EFICIÊNCIA ENERGÉTICA PARA SOFTWARE IOT EM LARGA ESCALA

1. NARRATIVA PROFISSIONAL

Trabalho no campo de *Linguagens de Programação* com o foco em *Sistemas Reativos de Tempo Real*. Sistemas reativos interagem continuamente com o mundo externo através de sensores e atuadores (e.g., botões, LEDs e motores). Quando essas interações têm que cumprir prazos estritos, o sistema é dito de *tempo real*. A maioria das aplicações do dia-a-dia, tais como suítes de escritório e jogos eletrônicos são reativas mas bastante tolerante a atrasos. Em contraste, sistemas embarcados de tempo real, tais como aplicações médicas, sistemas de aviação, e dispositivos de IoT, devem cumprir prazos mais estritos.

No início do meu Mestrado, me juntei ao grupo de desenvolvimento do *Ginga*, o padrão de software do Sistema Digital da TV Brasileira [20], que também se tornou um padrão ITU/UN [15]. Durante esse período, trabalhei numa integração não intrusiva entre as duas linguagens padronizadas NCL e Lua através de uma interface de programação reativa. O resultado desse trabalho foi publicado no *ACM DocEng* de 2009. Também escrevi o capítulo sobre desenvolvimento em Lua e NCL do livro de referência do Ginga.

Em 2009, iniciei o programa de Doutorado com o objetivo de projetar uma nova linguagem reativa desde o início, agora direcionada a sistemas embarcados com recursos restritos. Por um lado, essas plataformas são tipicamente seis ordem de magnitude menos poderosas que computadores tradicionais em termos de memória e processamento. Por outro lado, aplicações embarcadas necessitam de mais confiabilidade para operar sem a intervenção humana por longos períodos de tempo. Além disso, um subconjunto significativo de sistemas embarcados é implantado em locais sem linhas de energia e devem funcionar com baterias. Sendo assim, uma linguagem de programação direcionada a sistemas embarcados restritos deve ser pequena, confiável e eficiente energeticamente. Ao final de 2012, me tornei bolsista da *SAAB Aerospace* para um período de seis meses de pesquisa. Juntei-me a um grupo de pesquisa em uma universidade da Suécia e reescrevemos drivers e protocolos de rede de qualidade industrial. Conseguimos uma redução considerável no tamanho do código fonte, com uso similar de recursos em comparação com C. Em 2013, publicamos um artigo com esses resultados no *ACM SenSys*.

Após me graduar, contribuí com o projeto *Cidades Inteligentes*, que envolve 20 universidade e busca construir uma infraestrutura de IoT. Em 2015, em cooperação com um aluno de Doutorado, publicamos um artigo no *ACM TOSN*, mostrando que podemos reprogramar dispositivos embarcados remotamente usando menos energia do que o estado da arte. Durante o mesmo período, publiquei um artigo no *Int'l Conference on Modularity* propondo uma nova abstração de concorrência que gera

programas mais modulares. Também fui convidado a apresenta minha linguagem de pesquisa em um encontro anual do *Working Group on Language Design* grupo afiliado com a IFIP da UNESCO.

Atualmente, como professor associado em uma universidade brasileira, continuo a investigar como o projeto de linguagens pode afetar o desenvolvimento de aplicações reativas. Durante os últimos 4 anos, publiquei ou atuei no comitê do *Int'l Workshop on Reactive Languages*. Em 2017, publiquei um artigo consolidando o projeto da minha linguagem de pesquisa no *ACM TECS*.

2. Divulgação Científica: O Papel do Software IoT na Eficiência Energética

De acordo com a Agência Internacional de Energia (IEA) [21], existiam em torno de 14 bilhões de dispositivos conectados tradicionais em 2013 (e.g., telefones TVs inteligentes). Esse número deve crescer para 50 bilhões em 2020 com a proliferação de dispositivos IoT (e.g., lâmpadas inteligentes e tecnologia vestível). Dispositivos IoT e tradicionais já superam o número de pessoas no planeta por um fator de dois, e o tráfego de dados é esperado crescer a uma taxa exponencial nos próximos anos. No entanto, a maior parte da energia desses aparelhos é consumida quando eles estão em *standby* (*modo em espera*). A emissão anual de CO_2 relacionada a standby é equivalente a de 1 milhão de carros. A projeção de crescimento de IoT, juntamente com o efeito surpreendente dos efeitos de consumo de standby, fizeram com que a eficiência de standby para dispositivos conectados fosse um dos seis pilares do *Plano de Ação para Eficiência Energética* do G20[1].

Outras organizações também reportaram sobre a importância da economia de energia em dispositivos conectados. Para o *Internet Engineering Task Force (IETF)*, "o gerenciamento energético está se tornando um requisito adicional para redes devido a diversos fatores que incluem o aumento dos custos de energia, o impacto ecológico para a operação das redes, e a regulação de energia" [26]. Para o *American Council for an Energy-Efficient Economy (ACEEE)*, "o potencial para a nova eficiência energética permanece enorme, (...) devemos considerar uma abordagem sistêmica para escalar a eficiência energética. (...) a eficiência inteligente é adaptativa, antecipatória, e conectada" [10].

Considerando iniciativas concretas, o grupo de trabalho *Electronic Devices and Networks* da IEA foca especificamente na questão do standby em dispositivos conectados[2]. A iniciativa da ACEEE em *Intelligent Efficiency* promove uma abordagem sistemática para otimizar o comportamento cooperativo de dispositivos de modo a buscar ganhos de energia como um todo. Ambos as abordagens, por dispositivo e sistêmica, envolvem soluções de software, dado que a economia de energia é uma política dinâmica que depende das demandas das aplicações e níveis de baterias em um determinados momentos. Também existem padrões de baixo consume de energia para a infraestrutura de IoT com diferentes demandas de alcance, velocidade e distribuição física [22]. Como exemplo, o *Bluetooth*

[1]G20's Energy Efficiency Action Plan: `https://www.iea-4e.org/projects/g20`

[2]EDNA initiative: `https://edna.iea-4e.org/`

Low-Energy (BLE) é um substituto para o padrão Bluetooth clássico e é projetado para baixas velocidades em redes pessoais (PANs). O *6LoWPAN* adapta o padrão IPv6 para baixo consumo e em dispositivos de processamento limitado. Essas tecnologias permitem transmissões mais eficientes, suportam topologias flexíveis e reduzem o tráfego consideravelmente. Elas também possibilitam o uso de modos de standby mínimos. No entanto, essas tecnologias exigem o uso de software para controlar os modos ativos e de standby de maneira a construir uma IoT eficiente em termos de energia.

3. Resumo Científico

O uso efetivo de standby terá papel fundamental na eficiência energética para os 50 bilhões de dispositivos IoT esperados até 2020 [21]. Este projeto de pesquisa visa endereçar os desafios de software, conforme determinados pela IEA, em direção a uma IoT eficiente em termos de energia [21]: garantir que os dispositivos adotem os níveis mais econômicos possíveis de standby, e que os dispositivos permaneçam o maior período possível em standby.

Tendo em vista a escala projetada para a IoT e o papel do modo standby para a eficiência energética, este projeto de pesquisa tem os seguintes objetivos:

(1) Endereçar a eficiência energética com o uso criterioso de standby.
(2) Focar em arquiteturas embarcadas restritas que formam a IoT.
(3) Prover mecanismos de standby no nível de linguagens de programação para escalar para todas as aplicações.
(4) Suportar mecanismos de standby transparentes e não intrusivos para reduzir as barreiras de adoção.

Muitas linguagens, extensões e sistemas operacionais direcionados à energia foram propostos recentemente. Algumas propostas ajustam a QoS (qualidade de serviço) para reduzir o consumo [27, 2, 23, 28]. Outras propostas oferecem mecanismos para trocar o comportamento dependendo das demandas e níveis de bateria da aplicação [25, 7, 5, 6]. Nenhuma dessas iniciativas tira vantagem de modos de standby quando ociosas, mas somente adaptam ou eliminam computações quando em modo ativo (não satisfazendo o objetivo 1).

Tenho trabalhado no projeto de uma nova linguagem de programação reativa para sistemas embarcados restritos pelos últimos 8 anos. A linguagem é baseada no modelo de concorrência síncrono, que troca poder por confiabilidade e possui um modelo de tempo mais simples que cobre a maioria dos requisitos de aplicações IoT. Nesse modelo, todas as reações ao mundo externo são computadas em tempo finito, garantindo que as aplicações sempre chegam a um estado ocioso que é suscetível ao modo standby. Esperamos que aplicações existentes que não sejam cientes sobre o consumo de energia irão beneficiar-se de economias na ordem de 50% baseado nas estimativas da IEA (considerando as tecnologias atuais de standby [21]) e também em trabalhos em ciência transparente de energia [18].

4. Projeto Científico

4.1. **Primeiro Ano.** Necessitaremos de um orçamento de R$70k: R$60k para suporte a três estudantes, R$5k para equipamentos, e R$5k para publicações e custos de viagem.

4.1.1. *Infraestrutura de Hardware para IoT.* Usaremos Arduinos [17] como a principal plataforma de hardware para IoT. A maioria dos Arduinos é baseado em microcontroladores de baixo consumo da Atmel, tais como o *ATmega328p* [1], suportando seis modos standby que podem reduzir o consumo para níveis baixíssimos.

No ambiente acadêmica, existe bastante pesquisa com o uso de Arduinos no contexto de IoT [11, 9, 19, 12]. A popularidade do Arduino fará com que a nossa pesquisa seja mais acessível e reproduzível para outros grupos. Em educação, muitos cursos em universidades usam o Arduino [4, 3, 24, 16]. Nós temos usado o Arduino em um curso de graduação pelos últimos 3 anos, o que permitirá avaliar os resultados com programadores de sistemas embarcados menos experientes. Na comunidade hobista, há uma abundância de software publicamente disponível que poderemos adaptar para a nossa linguagem e avaliar os ganhos de eficiência.

4.1.2. *Infraestrutura de Software para IoT.* A maneira mais típica em Arduino de interagir com o mundo externo é através de *polling*, que amostra um periférico externo para detectar mudanças de estado. Polling gasta ciclos de CPU e previne que o dispositivo entre em modo standby. No Arduino, mesmo funcionalidades básicas, tais como temporizadores, conversores A/D, e SPI, usam ciclos de polling que gastam energia em modo ativo.

De modo a prover standby automático, as aplicações deve ser inteiramente reativas a eventos. Recentemente, adicionamos suporte a rotinas de interrupção (ISRs) como um conceito primitivo na nossa linguagem, o que permitirá reconstruir a infraestrutura de software IoT com ciência ao modo standby desde sua base. Esse processo consistirá primordialmente de reescrever device drivers, que são os pedaços de software que interagem diretamente com o hardware. Essa abordagem não irá afetar a maneira como as aplicações são escritas em níveis mais abstratos, que permanecerá similar às aplicações em Arduino. No entanto, em vez de gastar ciclos da CPU em espera ativa, as aplicações entrarão no modo de standby mais profundo possível em quanto estiverem ociosas.

4.1.3. *Aplicações IoT.* De modo a avaliar os ganhos de energia com a infraestrutura proposta, precisaremos avaliar o consumo em aplicações realistas. A comunidade do Arduino tem uma abundância de projetos open-source que podem ser reescritos na nossa linguagem para tirar proveito do modo de standby transparente. Então, poderemos comparar as versões originais e reescritas em termos de consumo para tirar conclusões sobre a efetividade do modo de standby transparente. Os cenários mais realistas de IoT usam comunicação por rádio extensivamente. Nesse contexto, iremos avaliar desde protocolos ad-hoc simples até protocolos mais complexos com ciência energética para ver até

que extensão nossa proposta contribuiria efetivamente em economias de energia. No longo prazo, esperamos mostrar o valor para desenvolvedores reescreverem suas aplicações na nossa linguagem para tirar proveito dos modos de standby automáticos. Nessa direção, avaliaremos o tempo tomado para reescrever as aplicações e os ganhos reais de eficiência energética.

4.1.4. ***Contribuições Esperadas***. O vocabulário dedicado a eventos oferecido pela linguagem aumenta o nível de abstração dos programas para um nível mais próximo do domínio de IoT, provendo mais segurança e expressividade para programadores. Até onde sabemos, suporte para ISRs no nível de linguagem ainda não foi tentado anteriormente.

Nossa proposta visa fazer com que todas as aplicações estejam sujeitas a modos de standby transparentemente. Sendo parte da infraestrutura de software, somente device drivers necessitarão de gerenciamento explícito de energia, e todas as aplicações construídas sobre eles se beneficiarão de eficiência energética automaticamente.

Nossa linguagem é um projeto de 8 anos e tem uma implementação open-source madura que está disponível publicamente para downloads. Com a proposta de adaptação ao contexto de eficiência energética para IoT, a linguagem pode se tornar uma alternativa prática ao Arduino no curto prazo.

4.2. **Os Três Anos Seguintes.**

4.2.1. *Ciência de Energia Sistemática.* Ciência de energia sistemática considera a rede IoT como um todo, com dispositivos cooperando para maximizar a eficiência de energia globalmente. Nesse contexto, *computação adaptativa* [13] é a capacidade da IoT se adaptar dinamicamente ao consumo de energia, baseando-se na demanda das aplicações e níveis de bateria durante sua execução. Fundamentalmente, o objetivo ainda é o de que cada dispositivo maximize o tempo ocioso para ser mais suscetível a modos de standby. Dessa forma, a pesquisa do primeiro ano é um pré-requisito para essa fase. Nós continuaremos a investigar como mecanismos de linguagens podem propiciar mais eficiência energética, mas agora em um contexto de computação adaptativa.

O método científico para energia sistemática será similar ao adotado para a abordagem por dispositivo (conforme discutido na Seção 4.1.3): pegar protocolos e aplicações IoT cientes de energia e reescrevê-las usando mecanismos não intrusivos, e usar os mesmo critérios de avaliação (i.e., tempo de reescrita, estética do código, e consumo de energia).

4.2.2. *Arquiteturas e Aplicações de IoT Complexas.* A IoT também consiste de dispositivos mais tradicionais, tais como roteadores, servidores, e smartphones. Em 2016, existiam 3.9 bilhões de assinaturas telefônicas globalmente, e esse número de alcançar 6.8 bilhões em 2022 [14].

Smartphones têm restrições similares de consumo de bateria e também podem tirar proveito das técnicas propostas para sistemas embarcados restritos. De modo a transpor a barreira de dispositivos restritos para smartphones para a IoT, percorreremos um caminho similar ao apresentado na Seção 4.1:

Infraestrutura de Hardware: Usaremos o BeagleBone Black [8], que compartilha objetivos similares ao do Arduino, provendo uma plataforma barata e aberta que é adequada a aplicações mais ricas, tais como interfaces gráficas, multimídia, e jogos.

Infraestrutura de Software: De modo a garantir standby automático para aplicações, toda infraestrutura de software, principalmente device drivers, terá que ser recriada usando ISRs em nossa linguagem.

Applicações: Além de aplicações IoT, as aplicações de smartphone, tais como mensagens instantâneas e navegação Web, também podem maximizar a eficiência energética através do modo de standby. Iremos reescrever desde aplicações simples, tais como um relógio gráfico, até aplicações em rede mais complexas, tais como um navegador, para avaliar o consumo de energia.

4.2.3. ***Contribuições Esperadas***. Esperamos que a ciência de energia sistemática irá beneficiar-se dos mecanismos não intrusivos da nossa linguagem, podendo ser implementada em um nível de abstração maior, resultando em aplicações eficientes que requerem menos esforço de programação.

A transição de telefones simples para smartphones resultou na degradação do tempo de vida das baterias. No entanto, a maior parte do tempo, os smartphones estão ociosos nos nossos bolsos mas gastando energia. Esperamos aumentar consideravelmente a autonomia das baterias mantendo toda a funcionalidade de um smartphone moderno.

REFERENCES

[1] Atmel. *ATmega328P Datasheet*, 2011.

[2] W. Baek and T. M. Chilimbi. Green: A framework for supporting energy-conscious programming using controlled approximation. In *Proceedings of the 31st ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '10, pages 198–209, New York, NY, USA, 2010. ACM.

[3] J. D. Brock, R. F. Bruce, and S. L. Reiser. Using arduino for introductory programming courses. *J. Comput. Sci. Coll.*, 25(2):129–130, Dec. 2009.

[4] L. Buechley, M. Eisenberg, J. Catchen, and A. Crockett. The lilypad arduino: Using computational textiles to investigate engagement, aesthetics, and diversity in computer science education. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 423–432, New York, NY, USA, 2008. ACM.

[5] A. Canino. Gradual mode types for energy-aware programming. In *Companion Proceedings of the 2015 ACM SIGPLAN International Conference on Systems, Programming, Languages and Applications: Software for Humanity*, SPLASH Companion 2015, pages 79–80, New York, NY, USA, 2015. ACM.

[6] A. Canino and Y. D. Liu. Proactive and adaptive energy-aware programming with mixed typechecking. In *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2017, pages 217–232, New York, NY, USA, 2017. ACM.

[7] M. Cohen, H. S. Zhu, E. E. Senem, and Y. D. Liu. Energy types. In *Proceedings of the ACM International Conference on Object Oriented Programming Systems Languages and Applications*, OOPSLA '12, pages 831–850, New York, NY, USA, 2012. ACM.

[8] G. Coley. Beaglebone black system reference manual. Technical report, BeagleBoard.org, 2013.

[9] C. Doukas and I. Maglogiannis. Bringing iot and cloud computing towards pervasive healthcare. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pages 922–926. IEEE, 2012.

[10] R. N. Elliott, M. Molina, and D. Trombley. A defining framework for intelligent efficiency. Technical Report E125, American Council for an Energy-Efficient Economy (ACEEE), 2012.

[11] V. Georgitzikis, O. Akribopoulos, and I. Chatzigiannakis. Controlling physical objects via the internet using the arduino platform over 802.15. 4 networks. *IEEE Latin America Transactions*, 10(3):1686–1689, 2012.

[12] K. Gomez, R. Riggio, T. Rasheed, D. Miorandi, and F. Granelli. Energino: A hardware and software solution for energy consumption monitoring. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2012 10th International Symposium on*, pages 311–317. IEEE, 2012.

[13] J. Henkel and L. Bauer. What is adaptive computing? *SIGDA Newsl.*, 40(5):1–1, May 2010.

[14] N. Heuveldop. Ericsson mobility report. Technical report, Ericsson, AB, 2017.

[15] ITU-T Rec. H.761. Nested Context Language (NCL) and Ginga-NCL for IPTV Services. Technical report, ITU, Geneva, 2012.

[16] P. Jamieson. Arduino for teaching embedded systems. are computer scientists and engineering educators missing the boat? *Proc. FECS*, 289294, 2010.

[17] D. Kushner. The making of arduino. *IEEE Spectrum*, 26, 2011.

[18] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, et al. Tinyos: An operating system for sensor networks. *Ambient intelligence*, 35:115–148, 2005.

[19] K. Mandula, R. Parupalli, C. A. Murty, E. Magesh, and R. Lunagariya. Mobile based home automation using internet of things (iot). In *Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2015 International Conference on*, pages 340–343. IEEE, 2015.

[20] Norma ABNT NBR 15606-2:2007. Digital terrestrial television—Data coding and transmission specification for digital broadcasting—Part 2: Ginga-NCL for fixed and mobile receivers—XML application language for application coding. Technical report, ABNT, 2008.

[21] OECD/IEA. More data less energy—Making network standby more efficient in billions of connected devices. Technical report, International Energy Agency, 2014.

[22] G. Reiter. Wireless connectivity for the internet of things. Technical report, Texas Instruments, 2014.

[23] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman. Enerj: Approximate data types for safe and general low-power computation. *SIGPLAN Not.*, 46(6):164–174, June 2011.

[24] J. Sarik and I. Kymissis. Lab kits using the arduino prototyping platform. In *Frontiers in Education Conference (FIE), 2010 IEEE*, pages T3C–1. IEEE, 2010.

[25] J. Sorber, A. Kostadinov, M. Garber, M. Brennan, M. D. Corner, and E. D. Berger. Eon: A language and runtime system for perpetual systems. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, SenSys '07, pages 161–174, New York, NY, USA, 2007. ACM.

[26] E. Tychon, B. Schoening, M. Chandramouli, and B. Nordman. Energy management (EMAN) applicability statement. Technical report, IETF, 2011.

[27] H. Zeng, C. S. Ellis, A. R. Lebeck, and A. Vahdat. Ecosystem: Managing energy as a first class operating system resource. *SIGARCH Comput. Archit. News*, 30(5):123–132, Oct. 2002.

[28] Y. Zhu and V. J. Reddi. Greenweb: Language extensions for energy-efficient mobile web computing. In *Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '16, pages 145–160, New York, NY, USA, 2016. ACM.