Um estudo de caso do desenvolvimento de um produto utilizando a plataforma EPOS

João G. Zeni and Arliones Hoeller Jr.

[1]Laboratório de Integração Software/Hardware
Universidade Federal de Santa Catarina
Florianopolis – SC – Brasil {jgzeni,arliones}@lisha.ufsc.br

Abstract. *This paper presents a case study of the development of software and hardware of a bracelet for fall detection using the free platform called* EPOS. EPOS *is a free platform (software + hardware + tools) which includes operating system and hardware for embedded systems. In this case study we go from a specification of application requirements to the rapid prototiping of the system using the free hardware modules* EPOSMOTE *and* TSG*, to the final product.*

Resumo. *Este artigo apresenta um estudo de caso de desenvolvimento do software e hardware para uma pulseira para detecção de quedas utilizando a plataforma livre* EPOS. *O* EPOS *é uma plataforma (software + hardware + ferramentas) livre que inclui sistema operacional e hardware para sistemas embarcados. Neste estudo de caso, partimos de uma especificação de requisitos da aplicação, à prototipagem rápida utilizando os hardware livres* EPOSMOTE *e* TSG*, e à confecção do produto final.*

1. Introdução

O projeto se inicio a partir da necessidade apresentada de um dispositivo capaz de detectar quedas de seres humanos. Sabendo disso foi escolhido para o desenvolvimento desse produto a plataforma EPOS visto que a plataforma oferece suporte tanto a software quanto a hardware e é livre.

Este texto apresenta ao leitor a plataforma e relata o processo de desenvolvimento das aplicações necessárias ao produto e o redesenho do hardware feito para que esse apresenta-se as características funcionais e físicas requisitadas pelo produto.

2. A plataforma livre EPOS

O EPOS (Embedded Parallel Operating System) [**?**] é um arcabouço baseado em componentes para a geração de ambientes dedicados de suporte de tempo de execução. O arcabouço do EPOS permite que programadores desenvolvam aplicações independentes de plataforma, e ferramentas de análise permitem que componentes sejam adaptados automaticamente para atender a requisitos destas aplicações particulares. Por definição, uma instância do sistema agrega todo suporte necessário para a sua aplicação dedicada, e nada mais.

O projeto modular do EPOS foi guiado pela metodologia Projeto de Sistemas Embarcados Guiado pela Aplicação (ADESD - Application-Driven Embedded System Design) [**?**]. A ADESD baseia-se nas conhecidas estratégias de decomposição de domínio por trás do Projeto Baseado em Famílias (FBD - Family-Based Design) e Orientação a

Objetos (OO) como, por exemplo, análise de atributos comuns e variabilidade, para adicionar o conceito de identificação e separação de aspectos ainda nos estágios iniciais do projeto. Deste modo, a ADESD guia a engenharia de domínio para famílias de componentes, das quais dependências de cenários de execução são fatoradas na forma de aspectos e relacionamentos externos são capturados em um arcabouço de componentes. Esta estratégia de engenharia de domínio trata consistentemente algumas das questões mais relevantes do desenvolvimento de software baseado em componentes: reuso, gerenciamento da complexidade e composição.

Aplicações de redes de sensores sem-fio apresentam requisitos específicos que vão além dos atendidos pelos serviços tradicionais de sistemas operacionais. Estes incluem gerenciamento de energia eficiente, reprogramação em campo, abstração uniforme de sensores e serviços de comunicação configuráveis. O EPOS foi estendido de modo a atender estes requisitos extras [**?**].

O EPOS provê serviços de gerência de energia dirigida pela aplicação que permite um consumo consciente de energia em sistemas profundamente embarcados sem comprometer a portabilidade da aplicação e sem gerar sobrecustos excessivos. O objetivo do subsistema de gerenciamento de energia do EPOS é permitir que aplicações expressem quando determinados componentes de software não estão sendo utilizados, permitindo que o sistema migre dispositivos (hardware) associados a estes componentes de software para modos de operação que consumam menos energia. Disto emergiram várias questões que dizem respeito a diferenças arquiteturais entre dispositivos distintos e ao acesso concorrente aos recursos de hardware por diferentes componentes de software. Para tratar destas questões, foram concebidas (1) uma interface genérica para gerenciamento de energia, (2) um mecanismo de propagação de mensagens e (3) um modelo de formalização das trocas entre modos de operação [**?**].

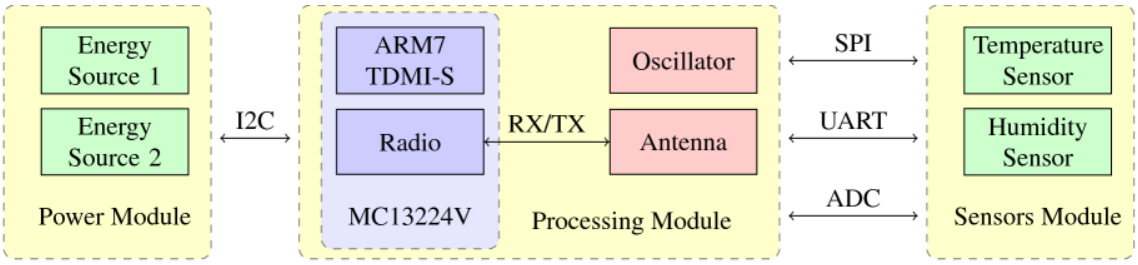
A infraestrutura de comunicação do EPOS para redes de sensores sem-fio é implementada pelo protocolo C-MAC (Configurable MAC) que provê suporte a comunicação de baixo nível [**?**], protocolos de roteamento como o ADHOP (???) [**?**], e uma pilha leve TCP/IP.

2.1. EPOSMOTE

O projeto EPOSMOTE tem por objetivo desenvolver uma família de módulos de sensoriamento que permita ampla configurabilidade tanto da plataforma quanto do ambiente de software (sistema operacional). O EPOSMOTE, apresentado na Figura 4, foi concebido com um projeto modular, sendo previstos três módulos, entre os quais foram estabelecidas interfaces padrão, permitindo o uso intercambiável de diferentes versões dos módulos. A Figura 5 apresenta os três tipos de módulos, que são os seguintes:

- **Módulo de Base:** o módulo base incorpora as funcionalidades de processamento e de comunicação. O projeto EPOSMOTE desenvolveu duas versões deste módulo, uma utilizando o single-package ZigBitTM e outra utilizando o SoC (single-die) MC13224V, da Freescale. Características específicas de cada versão do módulo base são apresentadas abaixo. Este módulo deve implementar detalhes de do suporte a estes dispositivos, como a regulação de tensão e o dimensionamento da antena, além rotear os pinos dos dispositivos de modo a manter o padrão das interfaces de alimentação e de entrada e saída.

- **Módulo de Entrada e Saída:** no módulo de entrada e saída devem ser implementadas as interfaces necessárias de entrada e saída, podendo um novo módulo destes ser desenvolvido para cada aplicação que se pretende desenvolver, permitindo o emprego dos sensores ou atuadores desejados para uma aplicação específica. O projeto EPOSMOTE desenvolveu um módulo de entrada e saída ao qual deu o nome de start-up board. Esta placa incorpora uma interface USB, sensor de temperatura, acelerômetro de 3 eixos, alguns LEDs e botões [**?**].
- **Módulo de Alimentação:** de modo a permitir o emprego de diferentes fontes de alimentação, uma interface de alimentação foi implementada. Módulos que se conectam a esta interface podem ser tão simples como uma bateria alcalina AA, ou tão complexas quanto um sistema com bateria de lítio recarregável ou com painéis solares. A interface de alimentação ainda disponibiliza uma interface I2C, permitindo que o módulo de alimentação se comunique com o de processamento. O projeto EPOSMOTE ainda não desenvolveu nenhum módulo de alimentação específico, mas trabalhos em andamento estão explorando tecnologias de captação de energia utilizando esta interface.



EPOS Mote II block diagram

Figura 1. EPOSMOTE block diagram

As interfaces padrão definidas pelo projeto EPOSMOTE para interconexão dos módulos desenvolvidos são as seguintes:

- **Interface de Entrada e Saída:** 32 pinos, sendo 2 para alimentação e outros 30 que podem ser utilizados ou como GPIO, ou com funções especificas que inclui ADC, UART e SPI.
- **Interface de Alimentação:** disponibiliza pinos aos quais devem ser conectados o terra e alimentação do modulo de alimentação. O modulo de processamento devolve ao modulo de alimentação o sinal com tensão regulada. Também existem os 2 pinos empregados na comunicação I^2C.

3. A aplicação de detecção de queda

A Figura 2 apresenta uma visão geral da aplicação proposta. Em um determinado local, diversas pessoas (usuários) utilizam pulseiras capazes de detectar uma eventual queda. Ao detectar uma determinada queda, a pulseira deve emitir um sinal sonoro e aguardar alguns segundos (e.g., 5 segundos) para que o usuário, eventualmente, sinalize, pressionando um botão, a ocorrência de um falso positivo, ou seja, de que a detecção foi indevida. Não ocorrendo a sinalização no tempo esperado, a pulseira deve emitir um sinal (pacote de dados) que deve chegar até um gateway capaz de remeter esta informação a um servidor, onde um sistema de monitoramento se encarregaria de providenciar atendimento

adequado à vítima da queda. Neste trabalho, focaremos no fluxo da informação sobre a queda da pulseira até o sistema de monitoramento. Não aboradamos, ainda, integração com serviços de emergência ou de comunicação extra para providenciar a prestação de socorro.

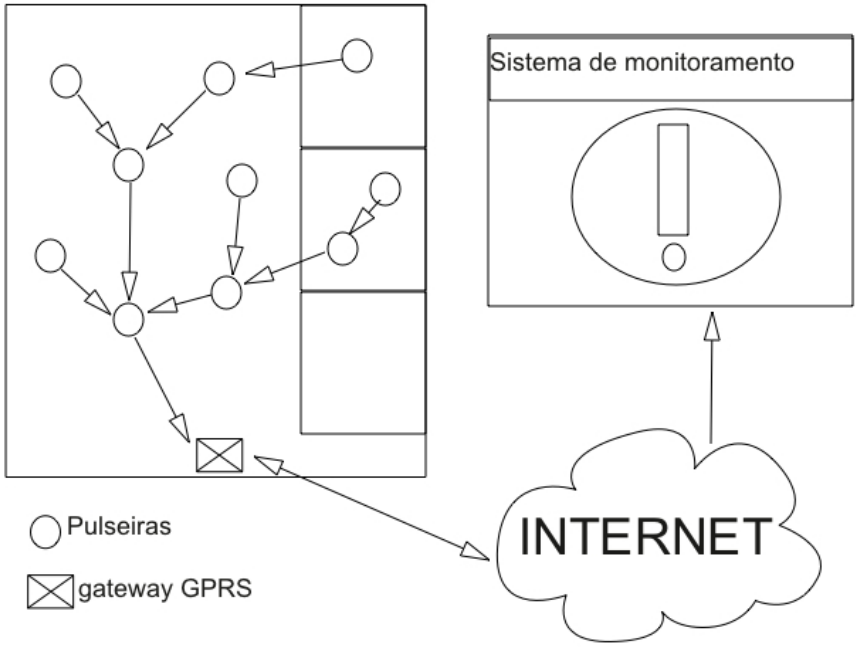


Figura 2. Planta baixa mostrando o funcionamento da aplicação

Os requisitos da aplicação são apresentados na Tabela 1. Neste ponto ressalta-se a importância do uso de uma plataforma completa. Muitos dos requisitos da aplicação são resolvidos praticamente de imediato com a seleção do hardware a ser utilizado e com a configuração adequada do sistema operacional.

O hardware de sensoriamento precisava possuir um tamanho reduzido e um formato que pudesse ser usado por uma pessoa, optou-se por uma pulseira, alem dos requisitos de forma precisava da funçao de monitor de queda, a qual foi realizada usando um acelerômetro de três eixos, uma interface com o usuário que consistia de um botão e de um LED que seriam ativado em caso de falso positivo, e por final a capacidade de enviar os dados sem fio a outros dispositivos.

O hardware de recepção precisava ser capas de receber o sinal de queda enviado e enviar esse sinal para a internet de modo a ser visualizado pela pessoa responável por monitorar os dispositivos.

Com base nas informações supra citadas inicio-se o projeto utilizando o hardware do EPOSMOTE para o desenvolvimento das aplicações de software, após a validação do software em cima do EPOSMOTE houve a necessidade do desenho de um hardware reduzido visto que o hardware de EPOSMOTE possui características que não eram úteis ao projeto e e demasiado grande para a finalidade desejada do projeto, sendo assim se projetou um hardware derivado do antigo de modo a não ser necessário refazer o software ja validado e para que esse hardware atende-se as necessidades de projeto tanto de tamanho

quanto de funcionalidade.

4. Desenvolvimento do Software

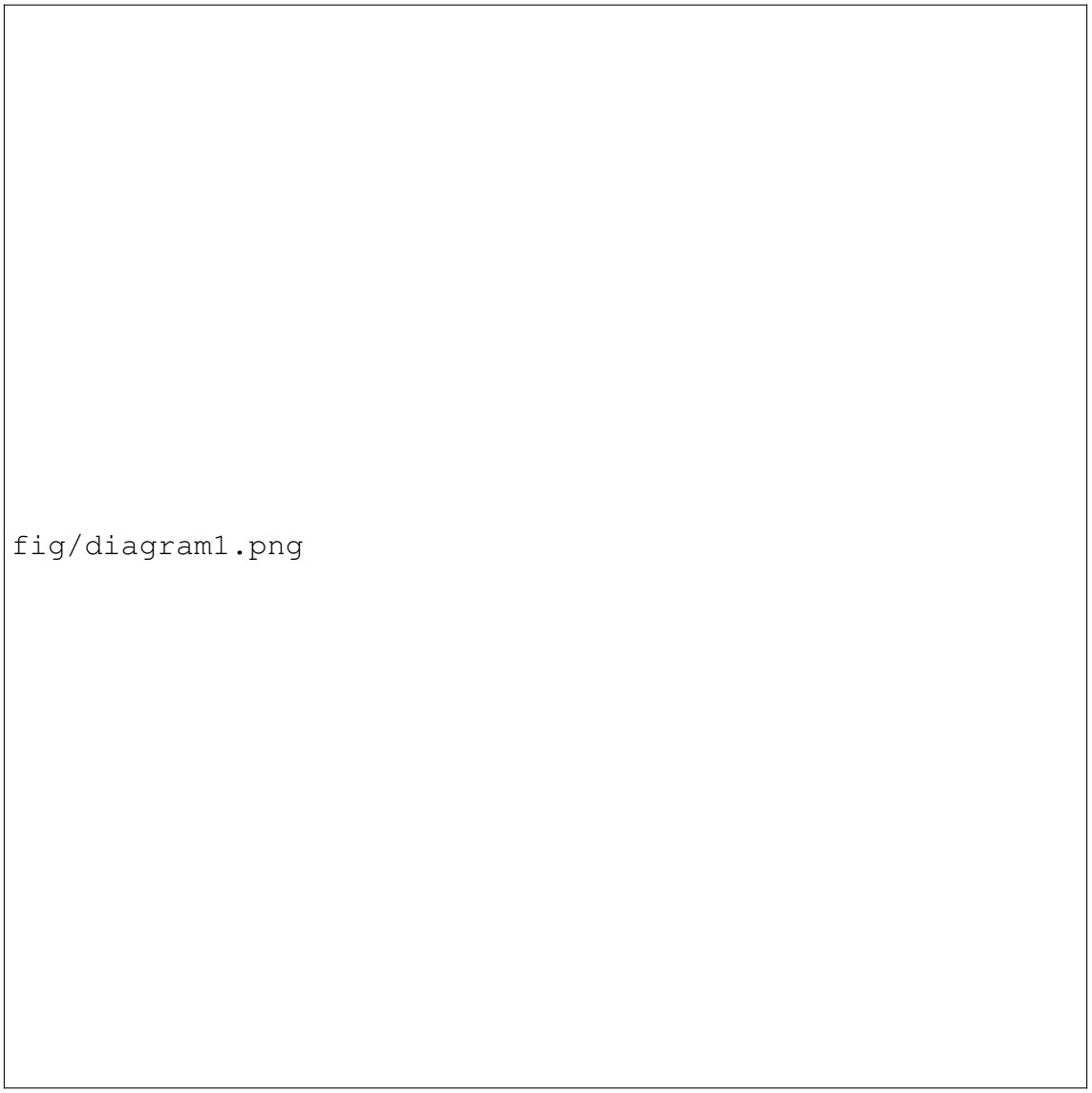


Figura 3. Maquina de estados da aplicação de detcção de quedas

O projeto de software do sistema foi feito com base na plataforma escolida, o EPOS, com isso se mostrou necessária a utilização de dois hardwares distintos, um para a pulseira e outro para gateway, e consequentemente notou-se também a necessidade de duas aplicações distintas, uma de detecção de quedas e envio de sinal e outra de gateway.

A aplicação de detecção de quedas e envio de sinal consiste de um algoritmo de detecção de quedas, de um sistema de comunicação com o usuário e de um sistema de envio de sinal para o gateway. O algoritmo de detecção de quedas foi implementado com base no manual do acelerômetro que contem uma aplicação que detecta quedas. A comunicação com o usuário foi implementado usando um botão e um LED de forma que quando a queda é detectada o LED é acesso e permanece assim por um determinado

período de tempo ou o até botão ser pressionado, caso o botão não seja pressionado o sinal de queda é enviado ao gateway. O sistema de envio de sinal usado foi o NIC já implementado no EPOS, enviando uma string que indica queda e informa o id da pulseira que a detectou.

Para o gateway enviar o sinal para internet foi utilizado um gateway GPRS que é capaz de pegar dados de uma porta serial e envia-los a um servidor. Sendo assim a aplicação do gateway foi feita de forma a enviar um sinal alertando uma queda e o id da pulseira que a detectou em caso de recebimento de sinal de uma das pulseiras.

Foi criado uma aplicação usando Django para mostrar os dados em uma pagina na internet. Essa aplicação abre uma porta do servidor em que são enviados os dados pelo gateway GPRS e lê a saída dessa porta, caso essa porta receba um sinal alertando uma queda a aplicação imprime na tela uma mensagem de queda e o id da pulseira que a detectou.

O projeto de software pode ser valido por inteiro usando o hardware padrão do EPOS sem que fosse necessário projetar e construir o hardware final do produto antes da validação da aplicação, o que se mostra eficiente uma vez que os projetos de hardware e software podem ser desenvolvidos em paralelo e o software pode ser valido antes de iniciar a produção do hardware.

Ao final do desenvolvimento do software os testes necessários para sua validação foram realizados no EPOSMOTE, visto que este apresenta as mesma características do hardware do produto. Isto nos economizou tempo, uma vez que não foi necessário esperar o hardware final ficar pronto para validarmos o software, e dinheiro, já que não foi preciso produzir um protótipo apenas visando testar o software.

5. Desenvolvimento do Hardware

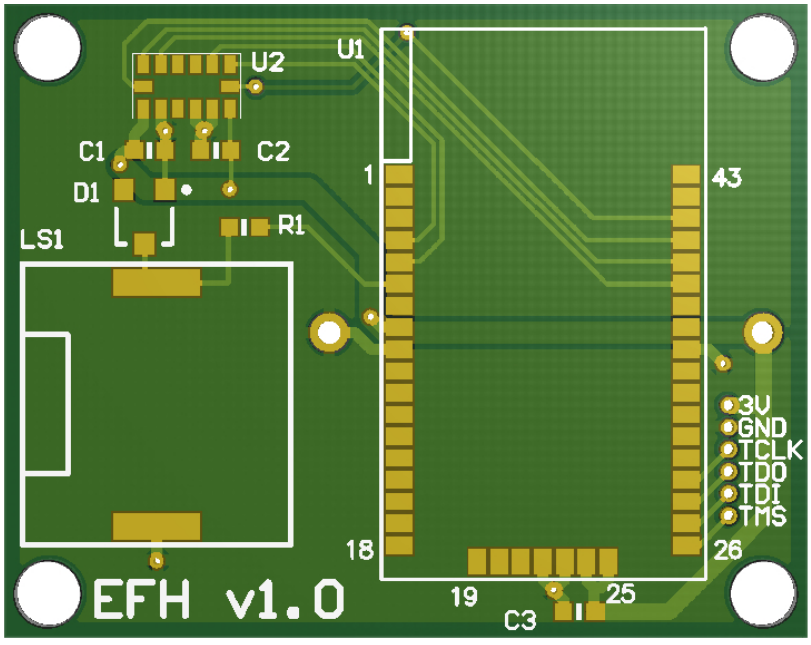


Figura 4. A PCB final do projeto

O EPOSMOTE apresenta mais funcionalidades do que as requisitadas pelo projeto, porém não apresenta as características físicas requisitadas pelo mesmo, sendo assim foi necessário um redesenho do hardware do EPOSMOTE que atende-se não apenas as funcionalidades como também apresenta-se tamanho e peso coerentes com o esperado do produto.

O hardware do EPOSMOTE foi redesenhado apresentando as mesmas características de sistemas do EPOSMOTE padrão, porém apresentando apenas os dispositivos externos necessários. Optou-se também por utilizar uma bateria CR2450 em função de seu tamanho reduzido e formato propício a ser utilizado em um hardware que necessita apresentar tamanho reduzido.

O processo de redesenho do EPOSMOTE foi simples uma vez que tem-se acesso ao desenho original do EPOSMOTE e esse apresenta todas as ligações entre processador e os dispositivos externos. Assim é apenas utilizar o desenho original retirando os dispositivos que não são utilizados e rearranjando as trilhas de modo que a placa fique a menor possível.

6. Conclusao

Este artigo apresenta uma plataforma de hardware/software livre, o epos, e como apartir da plataforma e dos requisitos de um projeto se desenvlveo um produto. O principal foco do artigo é no processo de desenvolvimento.

A partir desse texto é perceptivel não apenas a viabilidade do desenvolvimento de produtos com base em plataformas livre como tambem uma serie de vangentagem na utilização de plataformas livres para o desenvolvimento de produtos.

Referências

Requisitos funcionais		
ID	**Requisito**	**Descrição**
1	Detectar queda	Detectar queda do modo mais seguro possível, evitando geração de falsos positivos.
2	Comunicar queda	Progamar informação de ocorrência de uma queda.
3	Identificar origem da queda	Deverá ser possível identificar a pulseira (pessoa) que caiu, bem como sua localização aproximada.
4	Comunicar central	A informação da queda deve ser comunicada a uma central de controle remota conectada à Internet.
Requisitos não-funcionais		
ID	**Requisito**	**Descrição**
5	Dimensões	As dimensões devem ser reduzidas de modo a permitir o confortável do sistema como uma pulseira.
6	Energia	O consumo deve ser baixo de modo a permitir a maior durabilidade possível da bateria da pulseira.

Tabela 1. Tabela de requisitos da aplicação de detecção de quedas.