Motivação para Utilização de IP em Redes de Sensores sem Fios

Paulo A. C. S. Neves
Instituto de Telecomunicações,
Universidade da Beira Interior, Covilhã, Portugal
Instituto Politécnico de Castelo Branco, Castelo Branco, Portugal
e-mail: pneves@co.it.pt

Joel J. P. C. Rodrigues
Instituto de Telecomunicações,
Universidade da Beira Interior, Portugal
e-mail: joeljr@ieee.org

***Resumo*—As redes de sensores sem fios podem ser constituídas por milhares de pequenos sensores inteligentes com capacidades de comunicação sem fios. Estes pequenos nós comunicam entre si para fazer chegar a informação sensorial ao nó *sink*, um nó mais potente da rede. A necessidade de interligação das redes de sensores sem fios com a Internet foi rapidamente percebida pelos investigadores que procuraram soluções baseadas em conversão de protocolos entre Internet Protocol (IP) e os protocolos específicos. Recentemente, alguns investigadores não satisfeitos com as soluções existentes e a ideia que o protocolo IP não era apropriado para este tipo de redes, começaram a desenvolver pilhas protocolares TCP/IP para o nó sensor. Este artigo fornece uma visão sobre a aplicação do protocolo IP sobre redes de sensores sem fios, nomeadamente uma identificação dos desafios desta abordagem, uma revisão da literatura na introdução da pilha protocolar TCP/IP no nó sensor e informação sobre recursos disponíveis para criar uma rede de sensores sem fios com a pilha protocolar TCP/IP integrada no nó sensor.**

***Palavras chave*—Redes de Sensores sem Fios, IP sobre redes de sensores, computação ubíqua, hardware para redes de sensores sem fios**

I. Introdução

As Redes de Sensores Sem Fios (RSSF) nasceram nas aplicações militares, um início comum a algumas das tecnologias que hoje conhecemos. Estas redes são compostas por um número potencialmente elevado de nós sensores inteligentes, com um núcleo de processamento, memória para guardar o programa a executar, memória de execução, um ou mais sensores, tipicamente do tipo MEMS - *Micro Electro Mechanical Sensors*, uma unidade de comunicação sem fios e uma unidade de alimentação por bateria [1].

Um dos maiores desafios deste tipo de redes é a capacidade limitada de energia nos nós, dependentes de uma bateria. Em diversos cenários esta bateria é impossível de carregar ou substituir (por exemplo em cenários de ambientes hostis), o que leva à inutilização do nó quando a bateria estiver descarregada. Uma vez inutilizados alguns nós da rede, esta pode deixar de fornecer serviços. A estrutura básica de um nó sensor é apresentada na Figura 1.

Os dados sensoriais produzidos pelos nós sensores devem ser entregues a um outro tipo de nó da rede, tipicamente existente em muito menor número: o nó *sink* ou estação base. Este nó é tipicamente encarado como mais potente em termos de recursos computacionais e energia, sendo possível ter um ou mais por cada RSSF. O nó *sink* recebe a informação, eventualmente efectua algum processamento e apresenta-a ao utilizador e/ou disponibiliza-a para outro sistema.

As aplicações das RSSF estenderam-se para áreas de utilização civil, nomeadamente monitorização ambiental, monitorização de edifícios, aplicações industriais, saúde (através de e. g. redes de sensores corporais sem fios) e espaços inteligentes, entre outras [2]. O objectivo principal de uma rede de sensores sem fios é fazer chegar os dados sensoriais ao nó *sink*. No entanto as RSSF podem ter milhares de nós e eventualmente devido a limitações de cobertura e minimização de consumo, nem todos os nós têm capacidade de comunicar directamente com o *sink*. Para minimizar este problema, redes em malha e/ou encaminhamento hierárquico é muitas vezes utilizado.

Durante anos a investigação centrou-se na criação de protocolos específicos de encaminhamento e transporte para as RSSF, nomeadamente com considerações de consumo de energia. Os protocolos de encaminhamento dedicados podem ser divididos em três categorias fundamentais: *flat*, hierárquicos e baseados na localização [3]. Em relação aos protocolos de transporte podem ser divididos em protocolos orientados ao controlo da congestão (*congestion control*) ou protocolos orientados à garantia de fiabilidade (*reliability guarantee*), com estes últimos a poderem ser subdivididos em *upstream* e *downstream* [4].

A visão da computação de Mark Weiser em 1991 exigia tecnologias indisponíveis na altura, a visão da computação invisível para o utilizador [5]. A computação extrapola o computador de secretária - a visão de computação ubíqua -

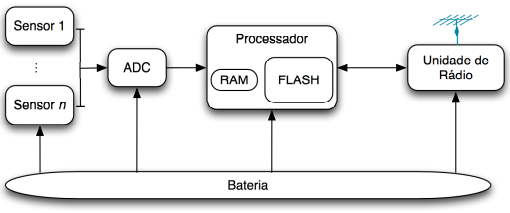

Fig. 1. Arquitectura básica de um nó sensor (adaptado de [1]).

onde os sistemas computacionais interagem com o utilizador e não o utilizador que interage com o computador.

A computação ubíqua fornece uma visão aumentada da realidade ao utilizador, tomando decisões em seu benefício, mas também precisa de dados para serem trabalhados. Esses dados podem ser a disponibilidade de lugares de estacionamento no local para onde o utilizador se dirige, ou até o seu estado de saúde que pode estar a ser enviado para análise por uma equipa médica [6]. Para as RSSF potenciarem a computação ubíqua é necessária a sua ligação à Internet, fornecendo serviços sensoriais à escala global [7].

A pilha protocolar TCP/IP é utilizada na Internet, forçando as RSSF a possuir mecanismos que permitam a interligação. Essa interligação pode ser efectuada de duas formas básicas: ou através de conversão de protocolos numa *gateway*, por exemplo no nó *sink*; ou através da integração da própria pilha protocolar TCP/IP no nó sensor. A primeira abordagem foi durante anos a preferida, muito devido aos mitos criados à volta do TCP/IP, tais como *overhead* no cabeçalho, considerações de energia e a possível lacuna de esquemas de endereçamento globais devido ao potencialmente grande número de nós suportados e as limitações do protocolo IPv4.

No entanto a necessidade de interligação entre as RSSF e a Internet foi rapidamente apreendida, uma vez que sem ligação à Internet as RSSF ficam muito limitadas, tornando-se uma abordagem isolada e limitada basicamente a uso local. No entanto a abordagem de IP integrado no nó sensor apresenta alguns desafios que são identificados neste artigo, bem como algumas das estratégias para os resolver.

O interesse deste artigo está focado em abordagens onde a pilha protocolar IP é integrada no nó sensor, aliviando o *sink* da parte de conversão de protocolos e permitindo uma integração na Internet facilitada, nomeadamente com a utilização de IPv6. Hoje é possível em poucos dias criar uma RSSF com IPv6 com base num sistema operativo de código aberto, por exemplo TinyOS [8] e ContikiOS [9], em hardware apropriado.

O artigo continua na secção II com a visão arquitectural da integração das RSSF na Internet e revisão da literatura. Na secção III apresentamos alguns recursos para criação de soluções, nomeadamente software e hardware. A secção IV apresenta uma introdução ao desenvolvimento de soluções com o sistema operativo ContikiOS e a plataforma de hardware TelosB da Crossbow [10] e finalmente a secção V conclui o artigo.

II. Visão arquitectural e revisão da literatura

Nesta secção apresenta-se as duas principais abordagens de interligação das RSSF com a Internet: a abordagem baseada em proxy e a abordagem baseada na colocação da pilha protocolar TCP/IP no nó sensor. Também se apresenta uma revisão da literatura.

Devido às vantagens do IPv6 em relação ao IPv4, nomeadamente o espaço de endereçamento, auto-configuração de endereço IP (*stateless address autoconfiguration*), entre outras, as implementações actuais privilegiam o IPv6. Com a especificação 6LoWPAN, que visa a introdução de uma camada de adaptação entre IEEE 802.15.4 e IPv6, reforça ainda mais a ideia de que as novas implementações devem ser efectuadas sobre IPv6.

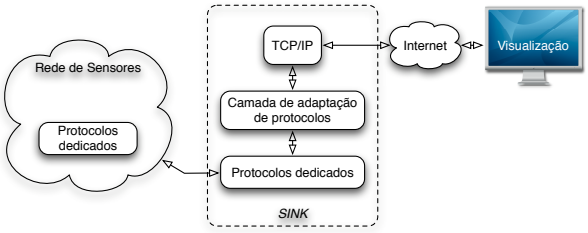

Fig. 2. Arquitectura de interligação baseada em Proxy.

A. Visão arquitectural

Existem basicamente duas abordagens para interligar as RSSF e a Internet: uma abordagem baseada em proxy, onde o nó *sink* poderá ser usado para conversão de protocolos e a abordagem de colocar a pilha protocolar TCP/IP directamente no nó sensor. A primeira abordagem ainda hoje subsiste como única alternativa para a utilização de protocolos dedicados não-IP nas RSSF, nomeadamente a ZigBee Bridge e ZigBee Gateway [11].

A Figura 2 apresenta a abordagem baseada em proxy. Esta arquitectura não introduz o IP no nó sensor, mas apenas no nó que se irá ligar à Internet, deixando para este nó o ónus de converter o protocolo IP em protocolos dedicados na RSSF.

Este tipo de arquitectura tem como vantagem a utilização de protocolos dedicados, permitindo a utilização de uma rede já implementada sem alterações. No entanto não permite a ligação individual de um nó sensor à Internet, o que poderá ser adequado a algumas implementações, por exemplo em redes sem nó sink (*sinkless*).

A Figura 3 ilustra a abordagem baseada na introdução de uma pilha protocolar TCP/IP no nó sensor, permitindo a troca de informação na rede, bem como integração na Internet. Dois cenários podem ser aqui equacionados, um em que o sensor comunica directamente com a Internet, outro em que o nó *sink* pode filtrar o tráfego da Internet, funcionando como *router*, apenas deixando passar o tráfego que se destina aos nós da rede de sensores. Desta forma o nó *sink* recebe dados de configuração, gestão e pedidos, enviando informação ao nós a partir da Internet quando apropriado. Um mecanismo de *cache* pode também ser implementado no nó *sink*, minimizando o tráfego dentro da RSSF, quando apropriado.

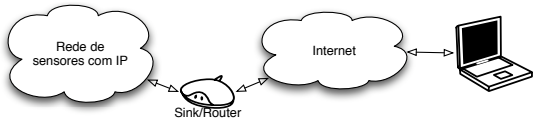

Fig. 3. Arquitectura de interligação baseada em pilha protocolar no nó sensor.

B. Revisão da literatura

Desde a primeira implementação de uma pilha protocolar de TCP/IP para micro-processadores de 8 e 16 bits, até à implementação e colocação em funcionamento de redes reais, passando por alguns protótipos, pode-se constatar que o protocolo IP tem vindo a percorrer um longo caminho nas RSSF. No entanto ainda hoje muitos investigadores negam-se a aceitar a sua utilização em redes de sensores.

Adam Dunkels em 2003 apresentou uma implementação da pilha protocolar TCP/IP para dispositivos integrados com muito baixo poder de processamento e capacidade de memória, nomeadamente para micro-processadores a 8 bits denominada uIP (microIP) e para micro-processadores de 16 bits (denominada lwIP) [12]. Ambas as implementações estão conformes o RFC 1122 e contêm implementações de IP, TCP e ICMP. A vertente mais exigente em termos de processamento e memória, a lwIP, que permite a criação de mais do que uma interface de rede e apresenta suporte para UDP. Esta implementação conseguiu desmistificar a ideia que o TCP/IP é demasiado complexo para plataformas de muito baixos recursos.

Uma utilização do uIP, mas já com o sistema operativo ContikiOS, surge através de uma rede de sensores para monitorização de intrusões, construída com hardware ESB da FU Berlin [13]. Para a configuração do endereço IP de cada nó os autores utilizaram um esquema baseado na localização espacial dos nós, através da atribuição de uma coordenada. É utilizada uma rede de sobreposição (*overlay network*) com um conjunto de nós principais a formar um *backbone* e um conjunto de nós sensor que enviam alarmes para o *backbone*. A rede *backbone* replica e transporta os eventos entre os seus nós e faz uso do NTTP - News Transfer Protocol para conectividade externa.

As redes de sensores corporais também podem beneficiar da integração de IP, nomeadamente através de um exemplo desenvolvido nos laboratórios da HP (HP labs) [14]. Nestas redes os sensores são colocados no ou próximo do corpo da pessoa a monitorizar, uma solução inovadora como exemplo de aplicação das RSSF. A rede corporal utiliza nós com uma pilha protocolar TCP/IP para monitorizar parâmetros de saúde do corpo humano de uma forma não intrusiva para a pessoa. Os investigadores dos laboratórios HP desenvolveram biosensores para capturar dados de movimento, embora a abordagem possa ser generalizada. Os nós utilizam o sistema operativo aberto TinyOS com uIP sobre IEEE 802.15.4, com uma quantidade de memória flash para suporte a operação desconectada.

A especificação 6LoWPAN [15] visa a criação de uma camada de adaptação entre IPv6 e IEEE 802.15.4, permitindo o transporte de IPv6 em tramas IEEE 802.15.4 com suporte a compressão de cabeçalho, suporte a endereçamento local e codificação de campos do cabeçalho para IPv6 e UDP com suporte a compressão parcial quando aplicável. Existem actualmente diversas implementações desta especificação, nomeadamente para os recursos de software identificados na secção III.

Uma arquitectura de rede de sensores com IP denominada IPSense, com capacidade de endereçamento flexível e suporte a mobilidade é apresentada em [16]. IPSense explora a agregação de nós através da utilização de um esquema hierárquico com "cluster heads" denominadas de Sensor Routers que efectuam a gestão da comunicação até ao nó *sink*. Os Sensor Routers são encarados como *gateways* entre a rede de sensores e outras redes e beneficiam de uma plataforma de hardware mais poderosa quando comparada com a dos nós sensores.

O grupo de trabalho da Internet Engineering Task Force (IETF) 6LoWPAN tem como objectivo o desenvolvimento da especificação de uma camada intermédia que permita mapear IPv6 sobre IEEE 802.15.4. Em termos de tramas, dois desafios foram identificados como principais: compressão do cabeçalho e a limitação de 102 bytes (máximo) para os dados numa trama IEEE 802.15.4. Para o primeiro o grupo de trabalho apresenta um esquema de compressão (denominado HC1, o HC01 está em estado de *draft*) e para o segundo um mecanismo de fragmentação adequado. Desta forma é possível transmitir um pacote IPv6 em uma ou mais tramas IEEE 802.15.4 e maximizar a capacidade de transmissão através de compressão do cabeçalho.

O uIPv6 é a evolução do uIP para IPv6 [17]. Esta camada de software potencia a criação de aplicações em ContikiOS com uma camada TCP/IPv6 e permite mapear os pacotes IP sobre IEEE 802.15.4 (utilizando 6LoWPAN), IEEE 802.11 e Ethernet. A camada TCP/IP suporta TCP, UDP, endereçamento IPv6, ICMPv6 e Neighbor Discovery (ND), utilizando apenas 2KB de RAM e 11,5KB de flash. Para uma solução completa sobre a plataforma de hardware Atmel Raven o código tem cerca de 35KB, que inclui os drivers para a plataforma, IEEE 802.15.4, 6LoWPAN com suporte a compressão de cabeçalho e fragmentação, uIPv6 e o ContikiOS. A camada de transporte UDP adiciona 1,3KB, enquanto que a TCP adiciona 4KB de código.

III. Recursos de hardware e software

Esta secção apresenta de forma relativamente resumida os recursos de software e hardware para implementar uma rede de sensores sem fios com suporte a IP. Do software destacam-se os dois principais sistemas operativos de código aberto o TinyOS [8] e o ContikiOS [9], [18]. Em termos de hardware destacam-se as propostas da Crossbow [10].

O sistema operativo TinyOS é desenvolvido e mantido pela Universidade de Berkeley e suporta a esmagadora maioria de plataformas de sensores sem fios da Crossbow, tais como a TelosB, IRIS, Mica (versões z e 2), Imote2, entre outras. O TinyOS, neste momento na sua versão 2.1 beneficia de uma base de contribuintes para a plataforma muito extensa, o que é comprovado pela dinâmica das respectivas listas de discussão. Neste momento também está disponível uma implementação 6LoWPAN, embora o ContikiOS tenha sido pioneiro neste campo. A implementação suporta auto-configuração, encaminhamento multi-salto e fragmentação para um MTU máximo de 1280 bytes. Recursos como o

ping, nc6 e tracert6 estão também disponíveis para efectuar depuração de código.

A linguagem de programação do TinyOS é o nesC, uma linguagem com sintaxe próxima do C, mas com algumas diferenças notórias, nomeadamente no modelo de programação, impondo uma certa curva de aprendizagem ao programador conhecedor de C. A página na Web disponibiliza toda a informação para descarregar o software, bem como documentação através de uma wiki e um livro de Philip Lewis "TinyOS programming Manual" e tutoriais entre outros recursos como por exemplo aceder às listas de discussão.

O ContikiOS é desenvolvido e mantido por um núcleo de investigadores liderados por Adam Dunkels do Instituto Sueco de ciência computacional (SICS - Swedish Institute of Computer Science), criador do uIP, lwIP e co-criador do uIPv6. Neste momento este sistema operativo também já tem uma panóplia interessante de plataformas de hardware que suporta, nomeadamente ESB da FU Berlin (a plataforma inicial), Tmote Sky (com a respectiva evolução, o TelosB), Atmel AVR Raven, Sentilla, entre outros. O ContikiOS recebeu o selo de prata IPv6 Ready e o código é completamente aberto, tal como o TinyOS.

O ContikiOS está actualmente na sua versão 2.3 e apesar de a base de contribuintes não ser tão extensa, é também uma lista com elevada dinâmica. O ContikiOS ao contrário do TinyOS utiliza C como linguagem de programação, o que pode facilitar a sua adopção face ao TinyOS. Sobressai no seu suporte a IP, devido à vasta experiência mesmo antes da criação da especificação 6LoWPAN. Esta plataforma de desenvolvimento apresenta também um sitio na Web onde é possível descarregar o software, aceder a tutoriais, notícias recentes sobre o software e não só, acesso a documentação, acesso a uma máquina virtual já completamente configurada com UBUNTU 8.04 LTS que pode ser utilizada sobre VMware, VirtualBox e Parallels, entre outros recursos como por exemplo inscrição nas listas de discussão.

Em termos de hardware, a esmagadora maioria das soluções são baseadas em duas plataformas de processador: uma da Texas Instruments através da família MSP430 e outra da Atmel, a ATmega. Em termos de unidade de rádio o standard IEEE 802.15.4 domina, com uma velocidade típica de 250Kbps, com a utilização do chip controlador CC2420, também conhecido por Chipcon CC2420. Todas as plataformas possuem RAM e flash ROM para execução de programas, enquanto que apenas algumas possuem memória flash adicional para outros fins, por exemplo para armazenamento de valores dos sensores.

A Crossbow constrói e vende uma panóplia de plataformas de hardware (também conhecidos como motes): IRIS, Micaz, Mica2, Imote2, TelosB e Cricket. Outros motes podem ser obtidos através da Scatterweb (spinoff da FU Berlin) [19], coalsenses iSense [20], Atmel AVR Raven [21]. Uma solução específica para redes de sensores corporais, a plataforma Shimmer [22], tem semelhanças em termos de recursos de hardware com a plataforma TelosB embora seja bastante mais pequena e tenha uma interface Bluetooth e uma interface para cartão MicroSD até 2GB. A Tabela I mostra um resumo de características de algumas dessas plataformas, nomeadamente o processador utilizado, a quantidade de memória, a unidade de rádio e os sistemas operativos que a suportam.

Da análise da Tabela I pode-se observar que a quantidade de memória típica de um mote oscila entre os 4KB de RAM no MICA2, MICAz e Cricket até cerca de 10KB (TelosB). Como seria de esperar todos os motes apresentam memória para programa que oscila entre os 48KB e os 128KB tipicamente. Uma forte excepção é a plataforma Imote2 que apresenta um processador de sistema móvel da arquitectura PXA, com 32MB de RAM e 32MB de flash o que o torna um excelente candidato a implementar um nó *sink* por exemplo. É também a mais cara, afastando-a de uma implementação sensível ao custo, sendo necessário adquirir à parte a placa de interface para programação (IIB2400) e a placa de sensores (ITS400). Neste caso os custos da plataforma sobem $150 e $249 respectivamente.

Em termos de energia a Crossbow optou por colocar uma unidade para 2 pilhas AA na maioria dos motes, sendo o Imote2 uma excepção ao utilizar uma unidade de 3 pilhas AAA. O TelosB que permite ligação directa por USB tem a capacidade de utilizar a corrente do barramento para alimentação. Já o Shimmer utiliza uma unidade de Li-Ion recarregável por USB de 230mAh bastante compacta, contribuindo para o seu baixo peso e

Alguns motes têm um módulo de memória flash dedicada para armazenamento de dados de sensores, permitindo a operação desconectada sem sacrificar a memória de programa disponível. Neste aspecto a plataforma Shimmer apresenta uma interface para 2GB de armazenamento através de interface para cartão MicroSD. Esta solução claramente melhora a conectividade, uma vez que o cartão pode ser extraído e com a utilização de um adaptador ser lido num computador pessoal, guardando os dados para análise posterior.

Outro exemplo de hardware para aplicações compatíveis com a especificação 6LoWPAN são as plataformas da Jennic [23]. Disponível em processadores integrados, nós de redes de sensores sem fios, e até um kit de avaliação 6LoWPAN (embora a um preço de 999USD). O fabricante disponibiliza uma SDK - *Software Development Kit* - com suporte para soluções baseadas em 6LoWPAN.

Apesar dos esforços do utilizador Jan Floras (http://www.nflora.dk/studie/) em conseguir o *porting* de Jennic para TinyOS, aparentemente a própria empresa não está interessada em deixar que o seu hardware seja utilizado com outro software. Esta situação pode limitar o desenvolvimento com esta plataforma, estando o utilizador limitado em termos não só de hardware, mas possivelmente também na escolha da plataforma de software.

IV. ContikiOS e TelosB

Esta secção fornece informação de início para o desenvolvimento de soluções baseadas em TelosB com o sistema operativo ContikiOS, numa perspectiva de tutorial. A forma mais rápida e fácil de começar o desenvolvimento

Tabela I
CARACTERÍSTICAS DE ALGUMAS PLATAFORMAS DE HARDWARE (MOTES)

Mote	Processador	RAM/programa/medição	Radio	Sistema Operativo	Preço (USD)
Crossbow IRIS	Atmel ATmega 1281	8KB/128KB/512KB	Atmel RF230	TinyOS	115/75
Crossbow MICAz	Atmel ATmega 128L	4KB/128KB/512KB	TI CC1000	TinyOS	99/75
Crossbow MICA2	Atmel ATmega 128L	4KB/128KB/512KB	TI CC1000	TinyOS/nano-RK	115-125/75
Crossbow Imote2	Marvel PXA271	32MB/32MB/-	TI CC2420 (IEEE 802.15.4)	TinyOS	299/150
Crossbow TelosB	TI MSP430	10KB/48KB/1MB	TI CC2420 (IEEE 802.15.4)	TinyOS/ContikiOS	99-139
Atmel AVR Raven	Atmel ATmega 1284P+3290P		Atmel RF320	ContikiOS	85 (kit)
Scatternode	TI MSP430	5KB/55KB/-	ISM band/19,2Kbps	Proprietary/ContikiOS	
Shimmer	TI MSP430	10KB/48KB/2GB	TI CC2420 (IEEE 802.15.4)	TinyOS 1.x	220
Sun SPOT	ARM920T	512KB/4MB/-	IEEE 802.15.4 radio	Java Programming	299 (edu. kit)

se o utilizador possuir uma máquina Windows ou Apple é utilizar a máquina virtual UBUNTU Linux, no entanto é preciso ter em atenção ao tamanho próximo dos 1GB. Esta máquina virtual funciona bem com VMware Workstation (Windows) e Fusion (Mac OS X), bem como Parallels Desktop (MAX OS X). Alguns utilizadores reportam sucesso com a utilização do VirtualBox, uma solução sem licença paga da Sun Microsystems [24]. Através do sistema CVS da Sourceforge - *Concurrent Versioning System* - é possível actualizar o ContikiOS para a última versão mesmo dentro da máquina virtual.

A. Informação preliminar sobre ContikiOS

O ContikiOS utiliza a linguagem C para programação e fornece uma API extensa para várias plataformas. A directoria raiz do sistema operativo, normalmente "contiki-2.x", possui as sub-directorias apps, core, cpu, doc, examples, platform e tools. A directoria apps possui diversas "aplicações" desenvolvidas para as diferentes plataformas que fornecem funcionalidades tais como um servidor Web, um ambiente tipo shell por USB, servidor FTP, entre outras.

A directoria core tem o núcleo do sistema operativo, enquanto que a directoria cpu contém código específico à unidade central de processamento do mote, compiladores e *linkers*. A directoria doc tem ficheiros de documentação criados pelo sistema Doxygen, enquanto que a directoria tools tem ferramentas que ajudam no desenvolvimento de software. A directoria examples contém exemplos, enquanto que a directoria platform tem ficheiros específicos para as diversas plataformas de hardware.

Se necessário a documentação global numa forma adequada para visualização num navegador de Web pode ser consultada através de http://www.sics.se/ ˜adam/contiki/docs/. É de um dos recursos mais importantes durante o desenvolvimento de aplicações. Para teclar código pode ser utilizado um editor de texto normal, embora o kate em Linux seja uma das propostas mais interessantes.

O ContikiOS fornece um núcleo orientado ao evento, com eventos de sistema e eventos que podem ser criados pelo programador. O sistema de desenvolvimento ContikiOS permite a execução de programas em diversas plataformas sem necessidade de alteração das aplicações em si, suporte a multi-tarefa optimizado, suporte a eventos temporizados, e suporte a um sistema de ficheiros, entre outros recursos muito úteis durante a programação.

O sistema de compilação com Makefiles está no cerne do ContikiOS. Não é necessário nenhum conhecimento especial para começar a utilizar os exemplos mais simples, embora um pequeno tutorial ensine a testar o famoso "hello world". As makefiles são utilizadas através do comando "make", o ContikiOS precisa de uma makefile em cada directoria de trabalho (projecto/exemplo). A makefile mais simples é a do "hello world", que pode ser a base para a criação de novos projectos. A makefile da directoria de trabalho deve invocar sempre a makefile principal do ContikiOS: a Makefile.include. A variável de ambiente CONTIKI aponta para a directoria raiz do ContikiOS e é usada de forma extensiva pelas makefiles.

B. Utilização com o mote TelosB/Sky

O ContikiOS possui suporte para a plataforma Sky, em tudo semelhante à TelosB (sua sucessora). O TelosB foi desenvolvido e apresentado pela Universidade de Berkley na Califórnia, USA. Na versão 2.3 do ContikiOS foi adicionado suporte multi-salto sobre IPv6, tornando esta plataforma não só atraente em termos de custos, como também em termos de funcionalidades no ContikiOS.

Em termos de recursos de depuração, esta plataforma beneficia do porto USB, permitindo ao programador utilizar este porto para receber informações sobre a execução do programa. Quando tal não é possível, podem-se usar os 3 LEDs (azul, vermelho e verde) para codificar estados do programa em execução por exemplo. Em termos de entradas, o TelosB tem um botão de pressão para *reset* (início da execução do programa armazenado e limpeza da RAM) e um que pode ser utilizado para outro propósito, de forma livre, denominado de *user*.

A programação do TelosB em ContikiOS é relativamente fácil, atendendo à panóplia de exemplos existentes, desde um pequeno servidor web IPv6 (webserver-ipv6) até um exemplo com dois ou mais TelosB a comunicar por UDP sobre IPv6, utilizando a implementação sicslowpan da especificação 6LoWPAN sobre IEEE 802.15.4. O ContikiOS possui um mecanismo de múltiplas tarefas (threads), denominado PT_THREADS, que permite execução de diversos fluxos de código em concorrência virtual, com a vantagem de não utilizar uma pilha em memória para cada tarefa, mas apenas um única pilha. Esta vantagem tem no entanto

como consequência a perda entre chamadas das variáveis armazenadas na pilha, excepto se forem declaradas como estáticas.

O sistema de ficheiros Coffee do ContikiOS permite a utilização da memória flash relativamente generosa de 1MB para armazenar ficheiros de diversos tipos para acesso a partir do programa. O sistema de ficheiros é flexível, no entanto a plataforma (até onde os autores a conhecem) não permite o acesso directo a partir de USB a esta memória flash, estando o seu barramento de interligação com o processador partilhado com a unidade de rádio CC2420.

V. Conclusões

Este artigo apresentou uma visão dos desafios, principais contribuições, recursos e informação sobre o desenvolvimento de programas com ContikiOS sobre TelosB. Com esta abordagem esperamos fomentar a utilização de IPv6 em redes de sensores, pois os recursos estão acessíveis e a parte de software é gratuita e acessível para programadores de sistemas integrados. Com estes dados um utilizador pode começar a experimentar a pilha protocolar TCP/IP em RSSF hoje mesmo, nomeadamente com suporte para 6LoWPAN.

A demanda para a utilização de IPv6 está longe do fim. Ainda é exigido da parte do programador um esforço considerável para colocar soluções a funcionar. Em conjunto com o carácter *application-specific* das RSSF torna a generalidade muito complicada, o que indica que a tecnologia ainda não está pronta para o *prime time*. No entanto tudo está em movimento para conseguir esse desejo, o de potenciar a computação ubíqua e passar a ter não só computadores e dispositivos móveis na Internet, mas também minúsculos e baratos dispositivos que não precisam publicar uma página Web, mas fornecem dados fundamentais para tornar o nosso dia-a-dia mais profícuo.

As RSSF farão parte da Internet do Futuro, nomeadamente para fornecer serviços sensoriais de forma global e potenciar a computação ubíqua. Da monitorização ambiental a aplicações mais centradas no utilizador, como o exemplo do parque de estacionamento, existem um grande número de aplicações que podem beneficiar da integração das RSSF na Internet. A visão de Internet dos computadores é legada e deve ser alargada não apenas aos dispositivos móveis, mas também a redes que fornecem dados fundamentais para a nossa vida diária.

Agradecimentos

Parte deste trabalho foi suportado pelo Instituto de Telecomunicações, grupo NetGNA - *Next Generation Networks and Applications*, Portugal.

Referências

[1] I. F. Akyldiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks (Elsevier)*, vol. 38, no. 4, pp. 393–422, 2002.

[2] P. Baronti, P. Pillai, V. W. C. Chook, S. Chessa, A. Gotta, and Y. F. Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15.4 and zigbee standards," *Comput. Commun.*, vol. 30, no. 7, pp. 1655–1695, 2007.

[3] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, 2004.

[4] C. Wang, K. Sohraby, B. Li, M. Daneshmand, and Y. Hu, "A survey of transport protocols for wireless sensor networks," *IEEE Network*, vol. 20, no. 3, pp. 34–40, 2006.

[5] M. Weiser, "The computer for the 21st century," *Scientific American*, vol. 265, no. 3, pp. 94–104, 1991.

[6] S.-L. Chen, H.-Y. Lee, C.-A. Chen, C.-C. Lin, and C.-H. Luo, "A wireless body sensor network system for healthcare monitoring application," Nov. 2007, pp. 243–246.

[7] J. Stankovic, "When sensor and actuator cover the world," *ETRI Journal*, vol. 30, no. 5, pp. 627–633, 2008.

[8] "Tinyos homepage," http://www.tinyos.net, May 2009.

[9] "ContikiOS homepage," http://www.sics.se/contiki, May 2009.

[10] "Crossbow Technology homepage," http://www.xbow.com, Feb. 2008.

[11] M. Sveda and R. Trchailik, "Zigbee-to-internet interconnection architectures," 2007, pp. 30–35.

[12] A. Dunkels, "Full TCP/IP for 8 Bit Architectures," in *Proceedings of the First ACM/Usenix International Conference on Mobile Systems, Applications and Services (MobiSys 2003)*, San Francisco, May 2003. [Online]. Available: http://www.sics.se/ adam/mobisys2003.pdf

[13] A. Dunkels, T. Voigt, N. Bergman, and M. Jönsson, "The design and implementation of an ip-based sensor network for intrusion monitoring," in *Intrusion Monitoring, Swedish National Computer Networking Workshop*, 2004.

[14] A. Christian and J. Healey, "Gathering motion data using featherweight sensors and tcp/ip over 802.15.4," HP Labs, Tech. Rep.

[15] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of ipv6 packets over ieee 802.15.4 networks," RFC 4944 (Proposed Standard), Internet Engineering Task Force, Sep. 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4944.txt

[16] T. Camilo, J. S. Silva, and F. Boavida, "Some notes and proposals on the use of ip-based approaches in wireless sensor networks," *Ubiquitous Computing and Communication Journal*, vol. Special Issue on Ubiquitous Sensor Networks, pp. 627–633, 2007.

[17] M. Durvy, J. Abeillé, P. Wetterwald, C. O'Flynn, B. Leverett, E. Gnoske, M. Vidales, G. Mulligan, N. Tsiftes, N. Finne, and A. Dunkels, "Making sensor networks ipv6 ready," in *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems*. New York, NY, USA: ACM, 2008, pp. 421–422.

[18] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *LCN '04: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 455–462.

[19] "Scatterweb homepage," http//www.scatterweb.com, Jan. 2009.

[20] "coalesenses homepage," http://www.coalesenses.de/, Jan. 2009.

[21] "Atmel ATAVRRZRAVEN homepage," http://www.atmel.com/dyn/Products/tools_card.asp?tool_id=4291, 2008.

[22] "Shimmer Research homepage," http://shimmer-research.com, Oct. 2008.

[23] "Jennic homepage," http://www.jennic.com, Sep. 2009.

[24] "VirtualBox homepage," http://www.virtualbox.org, May 2009.