

Essas primitivas definem como um ponto pode determinar a energia de um canal ou a presença ou ausência de PAN relacionada a este.

MLME-SCAN.request:

-essa primitiva é utilizada para iniciar o processo de varredura de uma lista de canais. Essa varredura pode determinar o nivel de energia dos canais, descobrir o coordenador de PAN que este canal pode estar associado ou ainda procurar por todos os coordenadores de PAN que estão emitindo beacon aos quais este ponto está exposto em sua área de atuação.

-os parâmetros são:

Name	Type	Valid range	Description
ScanType	Integer	0 x 00—0 x 03	Indicates the type of scan performed: 0 x 00 = ED scan (FFD only). 0 x 01 = active scan (FFD only). 0 x 02 = passive scan. 0 x 03 = orphan scan.
ScanChannels	Bitmap	32 bit field	The 5 MSBs (b_{27}, ... , b_{31}) are reserved. The 27 LSBs (b_0, b_1, ... b_{26}) indicate which channels are to be scanned (1 = scan, 0 = do not scan) for each of the 27 valid channels (see 6.1.2).
ScanDuration	Integer	0—14	A value used to calculate the length of time to spend scanning each channel for ED, active, and passive scans. This parameter is ignored for orphan scans. The time spent scanning each channel is [*aBaseSuperframeDuration* * (2^n + 1)] symbols, where *n* is the value of the ScanDuration parameter.

Figura 51- Parâmetros do MLME-SCAN.request.

MLME-SCAN.confirm:

-essa primitiva retorna a camada de rede os resultados da tentativa de varredura sobre os canais do ponto.

-os parâmetros são:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS, NO_BEACON, or INVALID_PARAMETER	The status of the scan request.
ScanType	Integer	0 x 00—0 x 03	Indicates if the type of scan performed: 0 x 00 = ED scan (FFD only). 0 x 01 = active scan (FFD only). 0 x 02 = passive scan. 0 x 03 = orphan scan.
UnscannedChannels	Bitmap	32 bit field	Indicates which channels given in the request were not scanned (1 = not scanned, 0 = scanned or not requested). This parameter is only valid for passive or active scans.
ResultListSize	Integer	Implementation specific	The number of elements returned in the appropriate result lists. This value is 0 for the result of an orphan scan.
EnergyDetectList	List of integers	0 x 00–0 x ff for each integer	The list of energy measurements, one for each channel searched during an ED scan. This parameter is null for active, passive, and orphan scans.
PANDescriptorList	List of PAN descriptor values	See Table 41	The list of PAN descriptors, one for each beacon found during an active or passive scan. This parameter is null for ED and orphan scans.

Figura 52- Parâmetros do MLME-SCAN.confirm.

Na seção de seqüência de primitivas está uma figura que ilustra a seqüência de primitivas para o processo de varredura do canal.

Primitiva de informe do estado da comunicação:

MLME-COMM-STATUS.indication:

-essa primitiva é gerada pelo MLME e serve para informar a camada de rede o estado da comunicação estabelecida pelo ponto de rede. Essa primitiva é gerada espontaneamente após alguns procedimentos originados pela camada de rede.

-os parâmetros são:

Name	Type	Valid range	Description
PANId	Integer	0 x 0000—0 ffff	The 16 bit PAN identifier of the device from which the frame was received or to which the frame was being sent.
SrcAddrMode	Integer	0 x 00—0 x 03	The source addressing mode for this primitive. This value can take one of the following values: 0 = no address (addressing fields omitted). 0 x 01 = reserved. 0 x 02 = 16 bit short address. 0 x 03 = 64 bit extended address.
SrcAddr	Device address	As specified by the SrcAddrMode parameter	The individual device address of the entity from which the frame causing the error originated.
DstAddrMode	Integer	0 x 00—0 x 03	The destination addressing mode for this primitive. This value can take one of the following values: 0 x 00 = no address (addressing fields omitted). 0 x 01 = reserved. 0 x 02 = 16 bit short address. 0 x 03 = 64 bit extended address.
DstAddr	Device address	As specified by the DstAddrMode parameter	The individual device address of the device for which the frame was intended.
Status	Enumeration	SUCCESS, TRANSACTION_OVERFLOW, TRANSACTION_EXPIRED, CHANNEL_ACCESS_FAILURE, NO_ACK, UNAVAILABLE_KEY, FRAME_TOO_LONG, FAILED_SECURITY_CHECK or INVALID_PARAMETER	The communications status.

Figura 53- Parâmetros do MLME-COMM-STATUS.indication.

Primitivas para escrever os atributos MAC PIB:

MLME-SET.request:

-essa primitiva é gerada pela camada de rede e enviada para a camada MAC com o objetivo de escrever nesta algum atributo especificado dos MAC PIB.

-os parâmetros são:

Name	Type	Valid range	Description
PIBAttribute	Integer	See Table 71 and Table 72	The identifier of the MAC PIB attribute to write.
PIBAttributeValue	Various	Attribute specific; see Table 71 and Table 72	The value to write to the indicated MAC PIB attribute.

Figura 54- Parâmetros do MLME-SET.request.

MLME-SET.confirm:

-essa primitiva é uma resposta dada pela camada MAC à sua camada de rede quando do recebimento do MLME-SET.request.

-os parâmetros são:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS, UNSUPPORTED_ATTRIBUTE, or INVALID_PARAMETER	The result of the request to write the MAC PIB attribute.
PIBAttribute	Integer	See Table 71 and Table 72	The identifier of the MAC PIB attribute that was written.

Figura 55- Parâmetros do MLME-SET.confirm.

Primitivas para atualizar a configuração de superframe:

MLME-START.request:

-essa primitiva é gerada pela camada de rede, solicitando à MLME que o ponto passe a utilizar uma diferente configuração de superframe.

-os parâmetros são:

Name	Type	Valid range	Description
PANId	Integer	0 x 0000—0 x ffff	The PAN identifier to be used by the beacon.
LogicalChannel	Integer	Selected from the available logical channels supported by the PHY	The logical channel on which to start transmitting beacons.
BeaconOrder	Integer	0—15	How often the beacon is to be transmitted. The beacon order, *BO*, and the beacon interval, *BI*, are related as follows: for $0 \leq BO \leq 14$, $BI = aBaseSuperframeDuration * 2^{BO}$ symbols. If $BO = 15$, the coordinator will not transmit a beacon, and the SuperframeOrder parameter value is ignored.
SuperframeOrder	Integer	0—*BO* or 15	The length of the active portion of the superframe, including the beacon frame. The superframe order, *SO*, and the superframe duration, *SD*, are related as follows: for $0 \leq SO \leq BO \leq 14$, $SD = aBaseSuperframeDuration * 2^{SO}$ symbols. If $SO = 15$, the superframe will not be active after the beacon.
PANCoordinator	Boolean	TRUE or FALSE	If this value is TRUE, the device will become the PAN coordinator of a new PAN. If this value is FALSE, the device will begin transmitting beacons on the PAN with which it is associated.
BatteryLifeExtension	Boolean	TRUE or FALSE	If this value is TRUE, the receiver of the beaconing device is disabled *macBattLifeExtPeriods* full backoff periods after the interframe spacing (IFS) period of the beacon frame. If this value is FALSE, the receiver of the beaconing device remains enabled for the entire CAP.
CoordRealignment	Boolean	TRUE or FALSE	TRUE if a coordinator realignment command is to be transmitted prior to changing the superframe configuration or FALSE otherwise.
SecurityEnable	Boolean	TRUE or FALSE	TRUE if security is enabled for beacon transmissions or FALSE otherwise.

Figura 56- Parâmetros do MLME-START.request.

MLME-START.confirm:

-essa primitiva é gerada pelo MLME em resposta à MLME-START.request.

-os parâmetros são:

Name	Type	Valid range	Description
status	Enumeration	SUCCESS, NO_SHORT_ADDRESS, UNAVAILABLE_KEY, FRAME_TOO_LONG, FAILED_SECURITY_CHECK, or INVALID_PARAMETER	The result of the attempt to start using an updated superframe configuration.

Figura 57- Parâmetros do MLME-START.confirm.

A figura abaixo ilustra a seqüência de primitivas para atualizar a configuração de superframe do ponto.

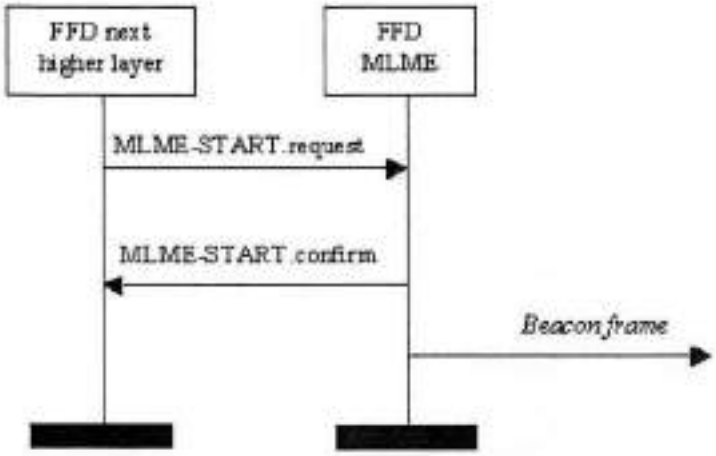


Figura 58- Troca de primitivas para atualizar configuração de superframe.

Primitivas para a sincronização com o coordenador de PAN:

Essas primitivas são utilizadas para estabelecer uma sincronização com o coordenador de PAN ou comunicar à camada de rede uma possível perda de sincronia.

MLME-SYNC.request:

-essa primitiva é gerada pela camada de rede de um ponto pertencente a um PAN que tem o sistema de beacon ativado e tem como objetivo pedir a sincronia com o coordenador desse PAN.

-os parâmetros são:

Name	Type	Valid range	Description
LogicalChannel	Integer	Selected from the available logical channels supported by the PHY	The logical channel on which to attempt coordinator synchronization.
TrackBeacon	Boolean	TRUE or FALSE	TRUE if the MLME is to synchronize with the next beacon and attempt to track all future beacons. FALSE if the MLME is to synchronize with only the next beacon.

Figura 59- Parâmetros do MLME-SYNC.request.

MLME-SYNC-LOSS.indication:

-essa primitiva é gerada pelo MLME de um ponto que pertencente a uma PAN perdeu sincronia com o seu coordenador de PAN, então é enviada à sua camada de rede.Essa primitiva pode ser gerada também pelo MLME de um coordenador de PAN na situação de conflito de PAN ID.

-os parâmetros são:

Name	Type	Valid range	Description
LossReason	Enumeration	PAN_ID_CONFLICT, REALIGNMENT, or BEACON_LOST	The reason that synchronization was lost.

Figura 60- Parâmetros do MLME-SYNC-LOSS.indication.

A figura abaixo ilustra a seqüência de primitivas no processo de sincronização com o coordenador de PAN em a) sem pedido de rastreamento de beacon e em b) com pedido de rastreamento.

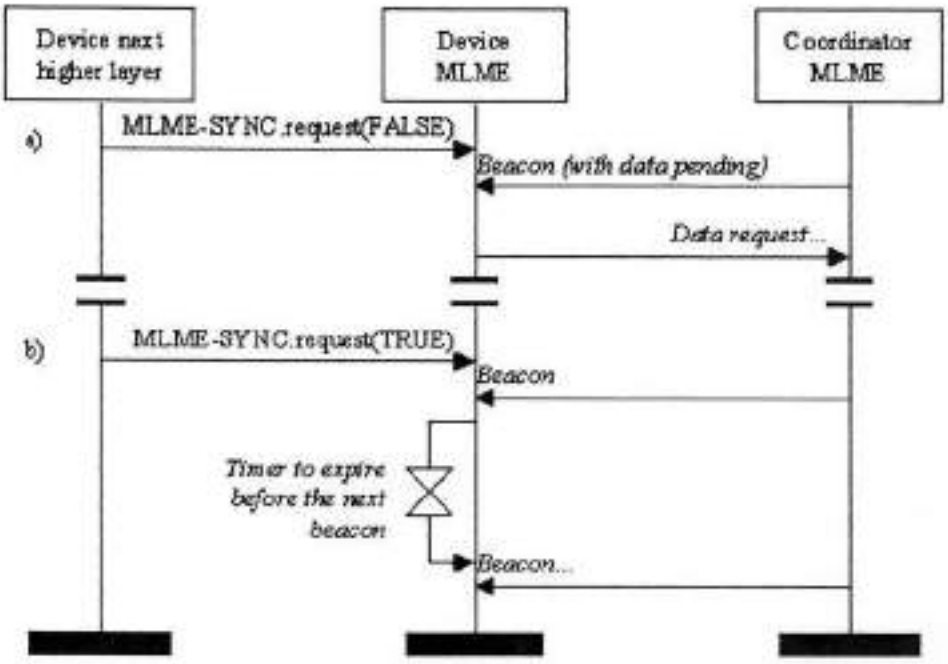


Figura 61- Troca de primitivas para sincronização com o coordenador.

Primitivas para pedir dados em espera ao coordenador de PAN:

MLME-POLL.request:

-essa primitiva é gerada pela camada de rede de um ponto comum e enviada ao seu MLME com o objetivo de gerar uma verificação no seu coordenador de PAN se existem frames de dados endereçados àquele ponto pendentes.

-os parâmetros são:

Name	Type	Valid range	Description
CoordAddrMode	Integer	0 x 02—0 x 03	The addressing mode of the coordinator to which the poll is intended. This parameter can take one of the following values: 2 = 16 bit short address. 3 = 64 bit extended address.
CoordPANId	Integer	0 x 0000—0 x fffe	The PAN identifier of the coordinator to which the poll is intended.
CoordAddress	Device-Address	As specified by the CoordAddrMode parameter	The address of the coordinator to which the poll is intended.
SecurityEnable	Boolean	TRUE or FALSE	TRUE if security is enabled for this transfer or FALSE otherwise.

Figura 62- Parâmetros do MLME-POLL.request.

MLME-POLL.confirm:

-essa primitiva é gerada pelo MLME do ponto que a camada de rede gerou uma verificação(MLME-POLL.request) e enviada para esta.

-os parâmetros são:

Name	Type	Valid range	Description
status	Integer	SUCCESS, CHANNEL_ACCESS_FAILURE, NO_ACK, NO_DATA, UNAVAILABLE_KEY, FAILED_SECURITY_CHECK, or INVALID_PARAMETER	The status of the data request.

Figura 63- Parâmetros do MLME-POLL.confirm.

A figura abaixo ilustra a seqüência de primitivas trocadas para o pedido de dados de um ponto ao seu coordenador de PAN. No caso a) não existem dados pendentes, no caso b) sim.

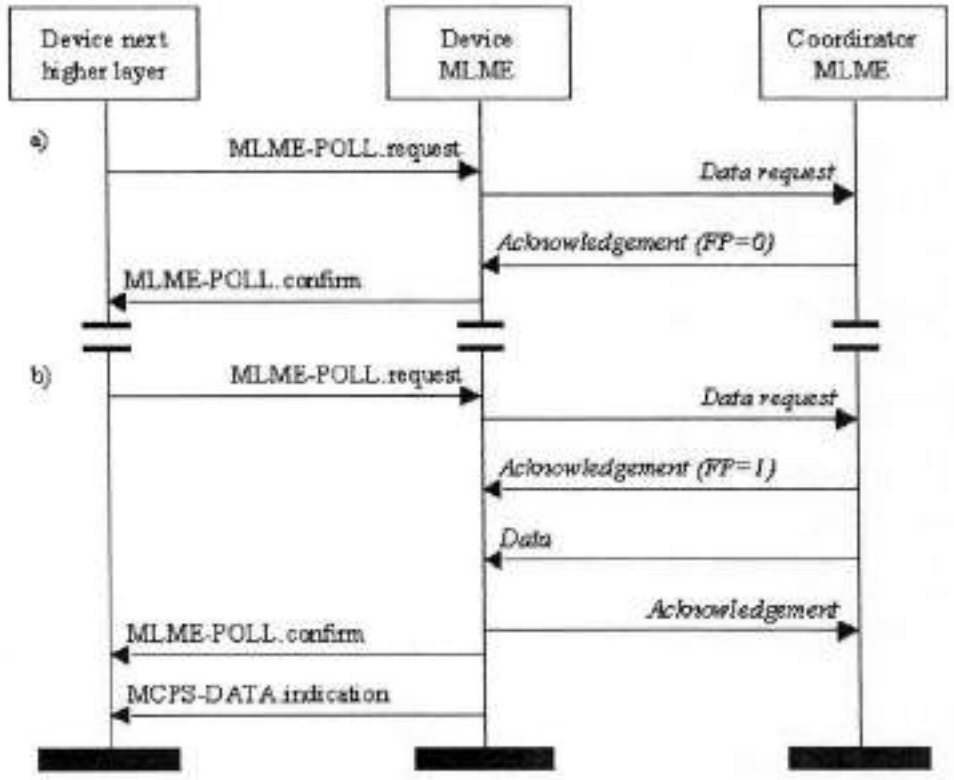


Figura 64- Seqüência de primitivas para o pedido de dados ao coordenador.

A tabela abaixo mostra o significado das possíveis respostas que possam vir nos parâmetros das primitivas da camada MAC:

Enumeration	Value	Description
SUCCESS	0 x 00	The requested operation was completed successfully. For a transmission request, this value indicates a successful transmission.
—	0 x 01—0 x df	Reserved for MAC command status and reason code values.
—	0 x 80—0 x df	Reserved.
BEACON_LOSS	0 x e0	The beacon was lost following a synchronization request.
CHANNEL_ACCESS_FAILURE	0 x e1	A transmission could not take place due to activity on the channel, i.e., the CSMA-CA mechanism has failed.
DENIED	0 x e2	The GTS request has been denied by the PAN coordinator.
DISABLE_TRX_FAILURE	0 x e3	The attempt to disable the transceiver has failed.
FAILED_SECURITY_CHECK	0 x e4	The received frame induces a failed security check according to the security suite.
FRAME_TOO_LONG	0 x e5	The frame resulting from secure processing has a length that is greater than *aMACMaxFrameSize*.
INVALID_GTS	0 x e6	The requested GTS transmission failed because the specified GTS either did not have a transmit GTS direction or was not defined.
INVALID_HANDLE	0 x e7	A request to purge an MSDU from the transaction queue was made using an MSDU handle that was not found in the transaction table.
INVALID_PARAMETER	0 x e8	A parameter in the primitive is out of the valid range.
NO_ACK	0 x e9	No acknowledgment was received after *aMaxFrameRetries*.
NO_BEACON	0 x ea	A scan operation failed to find any network beacons.
NO_DATA	0 x eb	No response data were available following a request.
NO_SHORT_ADDRESS	0 x ec	The operation failed because a short address was not allocated.
OUT_OF_CAP	0 x ed	A receiver enable request was unsuccessful because it could not be completed within the CAP.
PAN_ID_CONFLICT	0 x ee	A PAN identifier conflict has been detected and communicated to the PAN coordinator.
REALIGNMENT	0 x ef	A coordinator realignment command has been received.
TRANSACTION_EXPIRED	0 x f0	The transaction has expired and its information discarded.
TRANSACTION_OVERFLOW	0 x f1	There is no capacity to store the transaction.
TX_ACTIVE	0 x f2	The transceiver was in the transmitter enabled state when the receiver was requested to be enabled.
UNAVAILABLE_KEY	0 x f3	The appropriate key is not available in the ACL.
UNSUPPORTED_ATTRIBUTE	0 x f4	A SET/GET request was issued with the identifier of a PIB attribute that is not supported.
—	0 x f5–0 x ff	Reserved.

Figura 65-Parâmetros de respostas MAC.

Formato dos frames da camada MAC:

O pacote da camada MAC é o MPDU(MAC protocol data unit) e é composto por:

- MHR(MAC header), que contém o frame de controle, o número de seqüência e informação de endereço.
- MAC payload, que contém informações específicas para cada tipo de pacote.
- MFR(MAC footer), que contém o FCS(frame check sequence).

A figura abaixo ilustra o formato geral do pacote da camada MAC:

<table>
<tr><td>Octets: 2</td><td>1</td><td>0/2</td><td>0/2/8</td><td>0/2</td><td>0/2/8</td><td>variable</td><td>2</td></tr>
<tr><td rowspan="2">Frame control</td><td rowspan="2">Sequence number</td><td>Destination PAN identifier</td><td>Destination address</td><td>Source PAN identifier</td><td>Source address</td><td rowspan="2">Frame payload</td><td rowspan="2">FCS</td></tr>
<tr><td colspan="4">Addressing fields</td></tr>
<tr><td colspan="6">MHR</td><td>MAC payload</td><td>MFR</td></tr>
</table>

Figura 66-Formato do MPDU.

1. O formato do frame de controle é ilustrado abaixo:

Bits: 0–2	3	4	5	6	7–9	10–11	12–13	14–15
Frame type	Security enabled	Frame pending	Ack. request	Intra-PAN	Reserved	Dest. addressing mode	Reserved	Source addressing mode

Figura 67-Formato do frame de controle.

Conforme a figura ele é composto por:

- Campo do tipo de pacote, que pode assumir os seguintes valores para os respectivos tipo de pacotes:

Frame type value $b_2\, b_1\, b_0$	Description
000	Beacon
001	Data
010	Acknowledgment
011	MAC command
100–111	Reserved

Figura 68- Valores do campo do tipo de pacote

- Campo habilitador de segurança, que assume 0 para pacote não protegido por criptografia e 1 para pacote protegido.
- Campo de pendência de pacotes, que assume 0 se não existem mais pacotes pendentes que fazem parte da informação objeto desse pacote e 1 se existirem.
- Campo de "aknowledgment", que assume 1 se o receptor desse pacote deve mandar um pacote pela confirmação de recebimento deste e 0 se não.
- Campo "intra-PAN", que assume 1 se o receptor do pacote pertencer ao mesmo PAN que o emissor e 0 se pertencerem a PANs diferentes.
- Campo de modo de endereçamento de destino, que indica as opções da tabela abaixo com relação ao endereço de destino.

Addressing mode value $b_1\, b_0$	Description
00	PAN identifier and address field are not present.
01	Reserved.
10	Address field contains a 16 bit short address.
11	Address field contains a 64 bit extended address.

Figura 69-Modo de endereçamento do destino.

- Campo de modo de endereçamento de fonte, que indica as opções da tabela acima para o endereço do emissor.

2. O campo de número de seqüência é um número único que indica a seqüência do pacote.
3. O campo de identificação do PAN de destino indica como propõe o nome a identificação do PAN onde o pacote deve chegar.
4. O campo de endereço de destino indica o endereço do ponto onde o pacote deve chegar sendo de 16 ou 64 bits.
5. O campo de identificação do PAN do emissor é explicado pelo próprio nome.
6. O campo de endereço do emissor do pacote também pode ter 16 ou 64 bits.
7. O campo "frame payload" traz informações específicas para cada tipo de pacote.
8. O campo de FCS contém bits calculados através de um algoritmo para garantir integridade do pacote a partir dos bits do MHR e do MAC payload.

Agora serão ilustrados algum pacotes especiais que são MPDUs.

a. O pacote de beacon tem o seguinte formato:

Octets: 2	1	4/10	2	variable	variable	variable	2
Frame control	Sequence number	Addressing fields	Superframe specification	GTS fields (Figure 38)	Pending address fields (Figure 39)	Beacon payload	FCS
MHR			MAC payload				MFR

Figura 70- Formato do beacon.

Sendo os campos de GTS formatados da seguinte forma:

Octets: 1	0/1	variable
GTS specification	GTS directions	GTS list

Figura 71- Formato de campos GTS.

Os campos de endereços pendentes assumem a seguinte forma:

Octets: 1	variable
Pending address specification	Address list

Figura 72- Formato dos campos de endereço pendentes.

No MHR o campo de tipo de pacote deve ter como conteúdo o código que indica que esse pacote é um pacote de beacon.

O pacote de especificação de superframe deve tem a seguinte composição:

Bits: 0-3	4-7	8-11	12	13	14	15
Beacon order	Superframe order	Final CAP slot	Battery life extension	Reserved	PAN coordinator	Association permit

Figura 73- Composição do superframe.

b. O pacote de dados tem o seguinte formato:

Octets: 2	1	(see 7.2.2.2.1)	variable	2
Frame control	Sequence number	Addressing fields	Data payload	FCS
MHR			MAC payload	MFR

Figura 74- Formato do pacote de dados.

No MHR o campo de tipo de pacote deve conter o código que indica que este é um pacote de dados.

O Data payload são os dados propriamente ditos que devem chegar à camada de rede do ponto destino do pacote.

c. O pacote de "Acknowledgment" tem o seguinte formato:

Octets: 2	1	2
Frame control	Sequence number	FCS
MHR		MFR

Figura 75- Formato do pacote de "Acknowledgment".

No MHR o campo de tipo de pacote deve conter o código que indica que este é um pacote de "Acknowledgement".
Este pacote não contém payload.

d. O pacote de comandos da camada MAC tem o seguinte formato:

Octets: 2	1	(see 7.2.2.4.1)	1	variable	2
Frame control	Sequence number	Addressing fields	Command frame identifier	Command payload	FCS
MHR			MAC payload		MFR

Figura 76- Formato de comando da camada MAC.

No MHR o campo de tipo de pacote deve conter o código que indica que este é um pacote de comando da camada MAC.
O campo de identificação de comando da camada MAC deve assumir um dos seguintes valores segundo a sua função:

Command frame identifier	Command name	RFD	
		Tx	Rx
0 x 01	Association request	X	
0 x 02	Association response		X
0 x 03	Disassociation notification	X	X
0 x 04	Data request	X	
0 x 05	PAN ID conflict notification	X	
0 x 06	Orphan notification	X	
0 x 07	Beacon request		
0 x 08	Coordinator realignment		X
0 x 09	GTS request		
0 x 0a—0 x ff	Reserved		

Figura 77- Valores de identificação de comando da camada MAC.

O campo MAC payload guarda o comando em si.

Funcionalidades da camada MAC no acesso ao canal:

O protocolo 802.15.4 apresenta 2 possibilidades de acesso ao canal de transmissão implementadas na camada MAC e que tem profunda relevância quanto ao funcionamento das redes construídas sobre esses conceitos, tempo de funcionamento dos componentes da rede(como conseqüência influenciando no consumo de energia dos componentes e durabilidade das baterias disponíveis) e aplicabilidade desse padrão a diferentes situações. Essas características apresentam-se como diferenciais e nelas é que residem o principal apelo de aplicação desse protocolo e conceitos nele embutidos.

O acesso do canal pode ser executado utilizando ou não a estrutura de superframe.

1. Estrutura de superframe:

O acesso do canal utilizando a estrutura de superframe implica em uma transmissão de mensagens por ciclos(superframes) que tem como delimitadores os beacon. Em seguida a cada beacon o ciclo apresenta uma parte ativa de transmissão onde o coordenador de cada PAN deve atuar e os outros pontos componentes da PAN podem ou não atuar conforme a sua necessidade e uma parte inativa onde estabelece-se na rede dos componentes do PAN uma latência, onde esses dispositivos podem operar em um estado de baixo consumo de energia.

A estrutura do superframe é descrita por 2 valores, macBeaconOrder e macSuperframeOrder. O macBeaconOrder(BO) determina o intervalo ao qual o coordenador de Pan deve transmitir os seus beacon. O valor do macBeaconOrder deve ser de 0 a 14(se BO for 15, não será usado o superframe) enquanto que o valor do intervalo entre beacon(BI) deve ser :

BI = aBaseSuperframeDuration * (2^BO) símbolos

O macSuperframeOrder(SO) descreve a duração da parte ativa do superframe, incluindo o beacon. O valor SO deve ser menor que o BO e que 14, enquanto a duração do superframe(SD) e dada por:

SD = aBaseSuperFrameDuration * (2^SO) símbolos

Se o BO for menor que 15 e o SO for igual a 15 não terá parte ativa após o beacon no superframe.

A parte ativa do de cada superframe é dividida em um número(aNumSuperframeSlots) de slots de igual duração no valor de:

(2^so) * aBaseSlotDuration

Esses slots estão divididos em 3 grupos, o beacon, CAP(contention access period) e o CFP(contention free period) . O beacon deve ser transmitido sem o uso de CSMA(carrier sense multiple access) e começar no slot 0. O CAP deve vir logo em seguida ao beacon e o CFP se existir entre o CAP e o fim da parte ativa do superframe. Os GTS devem ser alocados dentre os slots pertencentes ao CFP.

Ilustração da estrutura de superframe:

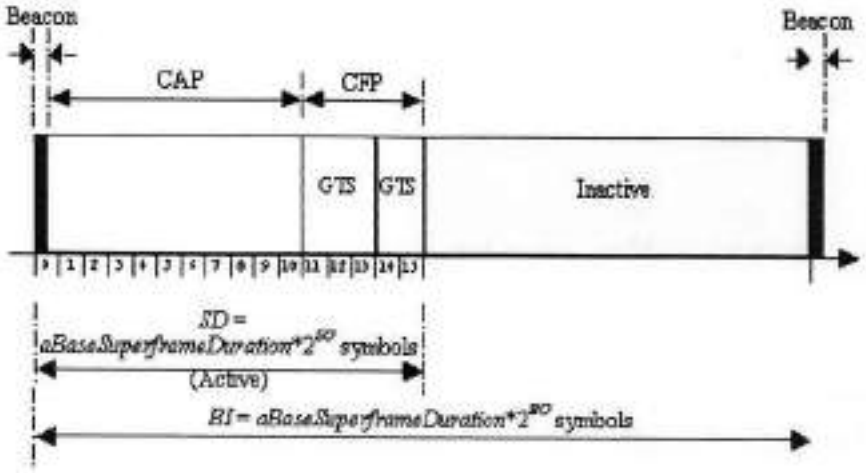


Figura 78- Estrutura de superframe.

Os pacotes de comando da camada MAC devem ser transmitidos no CAP. Todos os pacotes transmitidos no CAP exceto os de "aknowledgment" utilizam CSMA-CA.
Os pacotes transmitidos no CFP não podem utilizar CSMA.
O sistema de GTS consiste em dedicar alguns slots dentro do CFP para comunicação de um determinado ponto do PAN e seu coordenador. Portanto, a cada ciclo(superframe) existirá um momento em que apenas o coordenador do PAN e o ponto que tem um GTS alocado para si precisarão estar ativos, e o resto dos pontos dA PAN poderão estar em latência.
Apenas o coordenador de PAN tem a possibilidade de alocar GTSs e o número que ele pode alocar está limitado a 7.

Principais processos, com seus fluxos de primitivas(camadas MAC e PHY):

Mensagens de formação de uma PAN na perspectiva do futuro coordenador:

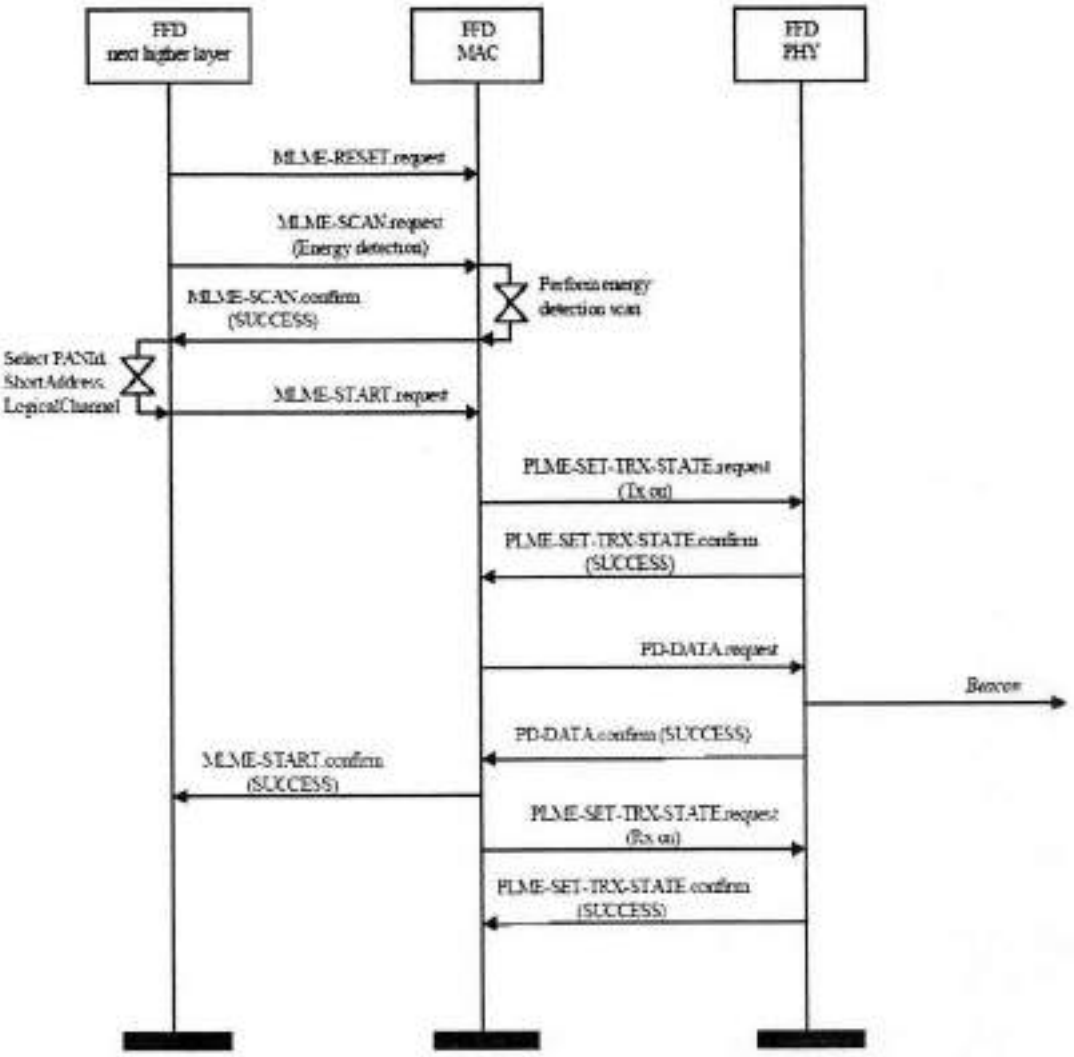


Figura 79- Formação de PAN- visão coordenador.

O procedimento de varredura de energia de um canal:

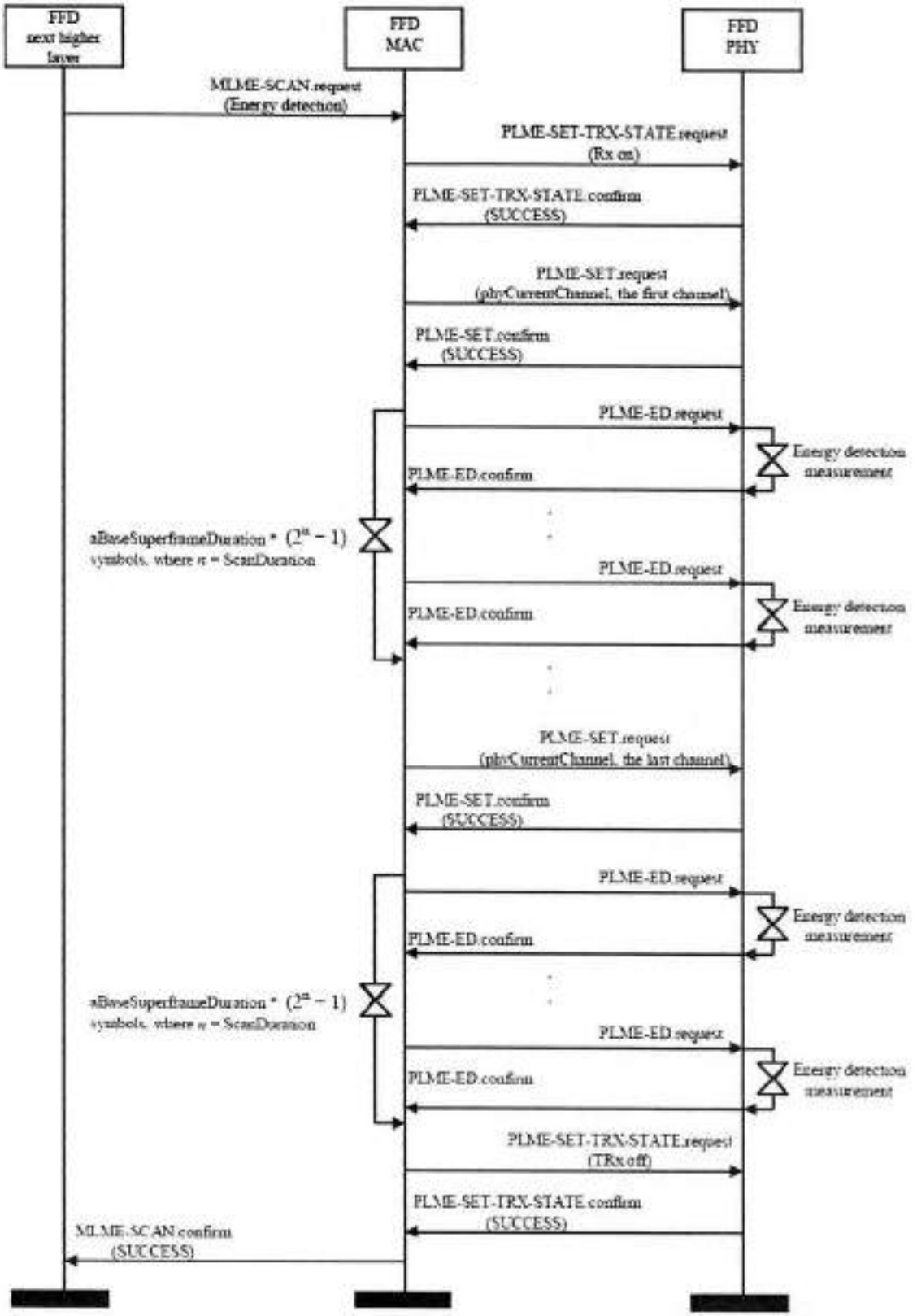


Figura 80- Varredura de energia do canal.

Seqüência de mensagens de associação a uma PAN existente na perspectiva de um nó comum não associado:

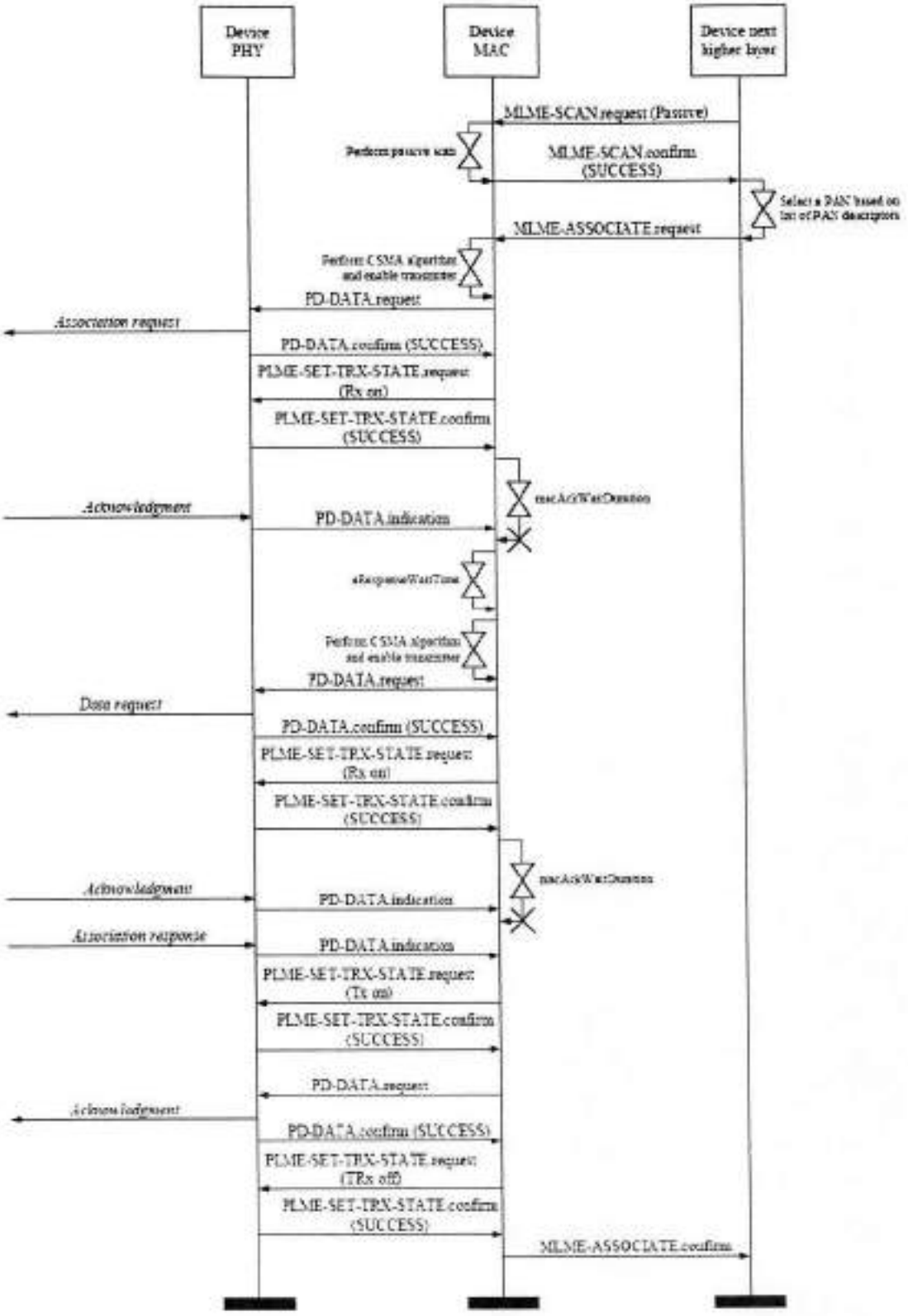


Figura 81- Associação à uma PAN-visão de um nó comum.

Mensagens para a associação de um ponto a uma PAN, na perspectiva do coordenador da PAN:

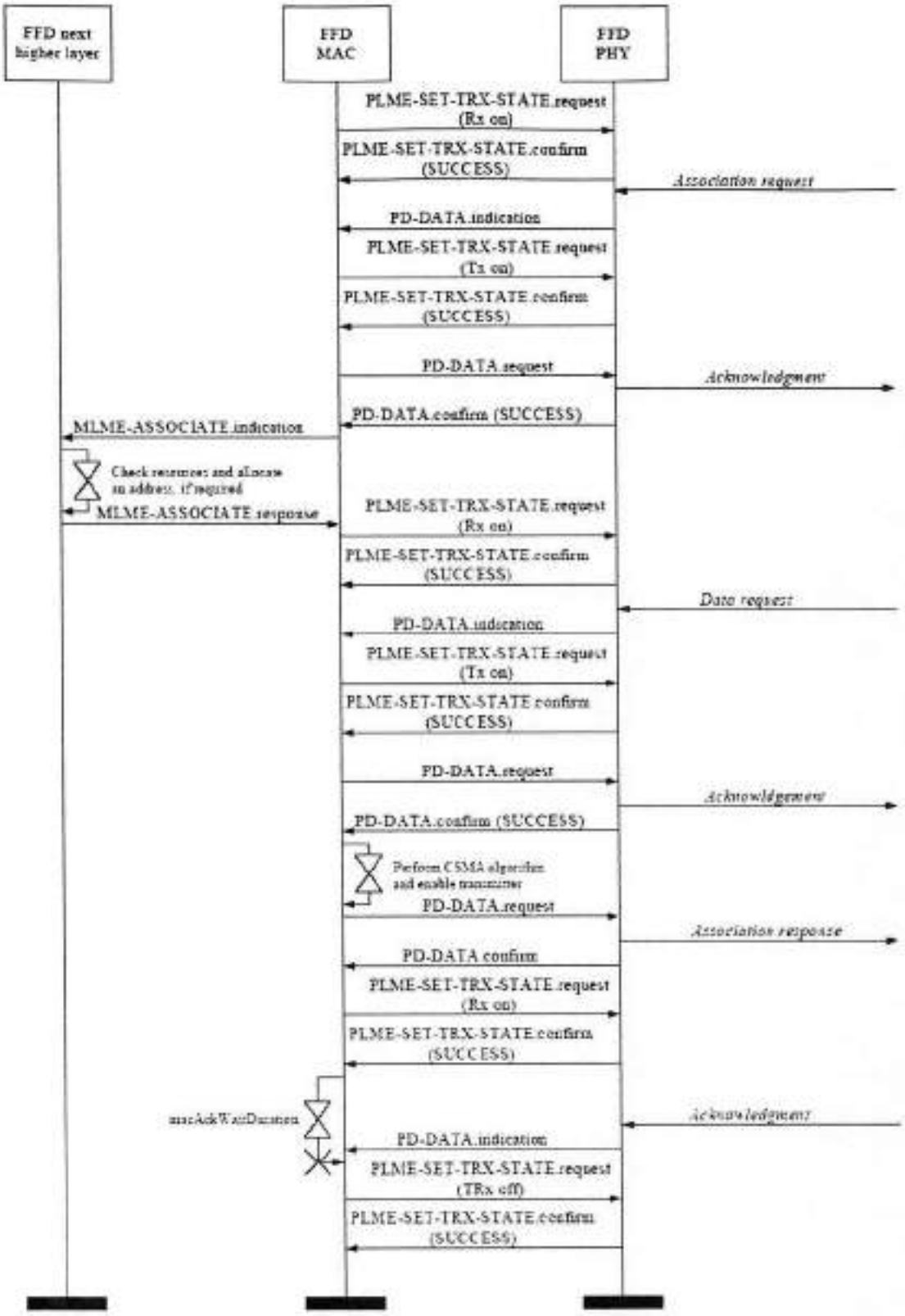


Figura 83- Associação à uma PAN-visão coordenador.

Troca de primitivas para a transmissão de dados entre pontos da rede, na perspectiva da origem da mensagem:

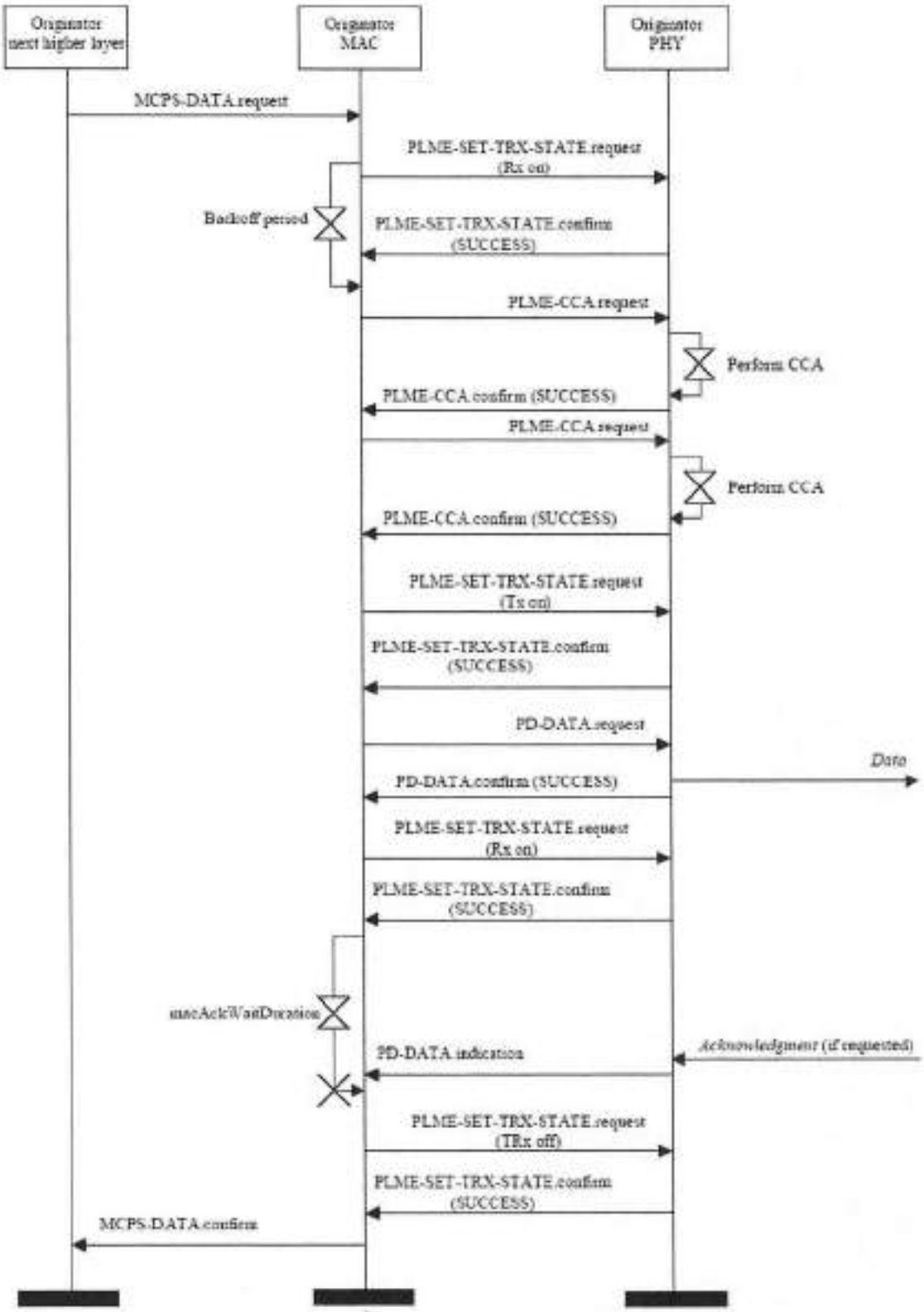


Figura 84- Transmissão de dados-origem da mensagem.

Troca de primitivas para a transmissão de dados entre pontos da rede, na perspectiva do receptor da mensagem:

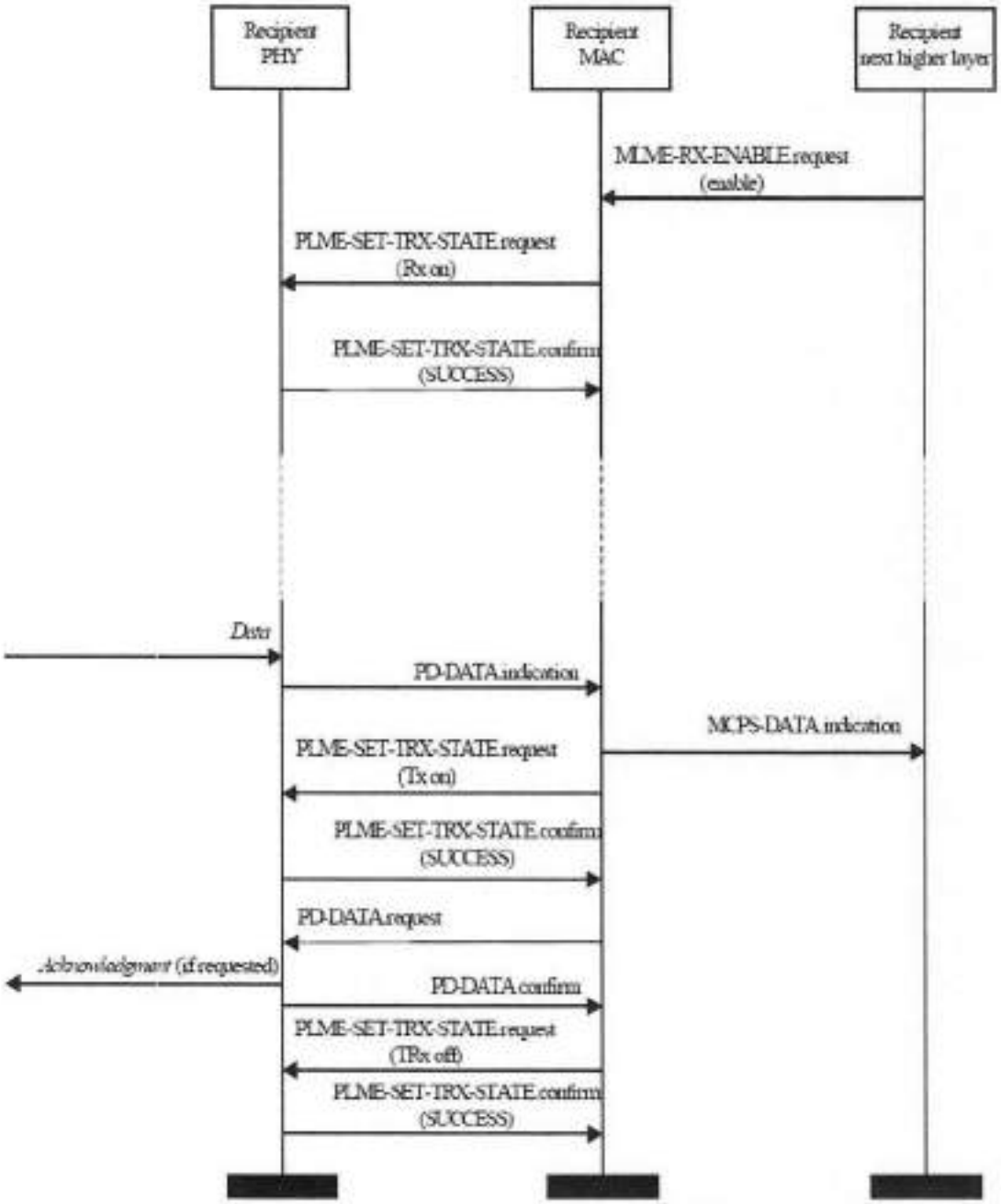


Figura 85- Transmissão de dados- receptor da mensagem.

ANÁLISE MERCADOLÓGICA DO PRODUTO

Posicionamento tecnológico e concorrencial do mercado

Características técnicas do zigbee:

O protocolo 802.15.4 foi desenhado para dar origem a redes sem fio de equipamentos de controle e monitoramento com baixo consumo de energia.

Redes sem fio de equipamentos de monitoramento necessitam que os dispositivos da rede tenham bateria de longa duração, muitas vezes a duração do tempo de vida do sensor de monitoramento. Os dispositivos do padrão 802.15.4 tem esta característica, possibilitada por uma taxa de conexão alta e, portanto, tempos de "adormecimento" mais longos, o que favorece o menor consumo de energia.

Outra característica necessária ao produto almejado é o baixo custo do dispositivo, pois as aplicações são normalmente com pontos de rede numerosos, para aplicações de baixo investimento. Característica que também é atendida pelo padrão em comparação com similares encontrados no mercado.

Essas aplicações impõem também aos dispositivos dimensões pequenas, para que os nós da rede possam ser incorporados a sensores sem alterar a sua funcionalidade. Outra característica que também foi alcançado com o padrão.

Destaca-se também outra característica adequada as necessidades das aplicações, que é o alto número de nós endereçáveis por rede.

O protocolo também contempla uma transmissão de dados com confiabilidade, necessária para algumas aplicações que são alvo do produto.

Aplicações que precisam das características do zigbee:

O padrão ZigBee (protocolo 802.15.4) foi projetado para atender as necessidades de aplicações de redes de sensores sem fio, isso implica sensores de todos os tipos para monitoramento e controle remoto de ambientes.

Diante desta perspectiva surgem aplicações para automação doméstica. Produtos como iluminação ou aquecimento de ambientes, com sensores e interruptores ligados em rede e comandados por um controle remoto, ou até mesmo, pelo celular ligado à internet.

Eletrodomésticos serão fabricados já com o chip zigbee embutido e quando instalados na casa, já farão parte da rede doméstica. Possibilitando a programação de funcionamento de máquinas de lavar, geladeiras, forno micro ondas, tudo de acordo com a conveniência do morador.

Fica explícito pelo perfil de aplicações até aqui exemplificados que características como baixo custo do dispositivo, longa duração de baterias, pequena dimensão do dispositivo são imprescindíveis para o sucesso da implementação destes produto no mercado, mostrando aderência do projeto zigbee a essas aplicações.

Este padrão também pode ser aplicado a produtos de segurança e sistemas antiincêndio de ambientes, ou mesmo de edifícios, pois o protocolo permite redes de muitos nós(255), possibilitando assim a instalação de muitos sensores e a integração destes a outras redes de informação ou mesmo de dispositivos de contenção.

Propõe-se também para esse padrão aplicações em ambientes industriais, sendo utilizados para conectar sensores de máquinas de linha de produção, controle de pessoal, segurança, etc.

Outro alvo do padrão são as aplicações em sensores ligados a área de saúde no monitoramento de pacientes e processos dentro de hospitais.

Potenciais concorrentes

O padrão Zigbee não é o primeiro padrão estabelecido para redes sem fio. Outros padrões já estão no mercado e atuando com relativo sucesso. Mas vale ressaltar que os diferentes padrões tem diferentes características o que os favorecem para determinadas aplicações e os desfavorecem para outras.

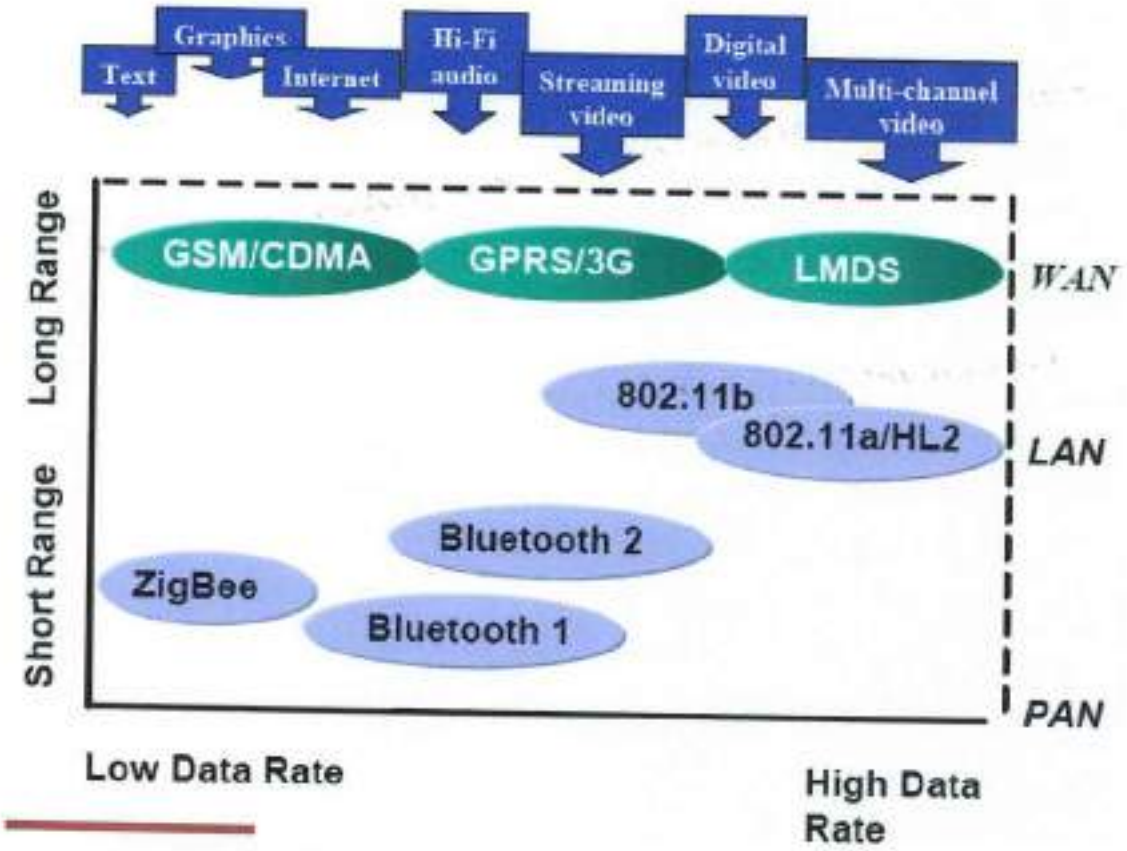


Figura 87- Posicionamento do Zig Bee.

O padrão Bluetooth é um padrão para redes sem fio que atua com alta transmissão de dados e pacotes maiores de dados, mas com alto consumo de energia e baterias recarregáveis. A rede bluetooth pode ter no máximo 8 nós. O alcance de transmissão de dados entre nós não passa dos 10 metros.

Diferenças

Conforme vimos o padrão zigbee diferencia-se dos padrões de rede sem fio existentes no mercado.

Com relação ao custo, pela sua simplicidade ele é o mais barato. O "Korean Advanced Instituteof scienceand Technology" anunciou que conseguiu produzir um protótipo do protocolo 802.15.4 com o custo inferior a 1 dólar. Em relação ao tamanho o padrão zigbee também mostra-se compatível com o mercado das suas

aplicações alvo, o mesmo instituto koreano conseguiu produzir o protótipo com dimensões de 8,75mm2.

Outra característica que diferencia o padrão zigbee do padrão bluetooth é a quantidade máxima possível de nós por rede(bluetooth 8, zigbee 255). Colocando o bluetooth em desvantagem para competir com o zigbee nas aplicações de rede de sensores que o padrão zigbee se propõe.

Um ponto crucial na comparação dos dois padrões é o consumo de energia. Enquanto o zigbee foi projetado para um baixo consumo de energia com procedimentos de "hibernação" do dispositivo fazendo a sua bateria durar até anos, o padrão bluetooth tem um consumo mais alto de energia, sendo equipado com baterias recarregáveis. Essa característica impede que o padrão bluetooth concorra com o padrão zigbee em redes de sensores como iluminação, sistemas antiincêndio e outros que são o principal foco do protocolo.

Conclusão

Diante desta análise, o protocolo zigbee mostra-se tecnologicamente aderente as aplicações a que se propõe, com características que o colocam a frente dos seus possíveis concorrentes presentes no mercado.

Custos do produto, apelo comercial e potencial de mercado

Tendo claras as características do produto e as aplicações para as quais ele foi projetado e tomando como correta a conclusão acima de que ele é tecnologicamente compatível as aplicações a que se propõe, discutiremos agora custos e viabilidades comerciais.

O "Korean Advanced Instituteof scienceand Technology" fez um protótipo baseado nas especificações dispostas no protocolo 802.15.4 do IEEE e obteve como custo base deste protótipo o valor inferior a 1 dólar. Este custo pode ser considerado baixo se considerarmos aplicações que envolvam o "embarcamento" do chip em

eletrodomésticos, que tem custo consideravelmente superior a este 1 dólar. Mesmo se considerarmos aplicações cujos os aparelhos ligados aos nós da redes são mais baratos como sensores ou interruptores(sistemas de iluminação, sistemas antiincêndio, etc), este valor ainda é aplicável comercialmente para a automação de ambientes, ainda se comparado com soluções de mercado que são proprietárias e conseqüentemente menos flexíveis.

O "West Technology Research Solutions" prevê que em um futuro próximo haverá em média de 50 a 100 nós de rede zigbee por residência automatizada. Este número aliado a estimativa de custo de 1 dólar por nó de rede nos leva a concluir que o preço a ser pago(com a rede) para automatizar ambientes de uma casa em termos de iluminação, segurança, aquecimento, eletrodomésticos, etc, é um valor relativamente baixo para o consumidor diante do benefício considerado.

A indústria, hoje, para fazer soluções similares ao que o projeto zigbee proporcionará utilizasse normalmente se soluções específicas para cada aplicação e protocolos proprietários. Esta situação encarece os produtos além do que os produtos gerados a partir destas condições são menos flexíveis e limitados em seus benefícios. Com a adoção pelo mercado do padrão aberto zigbee(regulado pelo padrão 802.15.4 do IEEE) o mercado terá a sua disposição um produto que terá mais flexibilidade e segurança para o desenvolvimento de aplicações baseadas nesta tecnologia, tornando assim os produtos com maior valor agregado e seus custos menores pela produção em escala.

Diante das possibilidades técnicas que surge com este novo padrão e conseqüentemente a enorme gama de aplicações que podem ser construídas baseadas nesta tecnologia espera-se uma rápida ascendência na utilização desses dispositivos pelo mercado.

A empresa de pesquisas "ABI" estima que o mercado de controles domésticos cheguem a cerca de US$4 bilhões em 2008, sendo que cerca de US$900 milhões apenas o mercado norte americano, com cerca de 20 milhões de chips "embarcados".

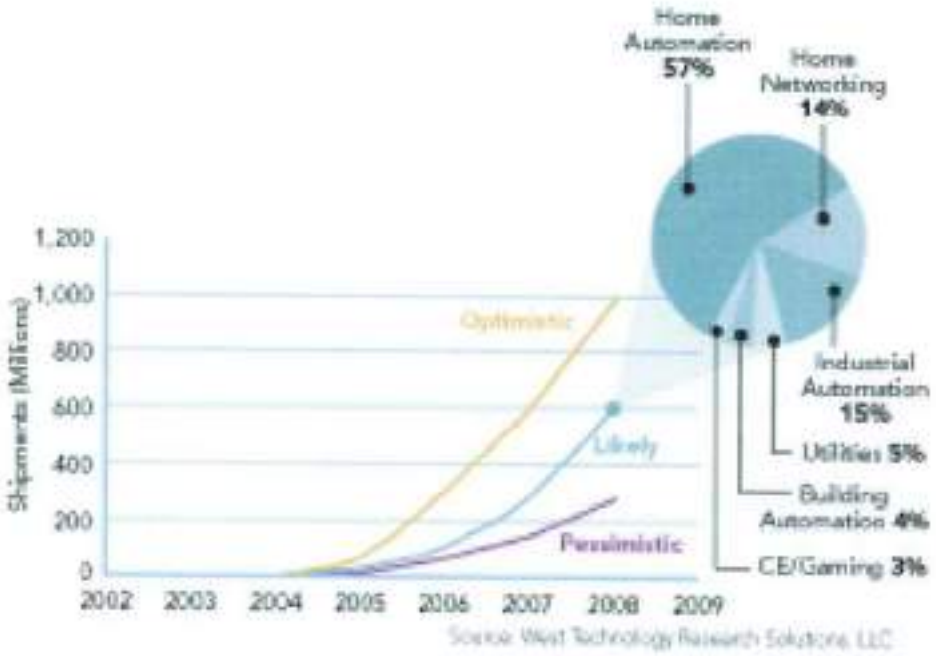


Figura 88- Colocação de chips no mercado.

Diante deste cenário, a conclusão é de que, com o potencial de aplicações baseadas na tecnologia e as previsões de crescimento de mercado, o padrão zigbee tem grande potencial de ser um sucesso comercial e servir de base tecnológica para que muitas empresas desenvolvam as suas aplicações e obtenham sucesso no mercado também.

SIMULADOR

O simulador possibilita a análise do comportamento de uma rede formada por um conjunto de nós que seguem o padrão zigbee de comunicação de dados wireless.
Para que esse estudo seja o mais aprofundado possível, em nosso simulador foi acrescentada a possibilidade de escolher um modo de operação. Para cada um deles, os respectivos detalhes mais importantes de cada uma das partes criticas envolvidas na comunicação são mostradas ao usuário.

A confecção do simulador teve como principal objetivo reproduzir o mais fielmente possível as estruturas das camadas, métodos utilizados em cada uma, variáveis, estruturas de dados e a nomenclatura para cada uma dessas.
Os nós foram definidos como blocos que encapsulam as camadas do protocolo zigbee, fornecendo a interface necessária de acesso ao meio físico.
Cada nó representa um dispositivo zigbee.
Armazenados dentro do nó encontramos as camadas:

- **Rede**
- **Mac**
- **Física**

Fora do nó temos apenas a o **meio físico**.
O protocolo zigbee até o presente momento não define um padrão para a camada de rede.
Essa parte da documentação não foi terminada e portanto, nos restou a tarefa de criar nosso próprio padrão.
O caminho estipulado pela equipe foi criar uma camada de rede com funcionalidades simples e utilizar o algoritmo Distance Vector, mais especificamente nos baseando no conhecido protocolo RIP. O motivo dessa escolha foi que isso nos agregaria maior conhecimento sobre esse assunto, além de possibilitar comparações de nossa implementação com aquilo que está descrito em sua documentação.
A camada Mac é a camada mais complexa de nosso projeto.
Apesar disso, é necessário ressaltar nossa dificuldade em entender algumas das funcionalidades dos serviços oferecidos por essa camada, efeito de não termos conseguido

a documentação da camada de rede, pois seria essa que descreveria o algoritmo de utilização dos serviços da camada inferior.

Como estudos extensos sobre a transmissão de sinais modulados pelo ar estão fora do escopo de nossa ênfase acadêmica, modelamos a camada física de maneira muito simples, apenas implementando-a como elemento de conectividade entre o meio físico e a camada Mac, sem levar em conta, por exemplo, o tempo de propagação das ondas eletromagnéticas. Além das camadas descritas no protocolo tivemos o cuidado em modelar o meio físico (ar) para que as transmissões acontecessem na mesma maneira em que um ambiente real adicionando, assim, tempo de transmissão e a possibilidade de colisões locais e remotas.

Não foi possível reproduzir totalmente os serviços especificados pelo padrão uma vez que o tamanho da especificação é muito extensa, saindo do escopo de um projeto de formatura.

Como seguimos o padrão documentado fica mais fácil para posteriores incrementos de funcionalidades e até mesmo de camadas, como por exemplo uma camada de aplicação com objetivos específicos.

Nosso programa é um simulador de tempo real, pois essa escolha é mais adequada ao problema. Como achamos interessante mostrar as saídas na tela ao invés de gerar um arquivo com os resultados obtidos, optar por um simulador de tempo discreto traria problemas além do fato de que um simulador de tempo real pode ser incrementado com maior facilidade por um programador que não esteja familiarizado com o código do simulador. Nossa intenção com a produção desse simulador é que ele sirva de ponto de partida para um futuro estudo do protocolo Zigbee.

O programa foi todo feito em linguagem Java. Esse fato foi decorrente de Java ser orientado a objeto, a alta portabilidade possibilitada pela máquina Virtual Java, o conjunto de APIs fornecidas(as quais aumentam a produtividade), além dos integrantes do grupo já terem uma certa familiaridade. Grande parte dos objetos foi modelada através de Threads para imitar o funcionamento exato de cada um dos dispositivos zigbee. A escolha da língua inglesa para a codificação do simulador (nome de métodos e variáveis) foi essencial para que continuássemos seguindo a nomenclatura utilizada nos documentos do padrão zigbee, para que tivéssemos um padrão na codificação além de possibilitar que programadores de outros países entendam facilmente o código uma vez que a língua inglesa é a mais popular

neste momento. Além da padronização da linguagem convencionamos adicionar um prefixo ao nome das variáveis para indicar a classe da qual essa variável faz parte.

Focamos nos esforços em produzir um simulador de alto nivel de desempenho e que fosse o mais próximo da especificação do padrão. Não no preocupamos em torná-lo visualmente atrativo para o usuário. Desta forma optamos por trabalhar com entradas e saidas de texto ao invés de utilizar recursos gráficos.

As entradas necessárias para o funcionamento do simulador são obtidas através de um arquivo. As informações de saída são impressas diretamente na tela do computador.

Uma grande bateria de testes foi realizada para nos certificarmos de que nosso projeto estaria funcionando em conformidade com o especificado.

DESCRIÇÃO DO MODELO DE CLASSES

Neste item fazemos a apresentação do modelo de classes do simulador. O digrama apresentado não contém os atributos ou métodos das classes. Optamos por fazer isso em razão do fato de que o número de métodos e de atributos ia tornar o diagrama excessivamente grande, dificultando sua visualização. Para maior detalhamento sobre os métodos e atributos de cada classe, sugere-se a consulta do próprio código-fonte, o qual está extensamente comentado.

Além do digrama de classes, fazemos em seguida uma breve descrição de cada classe, que serve como referência rápida para consulta sobre a função de cada classe no simulador.

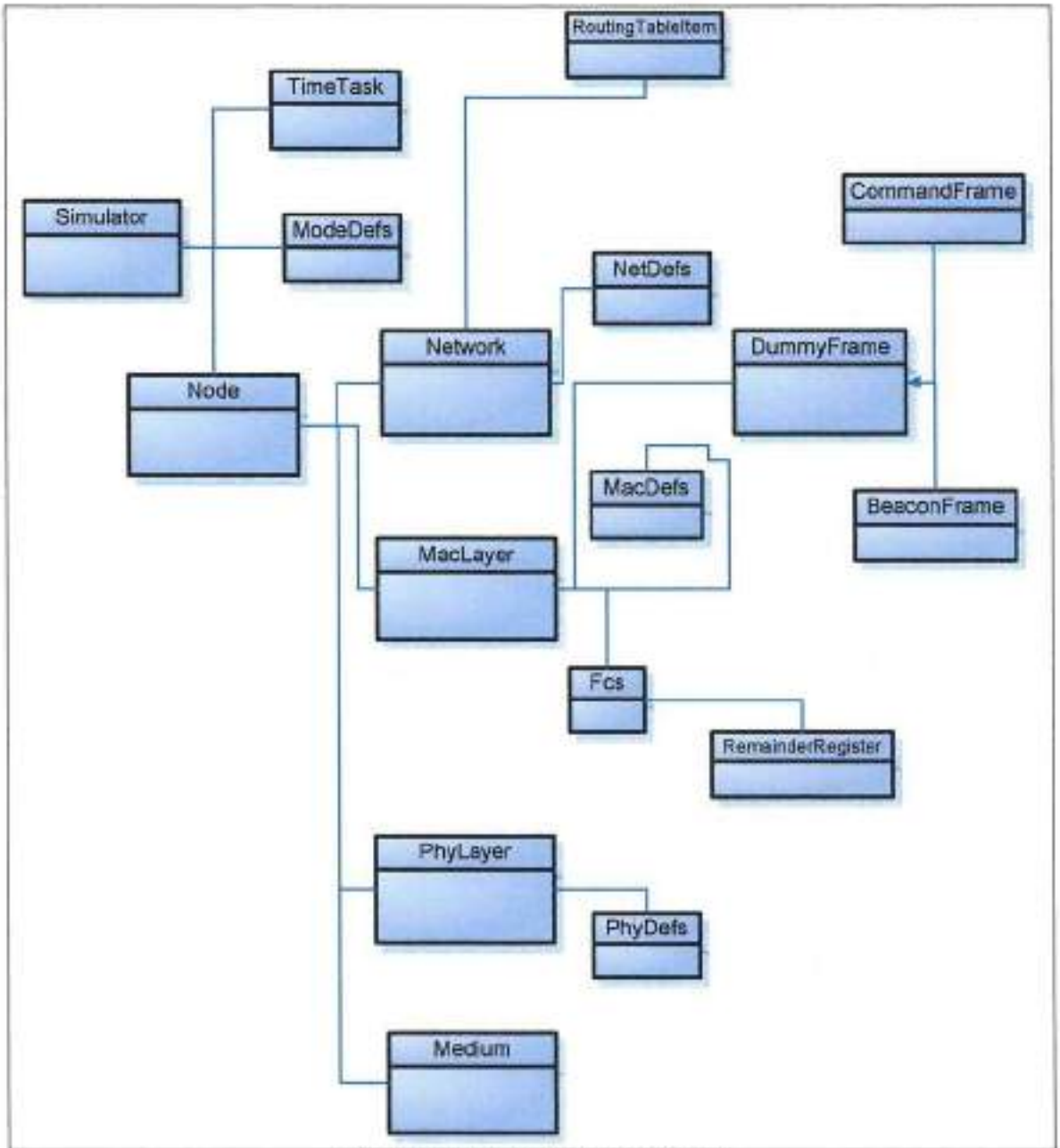


Diagrama de Classes Simplificado

Simulator
Classe que contém o método main do programa e que faz chamadas aos objetos das demais classes em função do modo de operação escolhido.

ModeDefs
Classe com as definições das constantes relativas aos modos de operação.

TimeTask
Classe responsável por terminar a simulação.

PhyLayer
Classe que implementa a camada física do protocolo IEEE 802.15.4.

PhyDefs
Classe com as definições das constantes relativas à camada física.

MacLayer
Classe que implementa a camada MAC do protocolo IEEE 802.15.4.

MacDefs
Classe com as definições das constantes relativas à camada MAC.

Fcs
Classe que possui um método estático que calcula o campo de FCS a partir de um frame passado como argumento desse método.

RemainderRegister
Classe que implementa estrutura de dados utilizada para calcular o FCS.

Network
Classe que implementa a camada de rede do simulador (não segue o padrão Zigbee).

NetDefs
Classe com as definições das constantes relativas à camada de rede.

RoutingTableItem
Classe que implementa um item (uma entrada) das tabelas de roteamento dos nós.

Node
Classe que representa um dispositivo da rede sendo simulada. Ele encapsula as camadas física, Mac e de rede.

Medium
Classe que implementa o meio físico por onde se dão as transmissões. Também é responsável pelo gerenciamento das colisões ocorridas na simulação.

DummyFrame
Classe que implementa os atributos comuns aos quatro tipos de frame do protocolo 802.15.4.

BeaconFrame
Classe que implementa um frame de tipo "beacon".

CommandFrame
Classe que implementa um frame do tipo "command".

DETALHAMENTO DA ESTRUTURA DO SIMULADOR

Classe Simulator

A classe Simulator tem como objetivo apenas operar os serviços fornecidos pelas camadas do protocolo zigbee. Diante dessa ótica ela pode ser entendida como uma camada de aplicação simplificada. Ela pode trabalhar em diversos modos de operação como descrito anteriormente (maiores detalhes podem ser obtidos na seção de instruções de uso do simulador).

Camada de Rede – Classe Network:

Como dito anteriormente, a principal dificuldade para produzir essa camada foi a questão da documentação que define o padrão zigbee ainda não estar terminada e justamente essa camada estar entre esses tópicos inacabados.
Apesar da descrição do algoritmo de funcionamento da camada de rede não estar presente na documentação obtida, duas formas de funcionamento já foram formalizadas:

- Star
- Cluster Tree

Na topologia Star todos os nós podem trocar informações apenas com o nó coordenador, ao passo que o método cluster tree possibilita que qualquer nó envie informação para qualquer outro.
A topologia cluster tree permite que a informação seja propagada para qualquer ponto da rede que tenha conectividade com o resto, pois a informação pode ser conduzida pelos nós até o destino final. A formação star permite apenas o acesso direto.
Para que a informação seja encaminhada através da rede é necessário que exista um protocolo de roteamento e de atualização das tabelas de roteamento.
Como a configuração Star é tão simples, pois apenas estabelece apenas um critério para as conversas entre nós, preferimos deixar esse fardo para a camada de aplicação (nossa classe

Simulator faz exatamente isso ao requisitar envio de mensagem apenas para nós específicos), focando nossos esforços em fazer nossa camada de rede compatível com a formação Cluster Tree.

Para tanto foi necessário definir um protocolo para tratar as tabelas de endereçamento e para conseguir encaminhar mensagens através da rede.

Para isso utilizamos um algoritmo distance-vector, nos baseando mais especificamente no funcionamento do protocolo RIP.

O resultado final foi a criação de um protocolo simples para controle dessas tabelas e roteamento de mensagens, eficiente e que atende todas nossas necessidades.

Mais adiante descreveremos os detalhes do funcionamento dos protocolos utilizados nessa camada.

Para que o nó consiga manter dados atualizados em suas tabelas de roteamento e trocar essas informações com os nós vizinhos criamos dois processos que devem ser ativados em cada nó.

O primeiro cuida da tarefa de descobrir os vizinhos do nó em questão (Neighbor Discover).

O segundo tem como objetivo trocar essas informações com os outros aparatos (Routing).

O método RipStart() é responsável pelo início das atividades desses dois processos.

O nome foi escolhido para essa função de forma que lembrasse a relação próxima entre nosso protocolo e os protocolos distance-vector.

Nossa camada de rede possui duas interfaces. A primeira provê serviços para uma camada superior e a segunda troca dados com a camada inferior (Mac)

O método responsável por enviar pacotes roteados através da rede chama-se sendPacket.

Para o recebimento de informações da camada Mac o acesso ao método receive deve ser feito.

Esse método é responsável por ler o cabeçalho de rede e tomar as medidas necessárias.

Vamos agora expor em maiores detalhes o protocolo.

As tarefas realizadas pela camada de rede podem ser divididas em blocos pequenos de forma a facilitar o entendimento.

Tabela de Roteamento:

A tabela de roteamento possui informações para qual deve ser o próximo nó para o qual a mensagem deve ser enviada para que ela consiga ser guiada até o destino final.
Cada nó possui a sua própria tabela.
Sempre que uma requisição de envio de mensagem (método sendPacket) é pedida, ou que uma mensagem chega até o nó sendo que este não é o destinatário desta mensagem, o nó deve verificar sua tabela de roteamento.
Essa tabela segue um padrão comum de tabelas de roteamento onde temos as informações:

- Endereço destino
- Next Hop
- Métrica

A métrica utilizada em nosso protocolo de roteamento é o número de Hops até o endereço de destino. Esta é a única métrica possível neste caso, uma vez que não existe diferença de taxa de transmissão, confiabilidade, atraso ou até mesmo custo para uma rota definida.
O endereço de next hop informa para qual nó deve ser enviada a mensagem de forma que ela chegue até o destino determinado pelo cabeçalho de rede armazenado no pacote.
Assim quando um aparelho é avistado diretamente por outro ele cria uma entrada na sua tabela (caso já não exista) indicando o endereço desse nó e colocando esse mesmo endereço como next hop com métrica 0.
Sempre que uma entrada for criada ou atualizada na tabela o valor do nexthop deve ser o endereço do dispositivo que lhe enviou a mensagem, pois esse será o dispositivo que saberá como conduzir a mensagem até o destino.
Como estamos estudando apenas ambientes estáticos em nosso simulador, uma entrada na tabela só poderá ser substituída por outra que leve ao mesmo endereço final com uma métrica menor. Em um ambiente dinâmico a possibilidade de retirada de uma entrada da tabela teria de ser implementada, pois um nó pode ser desligado, ativado, ou mudar de posição. Ações para evitar loops de roteamento (Poison Reverse, Split Horizon além de outras) também deveriam ser tomadas.

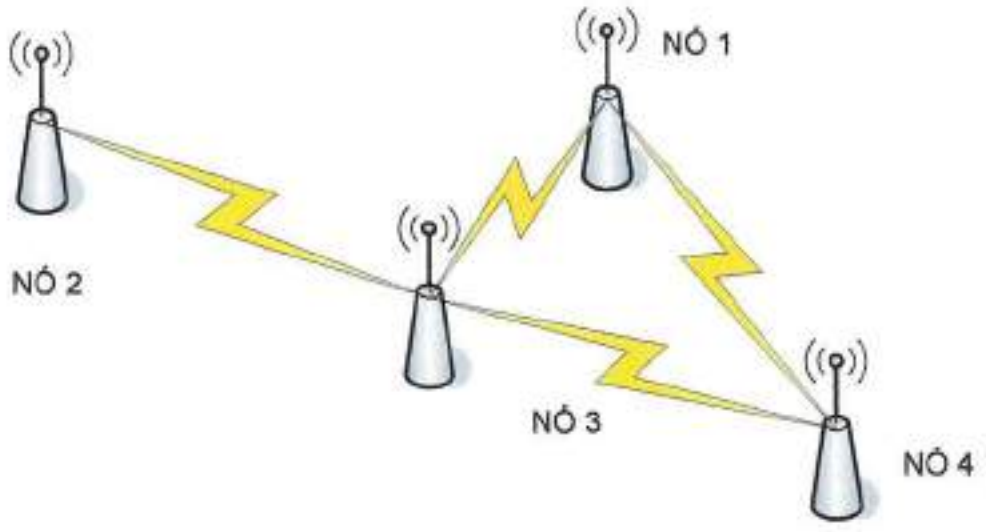
Temos a seguir um exemplo de uma topologia e as tabelas de roteamento para cada um dos nós após o protocolo ter convergido:

Tabela Roteamento		
Nó 1		
Endereço	NextHop	Métrica
2	3	1
3	3	0
4	4	0

Tabela Roteamento		
Nó 2		
Endereço	NextHop	Métrica
1	3	1
3	3	0
4	3	1

Tabela Roteamento		
Nó 3		
Endereço	NextHop	Métrica
1	1	0
2	2	0
4	4	0

Tabela Roteamento		
Nó 4		
Endereço	NextHop	Métrica
1	1	0
2	3	1
3	3	0



Exemplo de convergência de tabelas de roteamento

Neighbor Discover:

Esse processo tem como objetivo descobrir quais são os nós vizinhos (os quais estão no alcance) de um determinado nó. Para realizar essa tarefa formalizamos dois tipos de mensagens que podem ser trocadas.

Sempre que um dispositivo emitir uma mensagem de "HELLWORLD" os dispositivos que receberem essa mensagem devem responder com uma mensagem "HEREIAM".

A troca de mensagens é toda feita com endereçamento de broadcast, pois assim todos terão acesso às informações. Sempre que um nó conseguir escutar uma dessas mensagens ele deve tentar inserir o endereço do nó remetente em sua tabela de roteamento. Como as mensagens de camada de rede não se valem da utilização de pacotes com ack request, é necessário fazer com que a camada de rede de tempos em tempos reenvie as informações reiniciando o processo de neighbor discover, pois muitos dos pacotes são inutilizados por causa de colisões no meio. Apesar dos ambientes simulados serem estáticos, não estabelecemos um número máximo de vezes que esse processo deve ser reiniciado, pois por menor que seja sempre existirá a probabilidade de quem um nós ainda não tenha recebido corretamente mensagens dos outros e assim caso o processo de envio de "HELLWORLD" fosse terminado as tabelas estariam erradas podendo até mesmo causar a falta de comunicação com o resto da rede até o final da simulação.

Após testes percebemos que para um melhor desempenho do protocolo seria necessário fazer com que os nós não realizassem esse processo de maneira sincronizada. Para cada nó o processo de descobrimento deve começar em um período diferente, assim eles não emitiriam pacotes de "HELLWORLD" ao mesmo tempo, pois isso teria um impacto gritante no número de colisões na rede. Por esse motivo ao iniciar o processo de Neighbor Discover pela primeira vez o nó aguarda um tempo aleatório dentro de um intervalo pré-definido.

Routing

Esse processo se encarrega de trocar suas tabelas, com as informações obtidas pelo processo Neighbor Dicover, entre os nós para que esses conheçam caminhos para outros dispositivos além de seus vizinhos. Quando um aparato recebe uma tabela de um outro elemento ele tenta atualizá-la. Caso tenha sucesso ele envia sua tabela para seus vizinhos

(broadcast) e assim sucessivamente. As tabelas serão trocadas e convergirão estabilizando o processo.

Inicialmente a quantidade de informação trocada entre os nós é muito alta, mas uma vez que as tabelas começam a convergir, um menor número de atualizações acontecerá e dessa forma um menor número de pacotes é trocado.

Assim como no Neighbor Discover é necessário fazer com que as tabelas sejam reenviadas após intervalos de tempo determinados, além de fazer com que os nós realizem esse processo de maneira assíncrona em relação aos outros.

Diagramas de Blocos

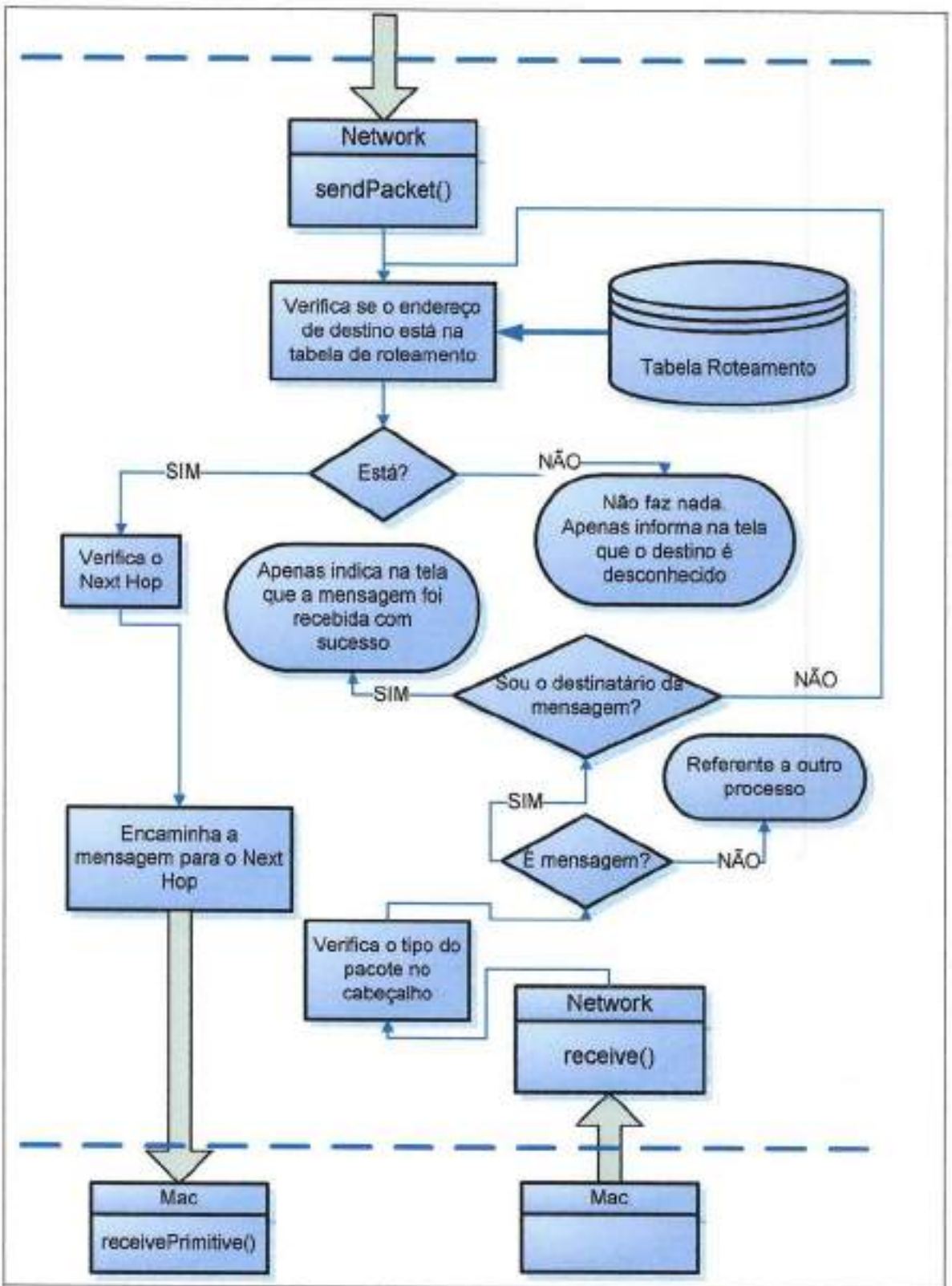


Diagrama do roteamento de mensagens

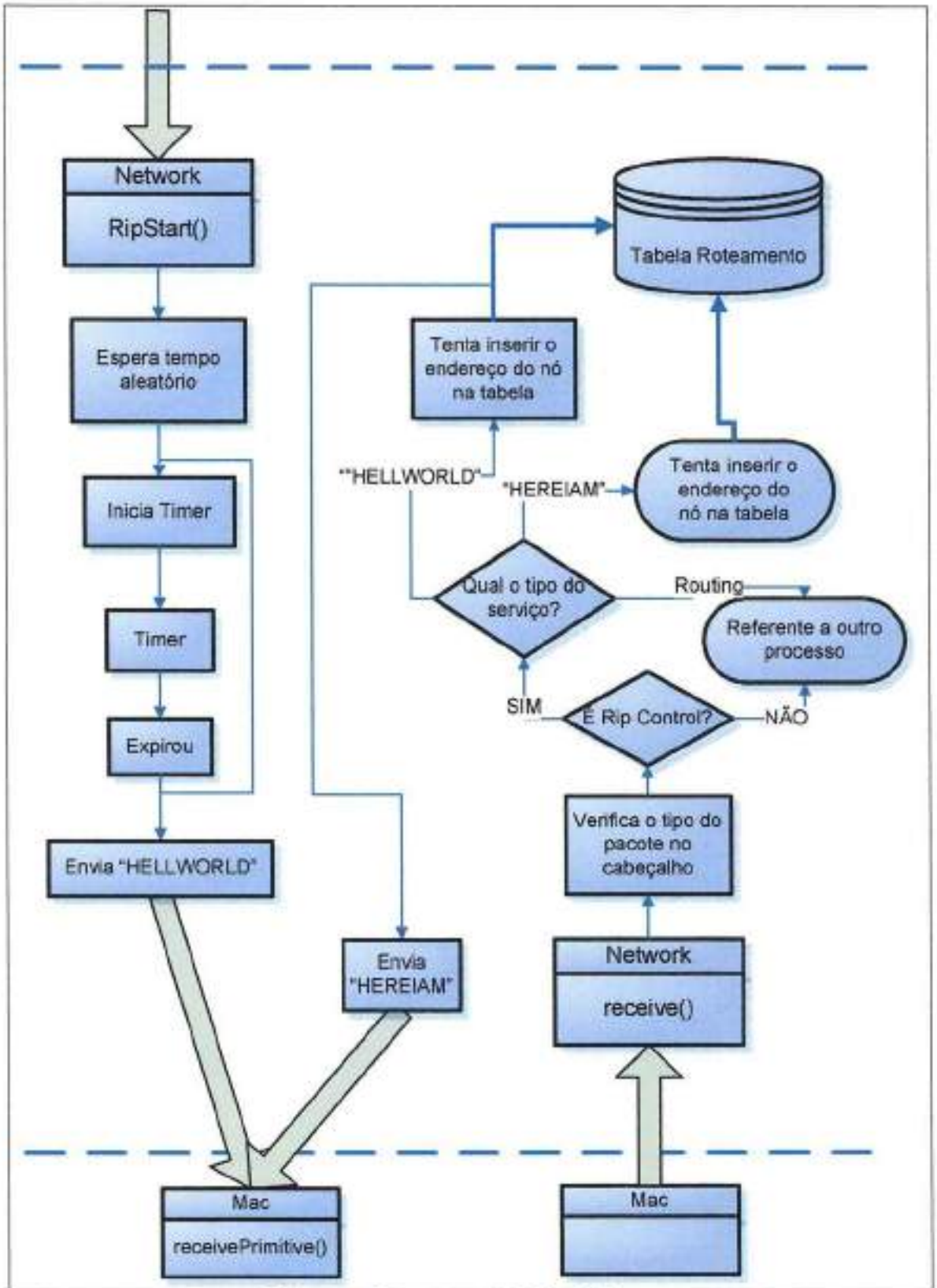


Diagrama do processo Neighbor Discover

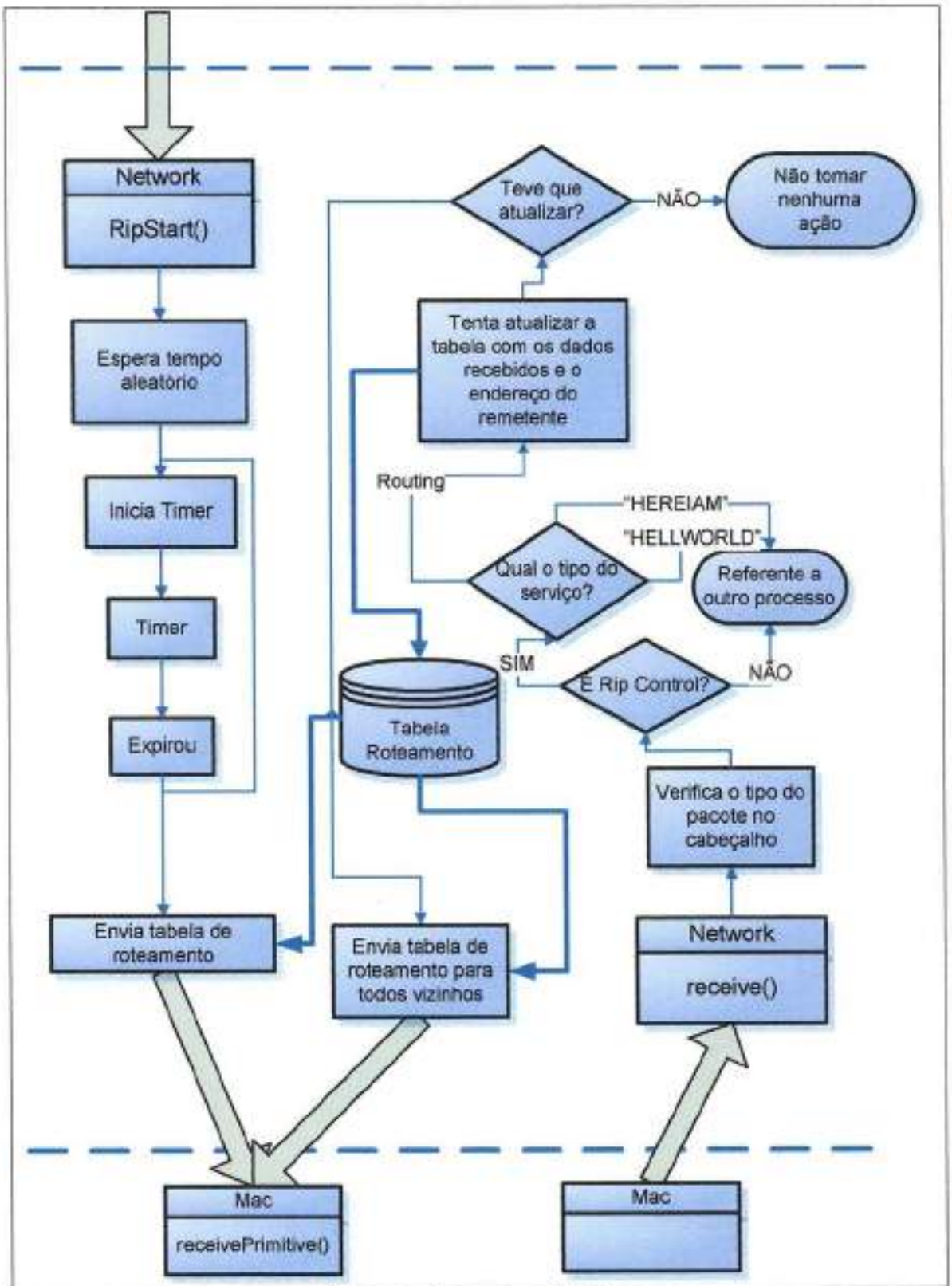


Diagrama do processo Routing

Cabeçalho das Mensagens de Rede

A camada de rede encapsula as mensagens que a camada de aplicação envia colocando um cabeçalho na frente da informação.

O cabeçalho da camada de rede possui tamanho e formato variável.

Três formatos de mensagem foram formalizados:

- "HELLWORLD" ou "HEREIAM"
- Tabela de roteamento
- Mensagem

- "HELLWORLD" ou "HEREIAM":

2 bytes	2 bytes
Tipo pacote	Serviço

- Tabela de roteamento

2 bytes	2 bytes	16 bytes	16 bytes	8 bytes	...	16 bytes	16 bytes
Tipo pacote	Serviço	Número de entradas na tabela	Endereço final (1)	Métrica (1)	....	Endereço final (n)	Métrica (n)

- Mensagem:

2 bytes	16 bytes	16 bytes	Não foi estipulado um limite
Tipo pacote	Endereço destino	Endereço origem	Payload

Os valores possíveis para esses campos são:

Tipo pacote	Valores
Rip Control	"00"
Message	"01"

Serviço	Valores
"HELLWORLD"	"00"
"HEREIAM"	"01"
Routing Table	"02"

A leitura das partes do cabeçalho é condicional. Por exemplo: Caso esteja presente o valor "01" no campo tipo do pacote é esperado que o próximo campo tenha 16 bytes e representará o endereço de destino.
É importante ressaltar que a escolha do tamanho de cada campo do cabeçalho e seus valores não objetivou a eficiência, sendo escolhidos por conveniência.

Comentários:

Foram realizados vários testes durante a elaboração do protocolo de camada de rede.
As decisões em relação a algoritmo e tempo de retransmissão de mensagens visam diminuir o tráfego na rede e conseqüentemente o numero de colisões. Elas foram tomadas baseadas nos resultados desses testes.
Algumas considerações devem ser feitas. Por causa das colisões no meio fisico em alguns momentos podemos ter tabelas de roteamento que não representam o melhor caminho até o destino final. Assim que as tabelas começam a convergir esses erros tendem a ser sanados.
Caso a camada de aplicação requisite o envio de uma mensagem sem que a convergência das tabelas tenha sido alcançada é possivel que a mensagem não consiga chegar ao destinatário apesar dele estar no raio de alcance da rede. É necessário, portanto, esperar um período antes que mensagens possam ser trocadas com confiança.
Quando um nó não sabe qual deve ser o próximo hop para o qual a mensagem deve ser enviada a mensagem de destino desconhecido é apenas impressa na tela, sem que a camada de aplicação do nó origem seja notificada.

CAMADA MAC - CLASSE MACLAYER:

A classe MacLayer tem a responsabilidade de implementar a camada MAC do protocolo 802.15.4. A implementação desta classe – que vem a ser a mais complexa do nosso simulador – foi baseada no documento do IEEE. Por se tratar de um software extremamente complexo e fortemente dependente de caracteristicas da camada fisica (que apresenta conceitos de telecomunicações, que estão fora do escopo do nosso projeto), não implementamos todas as suas funcionalidades, tais como descritas no item do relatório que fala sobre o protocolo 802.15.4. O que fizemos, foi implementar algumas de suas características mais representativas, de tal forma a fazer algumas demonstrações da

eficiência de tais características. A seguir, listamos as características implementadas em nossa camada MAC:

- Algoritmo de controle de colisões, *unslotted* CSMA-CA;
- Envio de beacons para implementação de mecanismo de transmissão indireta;
- Transmissão direta de dados entre dois nós da mesma PAN;
- Mecanismo de retransmissão de mensagens em transmissões com "ackRequest";
- Mecanismo de FCS para a rejeição de frames corrompidos.

Para deixar claro o que foi implementado na nossa camada MAC, eliminando dúvidas a respeito do que não foi considerado em nosso simulador, listamos aqui os itens principais que não foram implementados, mas que são definidos pelo protocolo:

- Criação de PANs a partir de um nó;
- Associação e desassociação de nós de uma PAN;
- Sincronização de nós com o coordenador através de beacons;
- Alocação de GTS.

Por se tratar de uma classe muito extensa, optamos por fazer uma descrição das principais primitivas de dados e serviços que foram implementadas (MCPS e MLME) e as funcionalidades a eles associadas. Para que se obtenha detalhes específicos da implementação, sugere-se a consulta ao código-fonte, o qual está extensamente comentado.

MCPS_DATA

Primitiva que implementa a solicitação de envio de dados pela camada superior (DATA-REQUEST), bem como a chegada de um novo frame de dados da camada física (DATA-INDICATION). Através desta primitiva, é possível solicitar o envio de um frame de dados de maneira direta ou indireta. No caso de transmissões diretas, a camada MAC deve montar um frame de dados a partir dos argumentos fornecidos e enviá-lo pela camada física. É possível que a camada superior solicite uma transmissão com reconhecimento, ou seja, a

flag ackRequest é igual a "true". Nesse caso, a camada MAC deve aguardar o retorno de um frame de ack, que indicará que o nó destino recebeu o frame de dados corretamente. Caso esse ack não chegue, a camada MAC deverá retransmitir o frame de dados, até que o ack seja recebido, confirmando o envio correto dos dados. Se o número de retransmissões chegar ao seu limite, o envio será cancelado e a transmissão será considerada uma transmissão falha. É possível enviar dados com o endereço MAC de destino igual a zero, o que se configura como o envio de mensagens de broadcast. Isto é usado nos modos de simulação que usam roteamento.

Há ainda o envio indireto de dados, no qual o nó coordenador de uma PAN habilitada com beacons deverá enviar nos seus frames de beacon a informação de quais nós possuem informações pendentes. Os nós que possuírem informações pendentes deverão fazer uma solicitação ao coordenador com um frame de comando, solicitando o dado pendente. Transmissões indiretas têm início quando enviamos um DATA-REQUEST ao nó coordenador com o flag indirectTx igual a "true". O coordenador, ao receber esta primitiva deverá, em lugar de transmitir diretamente o dado, armazená-lo em uma fila de transações pendentes. No próximo beacon enviado pelo coordenador, este deverá incluir no beacon uma lista contendo todos os endereços dos nós que têm pendências a serem recuperadas. No item "Testes" mostramos como o processo acontece com maiores detalhes e como observá-lo utilizando o simulador.

Relacionado ao envio de frames de dados e frames de comandos, está o uso do algoritmo de *unslotted* CSMA-CA (o algoritmo *slotted*, utilizado em redes PAN sincronizadas por beacons não foi implementado para o simulador). Este algoritmo previne o acontecimento de colisões no envio desses frames pelo meio físico (veja no item de Testes o impacto do algoritmo no número de colisões) através de um procedimento caracterizado por uma espera aleatória de um certo número de símbolos seguida pelo teste de CCA (vide lista de abreviações), que verifica se o meio físico está livre. Se o teste de CCA indicar que o meio está livre, a camada MAC envia o frame pela camada física. Caso contrário, repete o processo até que seja possível enviar o frame ou, até que o número máximo de backoffs (períodos em que o nó espera um número aleatório de símbolos) seja atingido, caso em que a transmissão é cancelada. É importante lembrar que frames de beacon e de ack não necessitam de usar o algoritmo de CSMA. No caso do beacon, este

possui uma porção do *Superframe* (vide glossário) reservada para si, durante a qual ninguém mais na PAN transmite nada. No caso dos acks, estes sempre são enviados imediatamente após a chegada de um pacote de dados, logo, desde que um certo limite de tempo não seja excedido, é possivel enviar o frame de ack imediatamente após a recepção da mensagem de dados, sem correr o risco de gerar colisões no meio fisico.

Por fim, a funcionalidade de FCS, associada à transmissão dos quatro tipos de frame utilizados no protocolo 80.15.4 – frame de dados, frame de comando, frame de ack e frame de beacon – é implementada para detectar frames que foram afetados por colisões no meio fisico (vide descrição da classe Medium para maiores detalhes sobre colisões no meio fisico). Todos os frames transmitidos recebem um campo formado por 16 bits que correspondem ao checksum do pacote. Ao chegar a um nó, o frame é testado quanto ao seu campo de FCS e se este teste acusar discrepância entre o campo de FCS e o que foi calculado do resto do frame, o frame inteiro é descartado imediatamente.

MLME-START

Esta primitiva está relacionada à interface da camada MAC que cuida dos serviços de gerenciamento da mesma (MLME). Ela faz com que o nó coordenador da PAN passe a enviar beacons periodicamente. O periodo de envio é função de um dos argumentos passados pela primitiva, BeaconOrder (BO). A partir desse valor a camada MAC calcula o tempo (em símbolos) decorrido entre o envio de dois beacons consecutivos.

CAMADA FÍSICA - CLASSE PHYLAYER

Como explicado anteriormente, o estudo da modulação do sinal eletromagnético está fora das nossas intenções com esse estudo. Desta forma modelamos essa classe apenas como um elemento de conectividade entre as classe Mac e Medium.

Quando a camada Mac deseja transmitir, ela verifica se a camada Física está desocupada. Assim que ela ficar livre, a camada Mac direciona o frame para ela e ela simplesmente repassará essa informação para a classe medium. Ela cuida somente do controle de transmissão e recebimento de frames.

MEIO FÍSICO - CLASSE MEDIUM

A classe Medium é responsável por modelar o meio físico por onde os dados são transmitidos. Para implementar essa classe, consideramos dois aspectos básicos relativos à transmissão de ondas eletromagnéticas no ar: alcance dos transmissores e colisões. Optamos por dar ênfase a estes aspectos pelo fato de que algumas funcionalidades do padrão 802.15.4 endereçam problemas relacionados a tais aspectos. A seguir, detalhamos ambos aspectos.

- Alcance dos transmissores

O alcance dos transmissores dos nós que compõem uma rede real do padrão 802.15.4 localiza-se na em uma faixa que varia aproximadamente de 20 a 75 metros. Isso requer que, obviamente, implementemos um meio de fazer com que os nós da simulação tenham também alcance finito, caso contrário a simulação seria prejudicada. Foi preciso definir uma forma de fazer com que um determinado nó tivesse um raio de alcance, além do qual, suas transmissões não seriam detectadas. Da mesma forma, nós localizados fora do raio de alcance de um determinado nó não poderão transmitir mensagens diretamente para ele. Conseguimos isto por meio de uma estrutura de dados, implementada na classe Medium, encarregada de manter um registro de todos os nós da simulação e quais nós cada nó pode alcançar. Esta estrutura de dados, med_reachListVector, será descrita em um item posterior.

- Colisões

As colisões são fenômenos caracterizados pela transmissão de duas ou mais ondas de rádio-freqüência no mesmo canal físico. Isso resulta na deterioração dos dados que se deseja transmitir, impondo uma perda na sua qualidade, o que pode até comprometer a comunicação. O protocolo 802.15.4 possui algumas funcionalidades que lhe permitem contornar essa restrição do meio físico, possibilitando o envio e recebimento de dados de forma confiável. Sendo assim, implementar o efeito das colisões nas transmissões pelo meio físico é de extrema importância se quisermos avaliar a performance do protocolo.

A maneira como implementamos o fenômeno das colisões também está relacionada com a estrutura de dados mencionada anteriormente, med_reachListVector. Contudo, antes de descrevê-la, atentemos aos dois tipos de colisões que nós consideramos em nosso programa: colisões locais e colisões remotas.

Colisões locais caracterizam-se por serem causadas por dispositivos que estão na mesma área de alcance, ou seja, dois nós que podem transmitir diretamente um para o outro tentam transmitir ao mesmo tempo, causando uma colisão. Colisões locais podem ser reduzidas se o nó que deseja transmitir, antes de fazê-lo, verifica se o meio físico está livre. É nisso que se baseia o algoritmo CSMA-CA, implementado na classe *MacLayer*.

Colisões remotas por sua vez apresentam a característica de não ocorrerem dentro da área de alcance do nó que está transmitindo. A diferença em relação às colisões locais é que as colisões remotas não podem ser evitadas através do algoritmo CSMA. Suponha que, na ilustração abaixo, o nós 1 e 3 podem alcançar o nó 2, mas estes não podem comunicar-se diretamente entre si, de modo que as transmissões do nó 1 não atingem o nó 3 e vice-versa. Portanto, o nó 1 consegue detectar se o nó 2 está transmitindo, mas não pode afirmar nada sobre o nó 3. Dessa forma, é possível que tanto o nó 1 quanto o nó 3 transmitam ao mesmo tempo. Nas regiões de alcance desses nós não haverá colisões, dado que um não recebe as transmissões do outro. Contudo, como o nó 2 pode receber transmissões de ambos, este estará sujeito a uma colisão que os nós 1 e 3 não puderam prever. Trata-se de uma colisão remota no nó 2.

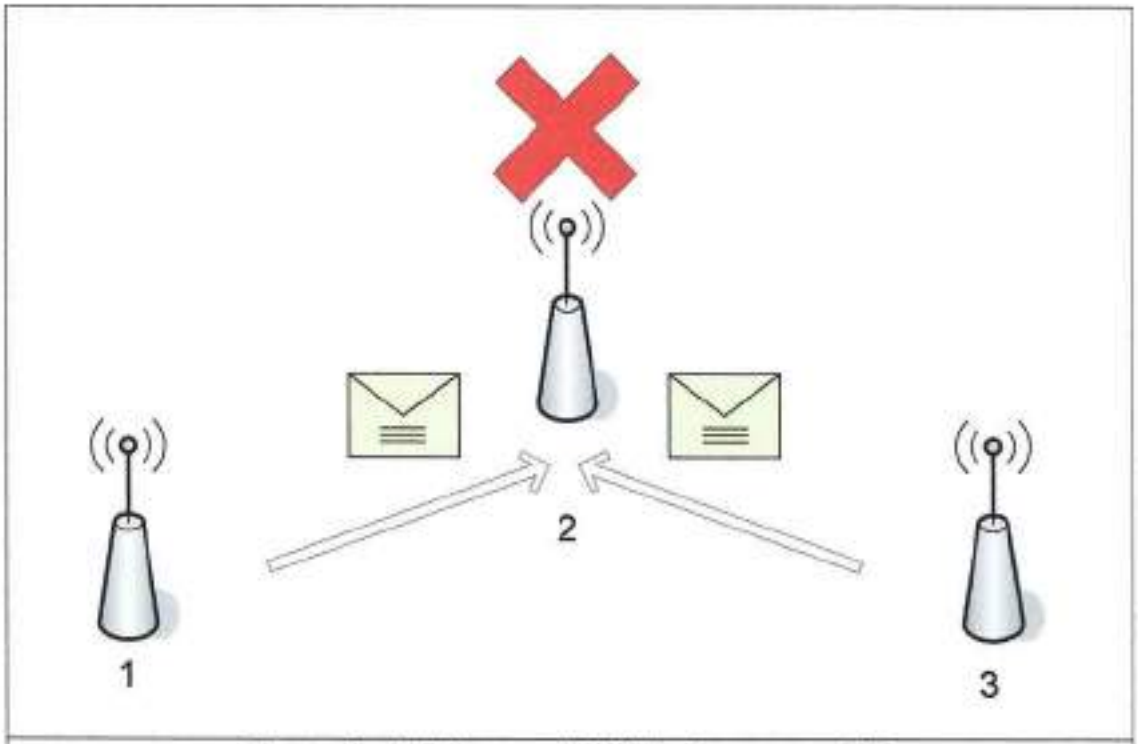


Ilustração do fenômeno de Colisão Remota no nó 2

Com relação à maneira com que modelamos o meio físico, é preciso lembrar que não consideramos os efeitos do tempo de propagação. Ou seja, desde que um dado nó esteja dentro do raio de alcance do nó que está transmitindo, este irá receber seu frame no mesmo instante em que os demais nós receberão, ainda que estes nós estejam mais próximos do nó que está transmitindo. Apesar de não considerarmos os efeitos do tempo de propagação, consideramos o efeito da taxa de transmissão. Consideramos que os nós do simulador utilizam a freqüência de 2400 MHz, uma das bandas de freqüências adotadas pelo Zigbee. Nesta freqüência, temos uma taxa de transmissão de 250 kbps e uma taxa de 62500 símbolos / segundo. Como a unidade de tempo adotada para o simulador é o símbolo, efetuamos o cálculo de quantos símbolos são necessários para o envio do frame. Para isso, basta notar que a taxa de 250 kbps corresponde a uma taxa de 4 bits por símbolo. Dessa maneira, o tempo de propagação em símbolos de um frame corresponde ao seu tamanho em bits dividido por 4.

Por fim, antes de procedermos à descrição da implementação do meio físico, em relação às colisões implementadas em nosso simulador, sempre que ocorrer uma colisão, o

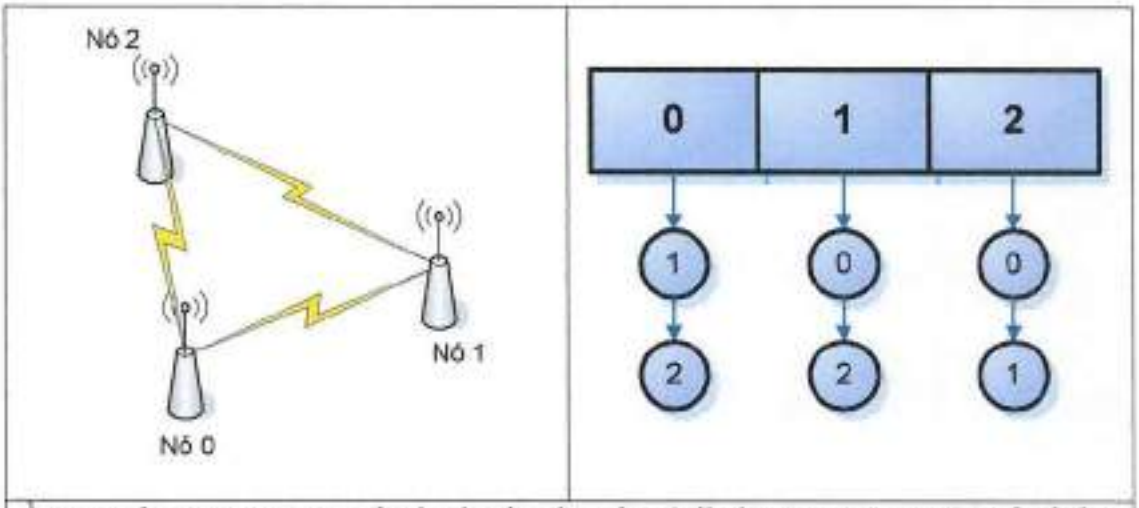
frame transmitido será completamente corrompido, o que causará sua rejeição no nó de destino, por causa de FCS.

Implementação da classe Medium

A classe Medium pode ser compreendida através da estrutura de dados utilizada para determinar o posicionamento dos nós no espaço. Antes de descrever esta estrutura, contudo, é preciso que entendamos como os frames são tratados por essa classe.

Para a classe Medium, os frames são apenas variáveis que devem ser propagadas aos nós de destino. A classe Medium não realiza operações sobre essas variáveis, a menos da "corrupção" destes frames em caso de colisões. Para determinar para quem deverá ser postada cada mensagem, a classe se baseia no id do nó que está transmitindo. Os frames, que representam o conjunto de bits sendo transmitidos no meio, são implementados na forma de String, uma classe da linguagem Java. Como estamos transmitindo pelo meio, essas Strings são compostas apenas dos caracteres '0' e '1'. Outra consideração importante a ser feita está relacionada ao tamanho máximo dos frames transmitidos pelo meio. No protocolo 802.15.4, existe um limite para o tamanho dos frames. Em nossa implementação, não há uma limitação para esse tamanho.

Para entender o funcionamento da classe Medium, é preciso entender a estrutura de dados que modela o meio físico e o alcance dos nós. Trata-se de um vetor de listas ligadas, med_reachListVector, implementado na classe Vector do Java. Cada elemento nesse vetor corresponde a uma lista ligada de objetos da classe nó (Node). A lista ligada é implementada através da classe LinkedList, do Java. O vetor de lista ligada é indexado através dos ids dos nós. Em cada posição desse vetor, temos uma lista ligada que corresponde à lista de alcance do nó cujo id é o índice desse elemento do vetor. Sendo assim, na posição 0 do vetor, encontramos todos os nós que podem ser alcançados a partir do nó 0. A figura a seguir ilustra uma topologia muito simples, composta por três nós, onde cada nó pode transmitir diretamente aos outros dois. Ao lado, mostramos como seria o vetor med_reachListVector correspondente a esta topologia.



À esquerda, temos uma topologia simples de rede e à direita, temos a estrutura de dados correspondente a esta topologia

Conforme já dissemos, as principais funções da classe Medium estão envolvidas com esta estrutura de dados. A primeira delas é a passagem da referência dos nós da simulação para o meio físico. Isso é feito através do método setNodeList( ), que fornece um vetor de objetos da classe Node. Com esse vetor, esse método cria uma lista ligada para cada um de seus elementos, criando a estrutura de dados med_reachListVector. Para cada nó do vetor fornecido como argumento deste método, determina-se quais outros nós podem ser alcançados pelo nó em questão, através de cálculos que se baseiam nas coordenadas x e y de cada nó. Todos os nós que estejam dentro do seu raio de alcance são incluídos na lista ligada. Por fim, a lista ligada construída é colocada na posição do vetor correspondente ao id do nó do qual acabou-se de construir a lista de alcance. O processo segue até que todas as listas de alcance já tenham sido criadas.

Outra operação essencial para a classe Medium é a verificação do meio físico, ou seja, os nós "perguntam" ao meio físico se este está ocupado. Trata-se do procedimento de CCA. Antes de transmitir, um nó pode tentar verificar se o meio está ocupado no seu raio de alcance. Isso é feito através do método performCCA( ). Este método simplesmente percorre a lista ligada do nó que deseja transmitir tentando descobrir se algum de seus nós está

transmitindo. Se for este o caso, significa que o meio físico está ocupado. Caso contrário, o meio está livre e o nó pode transmitir.

Por fim a última e mais importante função da classe Medium é a transmissão dos pacotes. Isso acontece através do método transmit( ), que recebe como argumentos o id do nó que está transmitindo e frame a ser transmitido. Esse método tem a incumbência de enviar o pacote a todos os nós alcançáveis pelo nó que solicitou a transmissão. Isso é feito percorrendo-se a lista de alcance (localizada no vetor de listas ligadas) do respectivo nó e transmitindo o pacote a cada um dos nós que estejam na lista de alcance. Contudo, neste momento é preciso detectar se haverá colisões locais ou remotas. Conforme dito anteriormente, a duração das transmissões é função do tamanho do frame (e da taxa de transmissão, que é fixa). Sempre que um frame é transmitido, não passamos o frame diretamente para o seu destino, caso contrário as transmissões seriam instantâneas. Em lugar disso, agendamos um timer que irá expirar no momento em que o pacote deve ser entregue ao seu destino. Quando o timer expira, ele passa o frame ao nó para o qual ele foi agendado. Ao implementarmos a temporização desta maneira, precisamos de um meio para saber se um dado nó está transmitindo dados ou não, o que, em última análise, nos permitirá concluir se uma transmissão irá resultar em colisão local. A maneira utilizada para armazenar o status de transmissão dos nós foi através de um vetor de variáveis booleanas, med_nodeTransmitting[]. Portanto, no momento de transmitir o frame, verificamos se algum dos nós da lista de alcance está transmitindo naquele momento. Se a resposta for sim, é preciso agendar a transmissão de um pacote ruidoso (corrompido) para todos os nós alcançáveis pelo nó que está causando a colisão local. Caso nenhum nó da lista de alcance esteja transmitindo, ainda precisamos determinar se vai ocorrer colisão remota. Para isso, além de percorrer a lista de alcance do nó que está transmitindo, temos também que percorrer a lista de alcance dos nós presentes na lista de alcance do nó que está transmitindo. Isso deve ser feito para ver se os nós que receberão o frame sendo transmitido não estão, naquele instante, sob a influência de outra transmissão feita por um de seus vizinhos (que não seja o próprio nó que está transmitindo). Se houver algum nó transmitindo na lista de alcance de um dos nós que receberão a atual transmissão, devemos agendar uma colisão remota para este nó. Podemos ver que a operação mais complicada é a de transmissão, pois ela faz uso de timers e ainda é disputada por várias threads (vale

lembrar que cada nó na simulação é uma thread), motivo que nos fez transformar este método em um método sincronizado (ou seja, apenas uma thread pode acessar esse método de cada vez), o que evita que colisões erradas aconteçam. Entendemos que não vale a pena descrever os pormenores de programação aqui, sugerindo-se a consulta direta ao código-fonte, o qual está extensamente comentado.

INSTRUÇÕES DE USO

O simulador recebe as entradas a partir de um arquivo texto simples codificado no padrão ANSI.

Siga os passos descritos adiante para conseguir executar a simulação pretendida:

1. Digite a localização do arquivo fornecendo o caminho completo até ele e pressione a tecla enter. Exemplo: C:\arquivo.txt
2. Caso o arquivo informado não exista ou não possa ser aberto o programa irá repetir a pergunta até que o usuário indique o caminho correto até um arquivo que possa ser lido, ou que ele cancele a operação.
3. É importante que o usuário ordene as informações de acordo com o padrão a seguir, pois caso a operação do simulador se tornará instável.
4. A simulação segue até que o tempo especificado no arquivo de configuração expire

Instruções para a criação do arquivo de configuração:

Os seguintes devem ser escritos no arquivo sem a presença de espaços ou outros caracteres especiais, a não ser que seja explicitamente definido por esse manual.

1. linha: Tempo de simulação em segundos
2. linha: frase que identifica o modo de operação do simulador(explicado mais adiante)
3. linha: Alcance dos transmissores dos nós na unidade desejada
   obs: Como dito na especificação um valor aceitável para o alcance estará aproximadamente entre 20 e 75 metros, apesar de ser possível inserir qualquer valor, sendo isso uma responsabilidade do usuário.
4. linha: Quantidade de nós.
5. linha em diante: Posição cartesiana de cada um dos nó (um por linha) na seguinte forma: x<ESPAÇO>y, onde x e y devem ser positivos e <ESPAÇO> representa a tecla espaço.

Como descrito anteriormente existem frases que identificam o modo de operação do simulador.

Existem 8 modos de operação. A seguir consta uma tabela com o modo de operação e a frase que deve ser escrita na segunda linha do arquivo para que esse modo seja ativado.

Modo de operação	Frase selecionadora (sem aspas)	Descrição
Colisão	"colisao"	Todos os nós tentaram transmitir informação, ao mesmo tempo, para o endereço 2
CSMA	"csma"	Todos os nós tentaram transmitir informação, ao mesmo tempo, para o endereço 2
Beacon	"beacon"	Nó 1 envia uma mensagem indiretamente para o nó 3
Data Request	"data request"	Nó 1 envia uma mensagem diretamente para o nó 2
Data Request com Ack para destino falso	"data request com ack para destino falso"	Nó 1 envia uma mensagem com ack request para um nó inexistente (total de nós + 1)
Data Request com Ack para destino real	"data request com ack para destino real"	Nó 1 envia uma mensagem com ack request para o nó 2
Rip	"rip"	O processo de atualização de tabelas é feito
Routing	"roteamento"	O nó 1 envia uma mensagem roteável para o nó 2, depois que as tabelas de roteamento convergiram

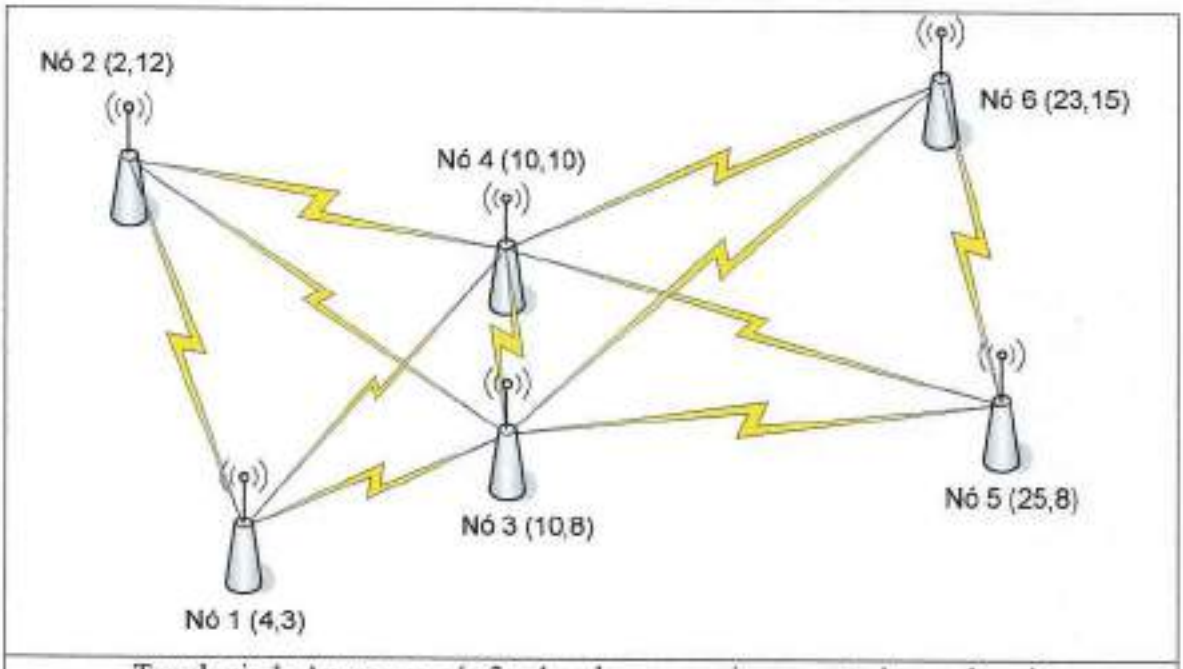
A diferença entre os modos de operação está apenas nas informações que são mostradas ao usuário e na forma como a camada de aplicação requisita os serviços.

TESTES

Para deixar a análise dos resultados dos testes mais simples, optamos por fazer a descrição separadamente, de acordo com os modos de funcionamento do nosso simulador. É importante insistir que tais modos de funcionamento não interferem na maneira como a simulação acontece, mas sim na ênfase que cada modo de funcionamento dá aos resultados da simulação, apresentando ao usuário apenas as informações relevantes ao modo de simulação escolhido. Dessa maneira, descrever os resultados separadamente torna a descrição mais simples e o entendimento dos resultados mais claro.

Para a realização dos testes, montamos diversas topologias, com um número variável de nós. Entretanto, algumas topologias específicas foram utilizadas mais extensamente, de modo que sua descrição neste momento é importante.

A primeira topologia importante, que chamaremos de Topologia 1, é uma rede composta por seis nós os quais não são mutuamente alcançáveis, ou seja, alguns nós não podem alcançar diretamente os demais. Essa rede está representada na figura abaixo:



Topologia 1: Apenas os nós 3 e 4 podem comunicar com todos os demais

Os nós que podem se intercomunicar são ligados por um raio. Podemos ver pela figura que apenas os nós 3 e 4 podem atingir diretamente todos os outros. Os nós 1 e 2 não conseguem "enxergar" os nós 5 e 6, o mesmo valendo para os nós 5 e 6 em relação aos nós 1 e 2. Essa rede é interessante no sentido em que para que haja uma transmissão entre todos os componentes da rede, é preciso de um esquema de roteamento de pacotes, o qual não está definido pelo protocolo 802.15.4. Conforme poderemos observar nos testes da camada de roteamento, é possível definir tabelas para cada um desses nós, o que possibilitará a comunicação entre quaisquer nós da rede. Essa figura também ilustra o fenômeno de

colisões remotas, pois os nós 3 e 4 podem receber, ao mesmo tempo, transmissões vindas do lado esquerdo (nós 1 e 2) como do lado direito (nós 5 e 6) da rede.

Outra topologia que merece destaque, e que chamaremos de topologia 2, é aquela em que todos os nós da rede são inter-alcançáveis, ou seja, cada um dos n componentes da rede é capaz de "enxergar" os n – 1 demais componentes. Conforme constatamos, o número de colisões locais nessa topologia é proporcional ao número de nós envolvidos. A posição exata de cada um dos nós não apresenta grande importância nesta configuração, pois, dado que os nós nessa topologia são inter-alcançáveis, independentemente de suas posições, eles estarão sujeitos a interferências dos demais nós (nossa simulação não considera o efeito do tempo de propagação do sinal).

Antes de descrevermos os testes, gostaríamos de lembrar que no fim da simulação (que ocorre depois que o tempo de simulação estipulado pelo usuário no arquivo de entrada acaba) são apresentadas as seguintes informações básicas, independentemente do modo de simulação escolhido:

- Número total de transmissões
- Número de colisões locais
- Número de colisões remotas
- Relação (total de colisões / total de transmissões)
- Número de desistências devido ao CSMA-CA

Tais informações são úteis, pois sumarizam os principais acontecimentos contáveis da simulação, permitindo-nos tirar algumas conclusões.

Testes de Colisão

Este modo de funcionamento do simulador (ativado quando escrevemos "colisao" na segunda linha do arquivo de entrada, a qual define o modo de simulação) dá ênfase aos fenômenos relativos ao meio físico de transmissão. Os fenômenos físicos que consideramos ao implementar nosso modelo de meio físico foram: alcance finito dos nós, colisões locais, colisões remotas e taxa de transmissão dos transceivers. Mais detalhes sobre estes fenômenos podem ser encontrados na descrição da classe Medium, que implementa o meio físico no simulador. Neste modo de funcionamento, o simulador exibe informações sobre

frames enviados e recebidos pelos nós, informações sobre erro de checksum (FCS) na chegada dos frames aos nós e ocorrências de colisões (locais ou remotas).

Os testes realizados indicam que, por ação do algoritmo de CSMA-CA, as colisões locais são consideravelmente diminuidas. Como o próximo item é dedicado exclusivamente ao CSMA, procuramos observar a influência de outras variáveis no número de colisões. A variável que mais impactou no teste de colisões foi o tráfego que desejamos impor na rede. Se o volume de dados transmitidos for muito alto, as chances do algoritmo vir a falhar são consideravelmente maiores, principalmente quando a topologia permite que possam ocorrer colisões remotas, as quais não podem ser detectadas pelo CSMA.

Foi realizado um teste que compara a mesma rede (topologia 1) sob duas condições muito diferentes de tráfego. Na primeira situação, fazemos com que os seis nós enviem frames de dados (um frame enviado por cada nó). Na segunda situação, impomos um tráfego muito mais intenso: a troca de tabelas durante a execução do algoritmo RIP de roteamento (discutido mais adiante). A tabela abaixo mostra os resultados obtidos, em termos de colisões / transmissões:

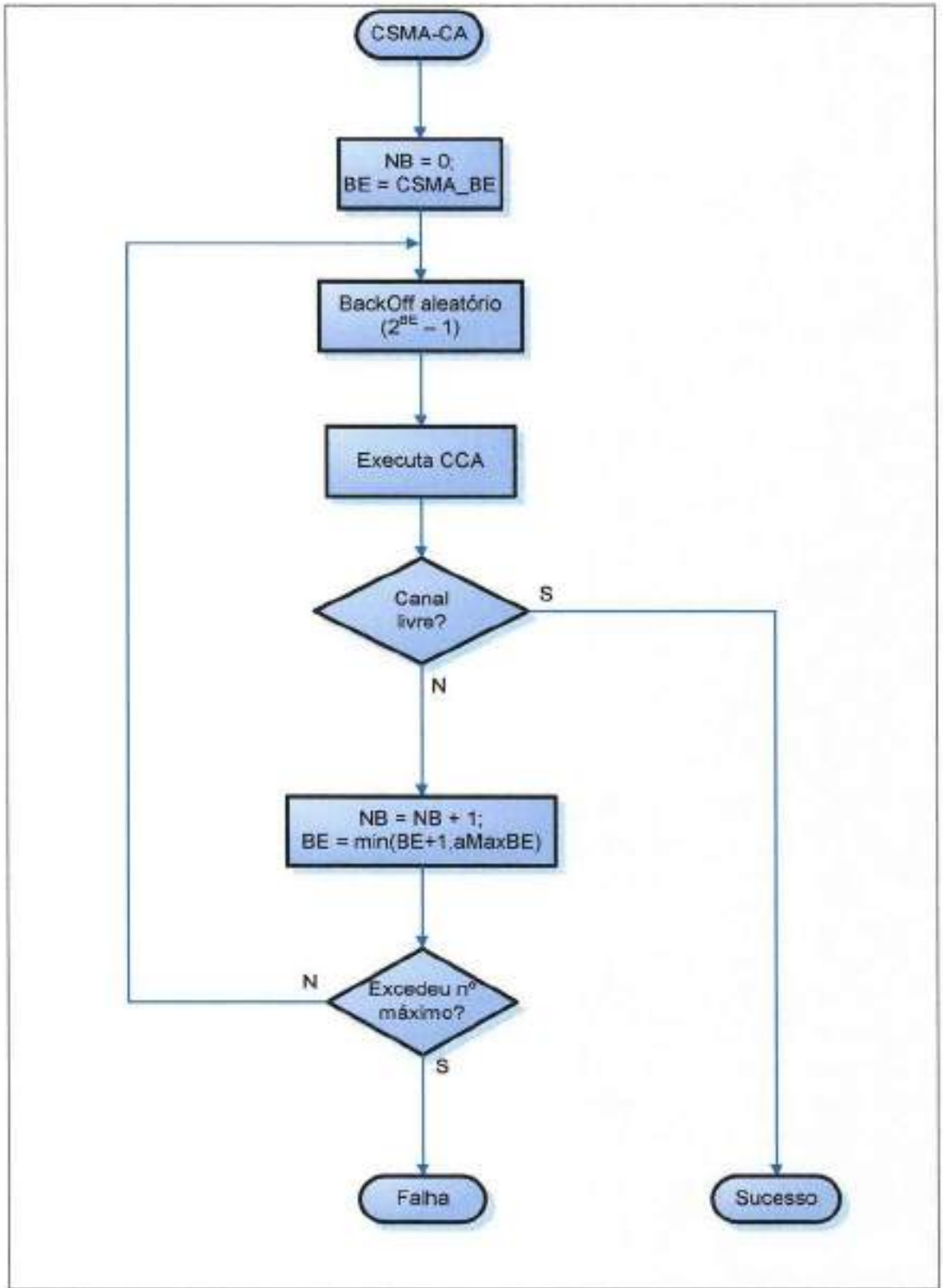
Situação 1	Situação 2
16 %	57 %

Como podemos observar, o volume de tráfego na rede tem grande influência no número de colisões. Em ambos os casos, a duração da simulação foi a mesma, o que garante que as duas situações diferem apenas pelo número de mensagens transmitidas. Para se ter uma idéia, na primeira situação foram transmitidos seis frames pela rede, ao passo que na segunda situação, foram transmitidos 88 frames.

Testes do algoritmo de CSMA-CA

Este modo de funcionamento (ativado quando escrevemos "csma" na segunda linha do arquivo de entrada) contempla informações relativas ao algoritmo de CSMA-CA, implementado na camada MAC do protocolo 802.15.4. Cabe aqui uma breve descrição deste algoritmo, que se assemelha em muito ao algoritmo CSMA-CD, utilizado em redes Ethernet. O fluxograma abaixo mostra o funcionamento do algoritmo CSMA unslotted,

implementado no simulador. O protocolo 802.15.4 também utiliza o algoritmo CSMA slotted para redes que utilizam beacons. A versão slotted não foi implementada no simulador. A figura abaixo representa a versão unslotted do CSMA-CA:



Algoritmo unslotted CSMA-CA

Em linhas gerais, o CSMA faz com que, antes de solicitar à camada física o envio de uma mensagem, a camada MAC aguarde um número aleatório de símbolos (unidade de tempo do protocolo) antes de executar o CCA, que verifica se o canal de transmissão encontra-se livre. Esse período de espera é chamado de período de backoff, que é sucedido pelo teste de CCA. Caso este teste indique que o meio está livre, a transmissão é iniciada, caso contrário, é preciso fazer mais uma espera de backoff antes de uma nova tentativa de transmissão. Se o número de backoffs exceder ao seu limite, a transmissão é cancelada e o frame descartado. O tempo de backoff é calculado multiplicando-se um número randômico entre 0 e (2BE – 1) por 20, que é a unidade básica do período de backoff. A variável BE (Backoff Exponential) tem o valor inicial a CSMA_BE, constante definida na classe MacDefs. A cada iteração do algoritmo, os valores de BE e de NB são incrementados. O valor de BE é incrementado até o valor máximo de aMaxBE, também definido em MacDefs.

Além das informações básicas, são apresentadas ao usuário informações sobre o tempo de backoff escolhido por cada nó e eventos de desistência de transmissão.

Realizamos um teste que mostrou como as transmissões ocorreriam sem o algoritmo de CSMA. O teste foi realizado da seguinte maneira: foi utilizada a mesma topologia (tipo 2) em duas execuções do simulador. Na primeira, submetemos a rede a um volume de tráfego em que os nós não utilizavam o algoritmo CSMA. Na segunda execução, a mesma rede e o mesmo volume de tráfego foram testados, mas desta vez, os nós utilizavam o algoritmo CSMA. A diferença na relação colisões/transmissões foi brutal:

Sem CSMA	Com CSMA
100 %	0 %

Apesar de ser um teste simples, ele nos mostra como o algoritmo é importante. Cabem duas ressalvas a este teste. A primeira é que ele não pode ser realizado no simulador, uma vez que não é possível desabilitar o algoritmo CSMA em nenhum dos modos de operação. O teste foi realizado alterando o simulador diretamente em seu código

fonte. A segunda ressalva relaciona-se ao número de nós da simulação, os resultados em porcentagens continuarão os mesmos, porém, o número de transmissões que deixarão de ocorrer na simulação realizada com o CSMA será cada vez maior, devido ao fato de que o número de backoffs excederá o limite máximo.

Ademais, esse modo de funcionamento do simulador nos permite determinar, conforme já dissemos, a duração do período de backoff de cada nó e para os casos em que o teste de CCA indica que o meio está ocupado, a duração do novo período de backoff, que na segunda vez é um pouco maior que na primeira por causa do incremento da variável BE. Veja na saída abaixo o momento em que o nó 3 define seu período de backoff pela segunda vez, após uma tentativa frustrada de acessar o meio:

```
O nó: 2 está executando o algoritmo de CSMA
O nó 2 definiu seu BackOFF = 5
O nó: 3 está executando o algoritmo de CSMA
O nó 3 definiu seu BackOFF = 3
O nó: 4 está executando o algoritmo de CSMA
O nó 4 definiu seu BackOFF = 4
O nó 3 definiu seu BackOFF = 12
```

Teste de envio de Beacons

Este modo de funcionamento configura um nó para ser o coordenador de uma PAN. Esse modo de funcionamento consiste de uma demonstração de uma funcionalidade do protocolo 802.15.4, não sendo útil para testar a performance do mecanismo de envio de beacons. A razão disso é que em nosso simulador foi implementada apenas a funcionalidade de transmissão indireta de informações entre coordenador e os demais nós. O simulador não implementa um esquema de sincronização por beacons, em que cada nó, ao receber um beacon, configura seus transceivers para ficarem habilitados apenas durante o instante em que o próximo beacon será transmitido. O mecanismo de transmissão indireta

de informações, o qual este modo de funcionamento se propõe a demonstrar, está descrito a seguir.

Primeiramente, é enviada a um dos nós da simulação a primitiva de START-REQUEST, a qual define o intervalo de transmissão dos beacons. Após o envio desta primitiva, o nó coordenador passa a enviar periodicamente frames de beacon no intervalo de tempo especificado pela primitiva fornecida. Nos beacons é enviada uma lista de todas as transações pendentes armazenadas no coordenador. O coordenador aumenta o número de transações pendentes quando recebe uma primitiva de DATA-REQUEST da camada superior com o flag de transmissão indireta (indirectTx)(1) setado. Nesse caso, ao invés de transmitir diretamente o pacote, o coordenador o armazena em uma fila de transações pendentes. Todos os beacons transmitidos contêm uma lista de quais nós da PAN possuem transações pendentes no nó coordenador. Ao receber um beacon, um nó checa a lista de transações pendentes nele contida, buscando encontrar seu próprio endereço MAC. Caso o encontre, o nó deverá enviar um frame de comando do tipo DATA-REQUEST para o coordenador, solicitando que este o envie seu frame pendente. Note que se trata de um frame de comando do tipo DATA-REQUEST, e não de uma primitiva. O nó coordenador, ao receber o frame de comando solicitando a transação pendente, deve procurar em sua lista de transações pendentes pela transação solicitada. Ao encontrar a transação requerida pelo nó, o coordenador despacha o frame solicitado para o nó envolvido. Este mecanismo permite que os nós da PAN fiquem inativos durante a maior parte do Superframe (vide glossário), acordando apenas nos momentos em que o beacon for transmitido e possibilitando uma grande economia da energia dos nós.

Para demonstrar o mecanismo de transmissão indireta de informação, basta executar o simulador no modo de transmissão de beacons (para isso, basta escrever beacon na segunda linha do arquivo de entrada). Nesse modo, passamos uma primitiva ao nó de endereço MAC 1 dizendo-lhe para enviar beacons. Nessa primitiva, passamos um valor que é utilizado para calcular o intervalo de tempo de transmissão de beacons, o BeaconOrder. Nesse modo se operação, passamos à camada MAC um valor de 3 para o BeaconOrder, que

[1] O protocolo 802.15.4 define que apenas os nós coordenadores podem receber o flag indirectTx setado. Como o simulador não implementa o início de uma PAN, essa verificação não é realizada. Maiores detalhes poderão ser encontrados na especificação do simulador.

define um intervalo de tempo de 7680 símbolos (os quais, aos olhos do usuário passam como 7,68 segundos). Em seguida, adicionamos uma mensagem pendente para o nó de endereço MAC 3, através do envio da primitiva DATA-REQUEST para o nó coordenador. Com isso, o simulador exibe os seguintes eventos em sua saída:

- Envio de um novo beacon pelo nó coordenador e sua lista de endereços pendentes;
- Recebimento do beacon pelos demais nós – se o endereço de um nó estiver na lista de pendências do beacon, é exibida uma mensagem informando que o nó irá tentar recuperar a pendência do coordenador;
- Recebimento de um comando de DATA-REQUEST pelo coordenador e uma mensagem dizendo que este está enviando a informação pendente ao solicitante;
- Recebimento do frame de dados pelo nó que tinha uma informação pendente.

Com esses eventos, podemos comprovar o correto funcionamento do mecanismo de transmissão indireta de dados.

Teste do envio de mensagens sem "ack"

Este modo (assim como os dois seguintes) também é uma demonstração de uma funcionalidade do protocolo 802.15.4 implementada em nosso simulador. Trata-se do envio direto de mensagens (vimos o mecanismo de envio indireto no item anterior). O mecanismo de envio direto de informações é muito simples: basta a camada superior enviar uma primitiva de DATA-REQUEST com o flag de transmissão indireta com valor zero. Isso fará com que a camada MAC, ao invés de colocar a mensagem em uma lista de pacotes pendentes, irá solicitar o envio do frame para a camada física (PHY), após executar o algoritmo de CSMA-CA. Quando executamos o simulador neste modo (escrevendo "data request" na segunda linha do arquivo de entrada), fazemos com que o nó de endereço MAC 1 receba uma primitiva de DATA-REQUEST que fará com que este envie um frame de dados diretamente ao nó de endereço MAC igual a 2. Como nessa transmissão o flag de ackRequest não está setado, a transmissão acaba quando o nó de destino do frame recebe a mensagem.

Na saída, o usuário pode ver o envio do frame de dados pelo nó 1 e o seu recebimento pelo nó 2.

Teste do envio de mensagens com "ack" para destino inexistente

Este modo de operação demonstra o funcionamento da funcionalidade de envio direto de frames com o uso de "acknowledgement". Um nó tenta enviar um frame de dados para um nó que não existe na simulação. Como o nó de destino não existe, o nó que originou a transmissão não receberá o frame de ack, fazendo com que o timeout de retransmissão expire. Dessa forma, o nó deverá retransmitir o frame até que o número máximo de retransmissões permitido atinja o seu limite.

Nesse modo de operação do simulador (acionado quando escrevemos "data request com ack para destino falso" na segunda linha do arquivo de entrada) envia-se uma primitiva de DATA-REQUEST com o flag de ackRequest setado para o nó de endereço MAC igual a 1. Essa primitiva especifica que o frame deverá ser enviado para um nó cujo endereço MAC não existe nesta simulação, fazendo com que o frame transmitido seja desprezado pelos nós que o receberem. Ao usuário é mostrado o envio do frame bem como as tentativas de retransmissão após o tempo de timeout. Quando o número de retransmissões atinge o limite, é mostrada uma mensagem dizendo que a transmissão falhou porque o destino não respondeu com uma mensagem de ack. Dessa forma, podemos verificar o mecanismo de retransmissões no caso de transmissões diretas com ackRequest.

Teste do envio de mensagens com "ack" para destino real

Este modo de operação do simulador (ativado quando escrevemos "data request com ack para destino real" na segunda linha do arquivo de entrada) assemelha-se muito com o modo descrito no item anterior. A diferença reside no fato de que neste modo de operação, o frame é transmitido para um nó que existe na simulação e que, portanto, reponde ao frame recebido com um frame de "acknowledgement". Logo, neste modo de funcionamento não ocorrem retransmissões, uma vez que o nó de destino existe e devolve o frame de ack solicitado.

As saídas exibidas ao usuário são: o envio do frame, a chegada do frame em seu destino e a resposta do frame de destino através de um frame de ack.

Teste do modo de roteamento RIP

Neste modo de funcionamento (ativado quando escrevemos "rip" na segunda linha do arquivo de entrada), diferentemente dos cinco anteriores, que testavam recursos da camada MAC, fala sobre o software da camada de rede, responsável por estabelecer roteamento entre os nós que compõem a simulação. Neste momento devemos lembrar um fato importante: o algoritmo implementado para a camada de rede do nosso simulador não corresponde ao algoritmo utilizado pela Aliança Zigbee no protocolo de rede do padrão Zigbee. A razão da escolha de uma versão simplificada do protocolo RIP, que implementa o algoritmo de "distance vector", deve-se ao fato de que no momento em que especificamos o nosso simulador, a Aliança Zigbee não havia ainda divulgado a especificação da camada de rede do padrão Zigbee ao público em geral. Além disso, o algoritmo "distance vector" é bastante conhecido no meio acadêmico, o que nos permite tirar conclusões baseados em um algoritmo compreendido por todos(2).

A camada de rede implementada nos nós, para criar a tabela de roteamento, envia mensagens de broadcast pela rede periodicamente, enviando informações da sua tabela para os demais nós. Isso gera um imenso tráfego na rede, refletido pelo elevado índice de colisões nas transmissões (um valor da ordem de 60% das colisões), indicando que o envio de pacotes pela rede através de broadcast não é o melhor cenário para utilizarmos o protocolo 802.15.4. Uma opção que possivelmente ofereceria um melhor resultado no número de colisões e que, portanto, seria mais adequado ao protocolo MAC seria através de um esquema em que os coordenadores das PANs seriam os responsáveis pelo roteamento, uma vez que cada coordenador conhece o endereço dos nós que estão dentro de suas respectivas PANs. Ao invés de cada nó transmitir suas tabelas de roteamento, apenas os coordenadores transmitiriam suas tabelas para os nós que sabidamente podem alcançar outros nós que o coordenador não pode. Não fizemos uso deste esquema de roteamento porque ele exige algumas funcionalidades do protocolo 802.15.4 as quais não foram implementadas no nosso simulador.

Apesar do elevado número de colisões, os testes realizados sugerem que o uso do algoritmo "distance vector" na camada de rede dos nós faz com que as tabelas de roteamento de cada um dos nós venha a convergir ao seu estado "ótimo" em um tempo

[2] Maiores detalhes sobre a camada de rede e seu algoritmo utilizados no simulador podem ser encontrados na sua especificação.

satisfatório. Por exemplo, uma rede montada de acordo com a topologia 1, apresenta convergência das tabelas em menos de 1 minuto de simulação (o que corresponde, em termos de tempo real, a menos de um segundo, ou seja, 60000 símbolos). Vale ressaltar que entendemos como tabela ótima como a tabela de roteamento que indica o caminho mais curto, em termos de número de hops, até o destino. Durante a execução do simulador nesse modo, podemos ver que cada nó vai construindo sua tabela a partir das tabelas enviadas pelos outros nós. Nesse processo, podemos observar que no início da simulação, as tabelas apresentam caminhos com métricas maiores à situação ótima, já que muitas tabelas transmitidas se perdem com colisões e os nós acabam inserindo em suas tabelas informações enviadas por outros nós, fazendo com que um nó diretamente alcançável pareça alcançável através de um nó intermediário, colocando um hop desnecessário à rota. Entretanto, com o passar do tempo, os nós acabam recebendo as tabelas de todos os outros nós, inserindo em suas tabelas apenas as rotas de menores métricas. Recomenda-se a observação da evolução das tabelas dos nós ao longo da simulação, até o instante em que estas cheguem a condição ótima.

Teste do envio de mensagens por uma rede roteada

Neste modo de operação do simulador (acionado quando escrevemos "roteamento" na segunda linha do arquivo de entrada), podemos ver o envio de um frame de um nó até outro, sendo que estes nós não necessariamente encontram-se no mesmo raio de alcance. Por exemplo, na topologia 1, poderíamos fazer com que o nó 1 envie um frame até o nó 6, mesmo que este não esteja em sua área de alcance. Contudo, antes de fazermos a transmissão entre dois nós, é preciso gerar as tabelas de roteamento dos nós da simulação. Dessa forma, o simulador executa o algoritmo de roteamento (cujo teste foi descrito no tem anterior) e após que se verifique que as tabelas de todos os nós estejam completas, o simulador faz com que um dos nós transmita um pacote para outro nó da rede. Nesse modo, o simulador exibe o nó de origem enviando a mensagem, as passagens dessa mensagem pelos nós intermediários e a sua chegada no nó de destino.

Os testes realizados nesse modo nos mostraram que a mensagem segue pela rede de acordo com o que está definido nas tabelas de roteamento, mostrando que o esquema de transmissão de mensagens pela rede roteada é eficiente, evitando a ocorrência de loops.

CONCLUSÃO

O projeto mostrou inúmeros desafios desde a sua concepção. Dificuldades para a obtenção da documentação nos fizeram mudar o escopo do projeto em um momento crítico. Tratou-se, portanto, de um grande exercício de flexibilidade para que pudéssemos nos adaptar à nova realidade do projeto.

Após delinearmos uma nova estratégia, tivemos dificuldades práticas no que se refere à parte de programação. Primeiramente, constatamos que a especificação do protocolo da camada de enlace (MAC) poderia ser considerada, dada sua grande complexidade, mais do que se espera de um projeto de formatura. Além disso, tivemos que resolver o problema de sincronismo entre as classes, que deve levar em consideração requisitos de aplicações de tempo real, o que envolveu o uso de threads e a necessidade de sincronizá-las.

Dessa forma, pudemos constatar que aspectos importantíssimos que permeiam um projeto complexo tais como a mensuração do risco e o planejamento detalhado das atividades fazem com que seu rendimento seja muito melhor. Pudemos comprovar que o conhecimento do projeto (seu risco e planejamento) torna-se muito maior à medida que o tempo avança. Tivemos a oportunidade de "engrenar" com o projeto (seus entregáveis) apenas quando faltavam dois meses para o seu fim, o que foi um revés para o projeto.

Em contrapartida, mesmo com todas as dificuldades verificadas, estamos bastante satisfeitos com nossos resultados, que foram fruto de bastante dedicação e sacrifício. Ainda que não tenhamos em mãos um simulador completo, acreditamos que encerramos as disciplinas de Projeto de Formatura com um conhecimento muito mais valioso: a consciência da necessidade de metodologia de projeto no trato de problemas complexos e a certeza de que tudo que estava em nosso alcance foi realizado.

REFERÊNCIAS DE INFORMAÇÃO

IEEE Std 802.15.4-2003, IEEE Standard for Information technology—
Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific
requirements—Part 15.4: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications for
Low-Rate Wireless Personal Area Networks (LR-WPANS)

Design News – "Move Over, Bluetooth; ZigBee is here"- acessado em:
http://www.designnews.com/article/CA387448.html?industryid=22214 - Outubro de 2004

RF Design – "Zigbee: Another wireless standard in the pipeline " - acessado em:
http://rfdesign.com/mag/radio_zigbee_wireless_standard/ - Outubro de 2004

e-principles – "ZigBee: Plugging the Gap in Bluetooth and WiFi" – acessado em:
http://www.e-principles.com/zigbee_-_plugging_the_gap_in_bluetooth_and_wifi.htm -
Outubro de 2004.

DEITEL, H.M.; DEITEL, P.J. Java: Como Programar. 4ª ed. Porto Alegre: Bookman, 2003

ZIGBEE ALLIANCE. Documentos diversos. Disponível em:
http://www.zigbee.org