ENERGY EFFICIENCY FOR IOT SOFTWARE IN THE LARGE

1. PROFESSIONAL NARRATIVE

I work in the field of *Programming Languages* with a focus on *Real-Time Reactive Systems*. Reactive systems interact continuously with the external world through sensors and actuators (e.g., buttons, LEDs, and motors). When these interactions have to comply with strict deadlines, the system is said to be *real time*. Most everyday applications, such as office suites, phone apps, and video games are reactive but fairly tolerant to delays. In contrast, embedded real-time systems, such as medical appliances, avionic systems, and IoT devices, must conform to stricter deadlines.

In the beginning of my Masters, I joined a group supporting *Ginga*, the software standard of the Brazilian Digital TV System [20], which also became an UN's ITU standard [15]. During this period, I worked on a non-intrusive integration of the two standardized languages NCL and Lua through a reactive programming interface. The outcome of this work was published in the *ACM DocEng* in 2009. I also wrote the chapter on developing with Lua and NCL in Ginga's main reference book.

In 2009, I started the PhD program with the goal of designing a new reactive language from scratch, now targeting resource-constrained embedded systems. On the one hand, these platforms are typically six orders of magnitude less powerful than off-the-shelf personal computers in terms of memory size and CPU speed. On the other hand, embedded applications need higher reliability to operate without human supervision for long periods of time. Furthermore, a significant subset of embedded systems is deployed in locations without power lines and need to run on batteries. Hence, a programming language targeting constrained embedded systems needs to be small, reliable, and energy efficient. At the end of 2012, I earned a scholarship from SAAB Aerospace for a six-month research period. I joined an University in Sweden and we rewrote industrial-grade network drivers and protocols using my research language and attested a considerable reduction in source code size with similar resource usage in comparison to C. In 2013, we published a paper with these results in the *ACM SenSys*.

After graduating, I contributed to the project *Cidades Inteligentes* (Smart Cities), which involves 20 Universities and aims to build an infrastructure for the IoT. In 2015, in cooperation with a PhD student, we published a paper the *ACM TOSN*, showing that we can reprogram embedded devices remotely using less energy than the state of the art. During the same period, I published a paper in the *Int'l Conference on Modularity* proposing a new concurrency abstraction that leads to more modular programs. I was also invited to present my research language in the annual meeting of the *Working Group on Language Design*, which is affiliated with UNESCO's IFIP.

Currently, as an associate professor at a Brazilian University, I continue to investigate how the design of languages can affect the development of reactive applications. For the last 4 years, I have have

been either publishing in or acting in the committee of the *Int'l Workshop on Reactive Languages*. In 2017, I published a consolidating paper on the design of my research language in the *ACM TECS*.

2. Science Outreach: The Role of IoT Software in Energy Efficiency

According to the International Energy Agency (IEA) [21], there were around 14 billion traditional network-connected devices in 2013 (e.g., mobile phones and smart TVs). This number is expected to increase to 50 billion by 2020 with the proliferation of IoT devices (e.g., smart bulbs and fitness wearables). IoT and traditional network-connected devices already outnumber people on the planet by a factor of two, and the amount of data traffic is expected to grow at an exponential rate in the next years. However, most of the energy to power these devices is consumed while they are in *standby mode*. The annual CO_2 emissions related to standby in Australia are equivalent to those of 1 million cars. The projected growth of IoT devices together with the surprising effects of standby consumption, made network standby efficiency one of the six pillars of G20's *Energy Efficiency Action Plan*[1].

Other organizations have also reported the importance of energy savings for networked devices. For the Internet Engineering Task Force (IETF), "energy management is becoming an additional requirement for networks due to several factors including the rising energy costs, the increased awareness of the ecological impact of operating networks and devices, and the regulation of energy" [26]. For the American Council for an Energy-Efficient Economy (ACEEE), "the potential for new energy efficiency remains enormous, (...) we must take a systems-based approach to scale up energy efficiency. (...) intelligent efficiency is adaptive, anticipatory, and networked" [10].

With regard to concrete initiatives, IEA's *Electronic Devices and Networks* focuses specifically on the issue of networked device standby[2]. ACEEE's *Intelligent Efficiency* promotes a systematic approach to optimize the behavior of cooperating devices in order to achieve energy savings as a whole. Both approaches, device and system based, involve mostly software solutions, since energy savings are dynamic policies that depend on the application demands and device battery levels at a given moment in time. There are also low-power standards for the IoT infrastructure that suit different needs of range, throughput, and physical distribution [22]. For instance, *Bluetooth Low-Energy (BLE)* is a replacement for classic Bluetooth and is designed for lower data throughput in personal area networks (PANs). *6LoWPAN* adapts the IPv6 internet standard to low-power and limited-processing devices. These technologies make radio transmissions more efficient, support flexible network topologies, and reduce data traffic considerably. They also enable sleep modes that cut energy consumption to minimum levels. However, these technologies ultimately require software to control their functionalities and to wisely switch between standby and active modes in order to build an energy-efficient IoT.

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

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

3. Scientific Summary

Effective use of low-power standby will play a fundamental role in energy efficiency for the expected 50 billion IoT devices by 2020 [21]. This research project aims to address the software challenges, as determined by IEA, towards an energy-efficient IoT [21]: to ensure that devices employ the lowest possible modes of standby, and that devices remain in longest possible periods of standby.

Given the projected scale of the IoT and the role of low-power standby towards energy efficiency, this research project has the following goals:

(1) Address energy efficiency through extensive use of standby.
(2) Target constrained embedded architectures that form the IoT.
(3) Provide standby mechanisms at the programming language level that scale to all applications.
(4) Support transparent/non-intrusive standby mechanisms that reduce barriers of adoption.

Many energy-aware languages, extensions, and operating systems have been proposed recently. Some proposals adjust QoS (quality of service) to reduce power consumption [27, 2, 23, 28]. Other proposals offer mechanisms to switch behaviors depending on application demands and battery levels [25, 7, 5, 6]. None of these initiatives take advantage of standby modes when idle, but only adapt or eliminate computations while in active modes (not addressing goal 1).

I have been working in the design of a new reactive programming language targeting resource-constrained embedded systems for the past 8 years. The language is grounded on the synchronous concurrency model, which trades power for reliability and has a simpler model of time that suits most requirements of IoT applications. In this model, all reactions to the external world are guaranteed to be computed in bounded time, ensuring that applications always reach an idle state amenable to standby mode. We expect that existing energy-unaware applications will benefit from savings in the order of 50% based on IEA's estimates considering current standby technologies [21], and previous third-party work on transparent energy awareness [18].

4. Scientific Project

4.1. **First Year.** We will require a budget of R$70k: R$60k to support three students, R$5k for equipment, and R$5k for publication and travel costs. The details are described in the long proposal.

4.1.1. *IoT Hardware Infrastructure.* We will use off-the-shelf Arduinos [17] as the main hardware IoT platform. Most Arduinos are based on low-cost microcontrollers from Atmel, such as the *ATmega328p* [1], supporting six sleeping modes that can reduce power consumption to very low levels.

In the academia, there is a lot of research using Arduino in the context of the IoT [11, 9, 19, 12]. The popularity of Arduino will make our own research more accessible and reproducible to other groups. In education, many University courses use Arduino [4, 3, 24, 16]. We have also been using

Arduino in an undergraduate course for the past 3 years, which will allow us to evaluate our results with less experienced embedded programmers. In the hobbyists community, there is an abundance of publicly available software which we can adapt to our language to evaluate energy efficiency gains.

4.1.2. *IoT Software Infrastructure.* The typical way to interact with the external world in Arduino is through polling, which samples an external device actively to detect changes on its state. Polling wastes CPU cycles and prevents the device to enter in standby. In Arduino, even basic functionality, such as timers, A/D converters, and SPI, uses active polling loops that waste energy in active mode.

In order to provide automatic standby, applications need to be entirely reactive to events. Recently, we have added support for *interrupt service routines (ISRs)* as a primitive concept of our language, which will allow us to rebuild the IoT software infrastructure with standby awareness from the ground up. This process will consist primarily of rewriting device drivers, which are the pieces of software that interface directly with the hardware. This approach will not affect how applications are written at higher levels, which will remain similar to Arduino applications. However, instead of wasting cycles in busy-wait loops, the applications will enter the deepest possible standby mode when idle.

4.1.3. *IoT Applications.* In order to evaluate the gains in energy efficiency with the proposed infrastructure, we will need to evaluate the consumption of realistic applications. The Arduino community has an abundance of open-source projects which can be rewritten in our language to take advantage of transparent standby modes. Then, we will be able to compare the original and rewritten versions in terms of energy consumption to draw conclusions about the effectiveness of transparent standby modes. The most realistic scenarios for the IoT use radio communication extensively. In this context, we will evaluate from simple handmade ad-hoc protocols to more complex energy-aware protocols and see to what extent our proposal would effectively contribute to energy savings. In the long term, we expect to make the case for developers to rewrite their applications in our language to take advantage of standby modes for free. Towards this direction, we will evaluate the time it takes to rewrite an application and the real gains in energy efficiency.

4.1.4. ***Expected Contributions***. The dedicated vocabulary of our language for events raises the programming abstraction to a level closer to the domain of IoT, providing more safety and expressiveness to programmers. As far as we know, ISR support at the language level has not been tried in the past.

Our proposal aims to make all applications subject to standby modes transparently. Being part of the software infrastructure, only device drivers will require explicit energy management, and all applications built on top of them will benefit from energy efficient automatically.

Our language is a 8-year project and has a mature open-source implementation that is publicly available. With the adaptation to the context of energy-efficient IoT, the language may become a practical alternative for Arduino in the short term.

4.2. **The Three Years to Follow.**

4.2.1. *Systematic Energy Awareness.* Systematic energy awareness considers the IoT network as a whole, with cooperating devices trying to maximize energy efficiency globally. In this context, *adaptive computing* [13] is the capability of the IoT to adapt energy consumption dynamically, based on application demands and levels of power supply during execution. Ultimately, the goal is still that each device maximizes its idle periods to be more susceptible to standby modes. Hence, the research of the first year is a prerequisite for this phase. We will continue to investigate how language mechanisms can aid energy efficiency, but now in the context of adaptive computing.

The research method for systematic energy awareness will be similar to the one adopted for device-based awareness (as discussed in Section 4.1.3): take existing energy-aware IoT protocols and applications for Arduino, rewrite them using non-intrusive mechanisms, and use the same evaluation criteria (i.e., time to rewrite, coding aesthetics, and energy consumption).

4.2.2. *Complex IoT Architectures and Applications.* The IoT also consists of more traditional networked devices, such as routers, servers, and smartphones. As of 2016, there were 3.9 billion smartphone subscriptions worldwide, and this number is expected to reach 6.8 billions by 2022 [14].

Smartphones have similar restrictions in battery consumption and could also take advantage of the techniques we propose for constrained embedded systems. In order to transpose the barrier from constrained devices to smartphones for the IoT, we will take a similar path as presented in Section 4.1:

Hardware Infrastructure: We will use the BeagleBone Black [8], which shares similar goals with Arduino, providing a cheap and open-source platform but which is suitable for rich applications, such as graphical user interfaces, multimedia, and games.

Software Infrastructure: In order to ensure automatic standby for applications, all software infrastructure, mostly device drivers, needs to be rebuilt from scratch using ISRs in our language.

Applications: In addition to IoT applications, typical smartphone applications, such as instant messaging and internet browsing, can also maximize energy efficiency through standby. We will rewrite from simple applications, such as a graphical clock, to more complex networked applications, such as a web browser, and evaluate their energy consumption.

4.2.3. ***Expected Contributions***. We expect that systematic energy awareness will take advantage of the non-intrusive mechanisms of our language to be implemented at a higher level of abstraction, resulting in energy-efficient applications that require less programming efforts.

The transition from simple mobile phones to smartphones resulted in a degrading effect on battery lives. Most of the time, however, smartphones are sitting in our pockets wasting energy. We expect to increase the battery autonomy considerably while keeping the functionality of a modern smartphone.

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.