Understanding the Needs of Novice Developers in Creating Self-Powered IoT

Chengshuo Xia
csxia@keio.jp
Guangzhou Institute of Technology, Xidian University
Guangzhou, China

Tian Min
welkinmin@keio.jp
Keio University
Yokohama, Japan

Daxing Zhang
zhangdx@xidian.edu.cn
Guangzhou Institute of Technology, Xidian University
Guangzhou, China

Congsi Wang
congsiwang@xidian.edu.cn
Guangzhou Institute of Technology, Xidian University
Guangzhou, China

ABSTRACT

The rise of the Internet of Things (IoT) has given birth to transformative and massively deployed computing applications that raise the significant issue of energy sources. It is impractical and irresponsible to rely on wires and batteries to power trillion-level devices. One promising prediction is that energy harvesting technologies will serve as alternative power sources for IoT devices. However, we might be losing this prophecy for lack of understanding of how novice developers comprehend energy in developing IoT. In response, we conducted a mentored physical prototyping study with a two-day workshop involving eight novice developers. The study consisted of qualitative and quantitative analyses, the artifacts, interviews with both novice developers and an expert, and implications of designs for future tools. The findings reveal informational gaps that demand educational efforts and assistive features to facilitate novice developers. We present major findings from the study and implications for the design of future tools.

CCS CONCEPTS

• **Human-centered computing** → **User studies**; **Interactive systems and tools**; **Empirical studies in HCI**.

KEYWORDS

Mentored Physical Prototyping, Novice Developers, Developer Supports, Sustainability, Battery-free computing, Sustainability, Energy Harvesting

ACM Reference Format:
Chengshuo Xia, Tian Min, Daxing Zhang, and Congsi Wang. 2024. Understanding the Needs of Novice Developers in Creating Self-Powered IoT. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24), May 11–16, 2024, Honolulu, HI, USA.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3613904.3642576

CHI '24, May 11–16, 2024, Honolulu, HI, USA

ACM ISBN 979-8-4007-0330-0/24/05
https://doi.org/10.1145/3613904.3642576

1 INTRODUCTION

Advances in electronics have promised to provide a bright future of distributed and embedded form factors realized by trillions of devices constituting the Internet of Things (IoT). Recent advances in energy harvesters have enabled a wide range of self-sustaining systems that do not need to be plugged in, preserving deployment flexibility, or require batteries, eliminating users' maintenance burden.

While energy from environments takes many forms (e.g., solar, wind) and is seemingly *infinite*, energy from developers is *finite* – it is considerably more exhausting for developers to adopt energy harvesting to power IoT systems than their dominant alternatives, namely, battery power [19]. With IoT systems that feature energy harvesting, energy availability can change over time, power failures may occur, and resources may be constrained. Although developers' expectations for energy harvesting are high, its performance in practice may be low. This "impedance mismatch" between developers and energy harvesters creates frustrations and drives people (especially novice IoT developers) away from adopting energy harvesting as a promising technology that could make IoT practical and sustainable.

Recent years have seen the recognition of self-powered computing with an emerging system technology that extends its performance and capability envelope. Previous research has demonstrated novel system technologies, such as battery-free cellphones [66], battery-free smart environment sensing [79], a battery-free Game Boy [10], and novel programming [21, 36, 38], runtime [3, 49], and architecture [9, 13, 20] level techniques, to provide reliable computation, sensing, and actuation. Despite the promise of these systems, little has been done to understand what information and feedback novice developers need as they build these systems. Unlike prior work which aims to investigate the usability of energy harvesting tools (e.g., Fliker [20], BFree [36], and Battery-Free MakerCode [39]), we set out to uncover *what novice developers need* with a general-purpose IoT platform featuring the state-of-the-art computing, sensing, actuation, communication, as well as energy harvesting technologies.

In this study, we target novice developers, as experienced electronic system developers tend to circumvent the difficulty we aim to expose: "expert blindness," which is when experienced developers are unable to perceive the difficulties encountered by novices when approaching a new domain. Therefore, studying novice developers

may be more informative. In addition, novice developers are more open to trying new technologies and approaches, and are likely to adopt them for future practices in their development. We consider that targeting novice IoT developers could be equally, if not more, effective than targeting experienced developers in the long run. Therefore, we believe that the contributions of this study can support developers who work on novice developer-oriented energy harvesting toolkits and can help produce a platform dedicated to energy harvesting similar to mainstream interactive electronic system development, such as that of Arduino [1].) We investigated novice developers' needs in a mentored physical prototyping study with a two-day workshop. The paradigm was derived from prior pair programming work that had been proven successful [72]. We paired novice developers with an expert who had extensive experience in energy harvesting and self-powered system development. The expert worked with the novice developers through a collaborative development process, addressed questions from the participants, and provided guidance when necessary. Each participant—expert team was required to finish two tasks featuring different application needs and energy sources. Semi-structured interviews were conducted after each task, aiming to probe where novice developers needed the most support when addressing the challenges.

Our findings can be categorized into two aspects, including *gaps between concept and practice* and *need for assistive features*, regarding the fundamental electronics practice lack and the energy harvesting technique dedicated assistance need. This insufficiency of knowledge constitutes many "unknown unknowns" (i.e., one might lack the required knowledge to be aware of critical information that one might be missing to accomplish certain tasks) reported by the expert. By revealing this gap and features, we call for action from computer science and electrical engineering educators and researchers to support tools for IoT development that would integrate the promise of energy harvesting techniques into practical and sustainable systems. The contributions of our study are as follows:

- A two-day mentored physical prototyping study involving eight novice developers and an expert that revealed insights into the needed support of novice developers when developing self-powered IoT.
- Design implications based on summarizing the existing platforms/tools and study findings and providing assistive features and caveats for future tools in developing self-powered IoT systems with energy harvesters.

2 RELATED WORK

2.1 Interactive Systems Powered by Energy Harvesters

One of the significant bottlenecks in IoT applications is the power consumption of a single device, which limits the life of the system [31]. In recent years, HCI researchers have begun to incorporate human behavior with energy generation through different energy harvesters to address energy-constrained issues [79]. Different energy sources are used depending on variations in environmental parameters, such as illumination, temperature, and mechanical energy. Among these energy sources, photoelectric conversion is the most popular in related designs due to the ubiquitous nature of light conditions [41, 46, 77]. For monitoring systems in the wild, thermometric generators can be adapted to power the computing sensor system continuously [24]. These generators can also be used on the human skin to reduce the burden of the body area sensor network's power [73]. Cooperating with human behavior, the kinetic energy generated from humans can also be used, such as device authentication and secret keys generation [42] and wearables for children [58]. Winkel et al. [10] introduced a platform utilizing solar and kinetic energy to power a game console. Intermittent computing was used to maintain the continuity of the game when the harvested energy was exhausted. Teng et al. [67] designed a kinetic energy harvesting-based system to enhance virtual reality haptics from the user's movement.

2.2 Technology Pillars to Improve the Reliability of Energy Harvesting

To improve reliability at the hardware level, mainstream solutions have proposed adding an energy storage unit as a universal solution to fluctuations in environmental parameters. Zhang et al. [78] designed a hysteresis circuit system for energy harvesting from microbial fuel cells (MFCs). Such a connection and operation approach has been extensively adopted in various energy harvesting-based systems [18, 19]. However, the energy storage unit usually has a low energy density and a high leakage current. This has led us to rethink the use of capacitors in energy harvesting systems. For example, Jackson et al. [28] reconsidered the use of batteries in IoT nodes due to the high performance of recent batteries, especially when combined with ultra-low power consumption sensors. There is no doubt that using connected supercapacitors or rechargeable batteries to buffer energy in energy harvesting circuits is widely accepted. For example, numerous commercial ICs are available for developers to create energy harvesting circuits using referenced schemes with an energy storage unit, such as Linear LTC3588 [43], TI BQ25570 [26], and charge pumps [25]. Energy harvesting IoT nodes normally operate periodically because of fluctuations in environmental energy. This introduces volatile data transmission into the system's characteristics. Many techniques have been proposed to optimize system operation to ensure that tasks can be correctly performed, such as a dedicated operating system [69] and cache structure [76]. The use of checking points enables switching the location of a variable storage in the sensing task program by nonvolatile memory to save important node information before running out of energy and completing the task after the next resumption of work [30, 48, 56]. For larger distributive networks, studying the energy-efficient protocol and coordinating method among the nodes can enhance working performance [1, 33, 65].

2.3 Integrated Systems to Facilitate Energy Harvesting Development

To facilitate energy harvesting uses, previous studies have focused on developing integrated plug-and-play energy harvesting systems to assist developers in adopting energy harvesters in both hardware and software. EnOcean [12] released self-powered sensor systems

[1] https://www.arduino.cc/

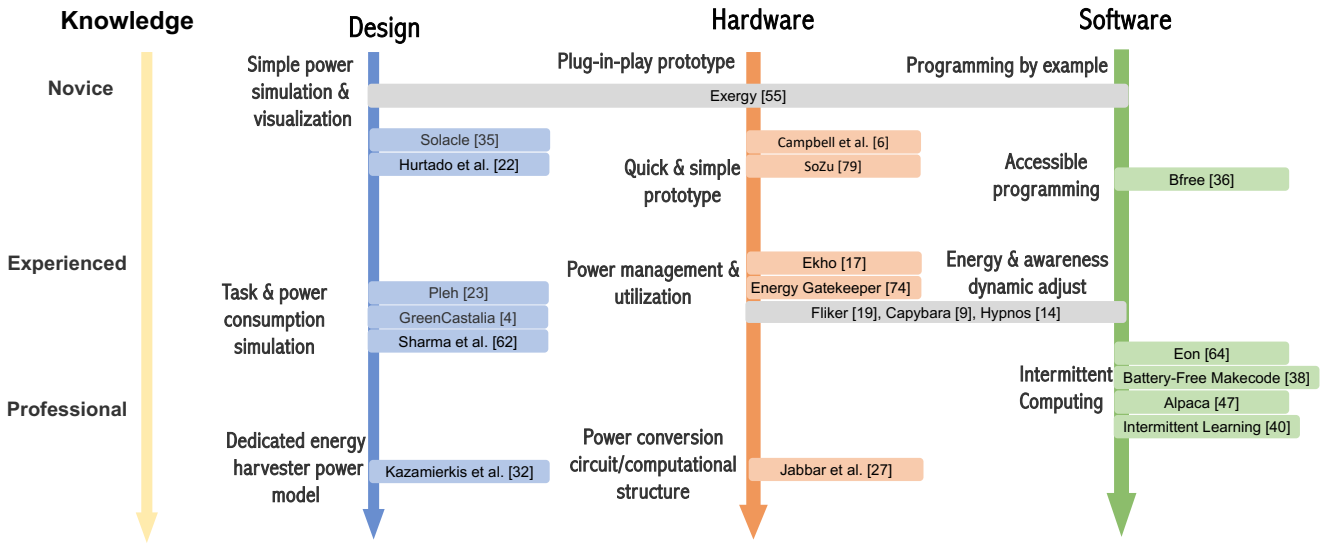

Figure 1: Summary of the existing works on facilitating energy harvesting development.

that use ambient light and kinetic and thermometric energy. Campbell et al. [6] proposed a solar cell-based IoT sensor node solution for indoor monitoring. Saoda et al. [60] presented guidelines for designing a generic platform for energy harvesting applications, suggesting that developers should have the flexibility to select energy sources and perform energy optimization on applications. Flicker [19] demonstrated a plug-and-play architecture that allowed kinetic, RFID, and solar energy harvesting with wireless transmitting capability and supported fault-tolerant computing in software that allows developers to perform storage calculations. Considering the programming aspect, BFree [37] is a system that integrates the Python language with energy harvesting hardware to help novice users build a long-term, ubiquitous computing system using energy harvesters. Park et al. [54] designed a simulation environment in which users could place the kinetic energy harvester in a virtual vehicle and receive a predicted power amount in a real situation. Iizuka et al. [23] introduced a simulation framework that fused the environment, storage, harvesting model, and tasks to facilitate the integration of energy harvesting systems.

3 RECALL CURRENT PLATFORMS AND TOOLS TO UNDERSTAND DIFFERENT LEVELS OF ENERGY HARVESTING DEVELOPMENT

Various works, including platform and bespoke tools, have emerged to help developers build self-powered IoT systems efficiently. These works were proposed to target developers or researchers with different knowledge levels of energy harvesting. To determine the main contributions of existing studies, we recalled some representative works, as shown in Figure 1. By interviewing experienced energy-harvesting developers in [55], three main development aspects of energy-harvesting development were identified: *design*, *hardware*, and *software*. We followed the taxonomy based on these three aspects.

Targeting the novice developer, Exergy [55] was designed to assist with a wind energy harvesting system. It shows how the toolkit is utilized as a creative design tool to engage novice developers throughout the design, hardware, and software aspects. Although it covers three development stages, the tools and techniques for each aspect are relatively simple. Specifically, for the design, some tools aim to simplify the simulation of harvested energy through visualization [22]. For example, Kim et al. [35] presented Solacle, a visualized tool to simulate the harvested light energy in indoor ambient conditions. By seeing its potential energy capability, users can easily determine where to locate the light energy harvesting IoT node indoors. These "shallow" tools address developers' understanding of how much power the energy harvester generates. With the development of design tools, some studies have focused on helping developers simulate sensing tasks and the corresponding power consumption of energy-harvesting IoT (i.e., how to use the simulated energy) to balance the sensing data transmission task during system design [4, 23, 62]. In addition to designing with simulated energy, more dedicated studies have investigated the simulation mechanism and principle, which can predict more accurate energy harvested according to the input of ambient factors [32]. This makes the tools progressively deeper and requires more knowledge of energy simulation.

For hardware, more straightforward tools have been developed to help quick and simple prototypes, which normally integrate the plug-and-play circuit system with multiple energy harvester inputs [6, 79]. For example, SoZu [79] is a tool for recognizing human motions through various energy harvesters' activities. More sophisticated tools focusing on the management and application of energy have emerged, with the ability to incorporate more peripheral, task, and environment-specific features [17, 19, 74]. These platforms

often have more complex hardware architectures, and some previous efforts also involve software development support to achieve dynamic task and power tuning. Developers are also required to be more familiar with the key hardware parameters of the energy harvesting system, such as output voltage, and maximum power tracking issues. One representative work is Flickers [19], which provides a faster and more usable energy harvesting tool that supports testing and enables dynamic energy storage without changing the hardware architecture. Similar works include Capybara [9] and Hypnos [14]. More in-depth hardware tools have become increasingly specialized, including higher-performance voltage conversion circuits [27] and power-efficient computer architecture.

At the software level, tools for novices often look for simple programming languages and example codes to get started quickly [55]. Novices typically need to switch their programming habits from other electronic systems to energy harvesting IoT [36]. Subsequently, more advanced system tools can implement energy-aware calculations. By monitoring the harvested energy, software algorithms are used to realize dynamic IoT node operations [14]. In addition, there has been a proliferation of programming tools for intermittent computing [38, 40, 47, 64]. Assisting users to have fast and easy intermittent computation experiences is the main contribution of these tools. For example, [38] instrumented intermittent computing in MakeCode improves the accessibility of programming languages for energy harvesting.

Among the three aspects discussed previously, energy harvesting technology involves more cumbersome expertise. This is one of the reasons why developing energy harvesting systems is challenging. Therefore, in existing works, novice developers are expected to work on the *design*, *hardware*, and *software* during the implementation of an energy harvesting system, which not only requires knowledge of power but also of hardware circuits and software programming. However, existing studies either expect novice developers to have a background in sensing, circuit design, or dynamic perceptual adjustments, or understand sensor energy consumption, power electronics, and so on. It is not necessary to consider sensing task simulations, circuit design solutions for energy management, and dynamic perceptual adjustments. Moreover, novice developers do not need basic knowledge of this kind of work, such as refined IoT node energy consumption, power electronics design, and energy sensing algorithms.

For novice developers, skills such as designing energy use and developing hardware and software in coordination usually need to be improved. However, most of the previous platforms aim to improve the usability of energy harvesting from a system perspective and only exist as standalone contributions. Little work has been done to improve the usability of energy harvesting from the perspective of novice developers. Exergy [55] is a novice-oriented development tool, but it targets only wind energy harvesting, which is a single object, and it does not explore the issues that users create in developing the system. Therefore, our study complements this important work by treating users' needs as a nuanced and dynamic variable—how novice developers apply energy harvesting techniques in their developments, what critical knowledge novice developers require, and what support they need.

4 EVALUATION PLATFORM

To explore and assess novice developers' needs, we designed an evaluation platform consisting of hardware circuits and a development tool chain. The evaluation platform helped the novice developers and the expert to better test the various modules of the system during the mentored physical prototyping study.

4.1 Design Considerations

As the output power from energy harvesters is small, deploying energy harvesters in applications typically needs to consider voltage regulation and energy accumulation. Thus, a power management circuit is important to connect energy harvesters to system loads. Specifically, power management ICs are supposed to perform DC–DC conversions with various typologies, such as boost converters, charge pumps, and fly-back converters. Considering adaptability to various kinds of energy harvesters, two application circuits for power management ICs were designed in the platform.

The first circuit is based on BQ 25504 [68], which fuses the maximum power point tracking (MPPT) function through impedance matching to enable maximum power output from energy harvesters. The configurable circuit parameters allow an adjustable output voltage. This circuit system accepts a minimum input voltage of 130 mV. The BQ series has become one of the most popular solutions for energy harvester-based systems, but its high cold-start voltage (over 600 mV) is one of its major drawbacks. The second power management IC is LTC3108 [43]. It uses a transformer-based voltage conversion and allows a minimum input voltage of 20 mV, the lowest input voltage allowed by IC energy management chips in the market. This IC does not maintain the MPPT function, and the conversion frequency is relatively low. These two types of power management circuits can satisfy most energy harvesters' input and can be used separately or together to form hybrid energy harvesting circuits.

As the main application scenario for energy harvest-based systems is IoT, we integrated a wireless communication module on the platform—the Bluetooth-enabled MCU, nRF52832. Using the Bluetooth protocol requires relatively low power consumption. This popular nRF52832 scheme has a wide source for application to terminal devices and the cloud.

4.2 Core Features

Figure 2 shows a block diagram of the design platform and the designed hardware evaluation platform. The entire platform was designed with a leaf-like form, and we left 16 general input/output pins from the microcontroller with castellated holes, which were easily tested and connected by external devices/systems.

Separate modules for testing. All three main parts (Bluetooth, power management IC1, and power management IC2) were designed separately and could be used individually (i.e., with separate input and output pins). This is useful for debugging the board's functionality and testing the operation of each part.

Wide support programs and application examples. Programming the Bluetooth module is relatively easy to access, with extensive online resources. Official mobile application development examples are available to help mobile design applications for terminals.

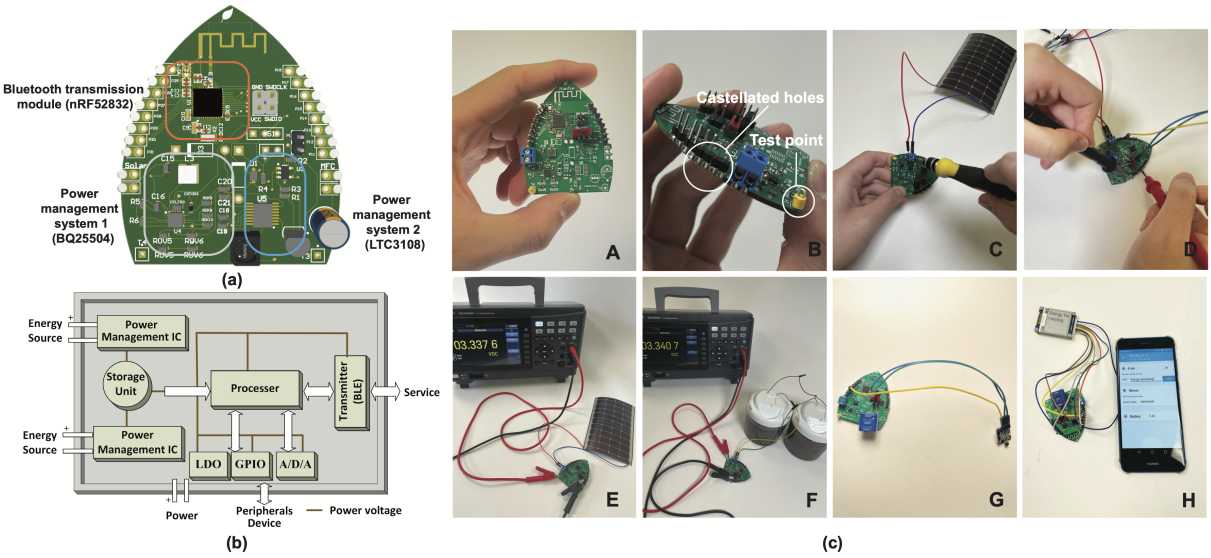

Figure 2: Designed evaluation platform. (a) 3D PCB design. (b) Functional block diagram of the designed platform. (c) Practical usage process of the platform. For example, it supports different input energy harvesters (E is the solar panel, and F is the MFC harvester) and different peripherals (G is the connection to the temperature sensor, and H is the E-ink display and the related mobile application).

Probe pins for voltage checking. Per the perspectives of the previous works [28], one of the main concerns is the uncertainty of energy harvester-based systems. Thus, we added probe pins to the board to facilitate the monitoring of the energy storage unit (e.g., super-capacitor/rechargeable battery) voltage and help the developers master the power conditions of the whole system.

5 USER STUDY CONFIGURATION

This study involved two tasks using an energy harvester to develop self-powered embedded systems. In addition to a wide set of energy harvesters (Figure 2) and an evaluation platform, we also provided a DC voltage source, digital multimeters, an oscilloscope, solder tools, supercapacitors, breadboards, a smartphone, and basic electronic components (e.g., jump wires, resistors).

We provided basic code examples and wrapper functions for the programming development aspect. The main programming development environment was based on the C language (i.e., Keil Embedded Development Tools), and the wireless signal-checking application was developed based on a smartphone. The study was conducted for two days for each novice developer, with each day allocated to one task. The duration of each task was 2 hours. Thus, each novice participant performed a 2-day, 4-h mentored physical prototyping study with the expert. On the first day of the study, the participants were first asked to sign an IRB consent form and provide demographic information, and the expert introduced the basic characteristics of the energy harvesters used (detailed process is presented in Section A.1). Afterward, the study commenced.

5.1 Methodology

Our work centered on a mentored physical prototyping study modeled after a *pair-programming study* [7, 59, 71, 80]. In the study, the recruited novice developer mainly implemented the system development, with the expert acting as a navigator and providing the necessary guidance. The novice developer was the driver, while the expert helped conduct assistive operations, such as soldering, according to the novice developer's design. If a novice developer was not proficient in C, the expert could help with coding according to the novice developer's design. These studies were recorded and transcribed later for analysis (the corresponding analysis process is shown in the Appendix). The responses from both novice developers and the expert were coded using affinity diagramming [16, 61] and thematic analysis [5, 63].

5.2 Participants

Novice developers: We recruited eight *novice developers* to participate in our study. All were novice electronic developers (i.e., undergraduate and junior Ph.D. students majoring in engineering disciplines) from an institution in North America. As indicated in Section 3, the recruited developers were new to all three levels of *design*, *hardware*, and *software* in terms of energy harvesting. They had no extensive experience with embedded systems, and none had experience developing systems using energy harvesters. This is also in accordance with the other novice-oriented toolkit [55]. Nevertheless, they had taken relevant classes and conducted electronic labs in their education and had more or less used some simple electronic prototyping platforms or components, such as Arduino, and sensor breakout

boards. Their detailed demographic information is shown in Table 1. Each novice developer was reimbursed for $100 in total.

Expert: The *expert* was a member of the research team (and a co-author of this paper) with extensive experience developing self-powered IoT devices using energy harvesting technologies.

Table 1: Demographic information of the novice developers in the user study. (Year indicates the year of development)

Users	Age/Sex	Year	Background
P1	24/Male	1	Simple electronic system (Arduino-based)
P2	26/Male	1	Temperature monitoring system (Arduino-based)
P3	23/Male	0.5	Embedded system software development
P4	27/Male	0.5	Simple sensor-based detection system (Arduino-based)
P5	25/Male	1	Remote control system (Raspberry Pi-based)
P6	25/Male	1	Motor controlling circuit system development
P7	24/Female	0.5	Laser detection system
P8	18/Male	1	Simple electronic system (Arduino-based)

5.3 IoT Development Tasks

We designed two basic self-powered IoT development tasks following a similar systematic structure: using a sensor to detect variations and an MCU to process the signal and control an actuator to execute. These two tasks were identified from the most common cases of IoT use.

Task 1: *An automatic outdoor lighting system.* The first task was to develop an automatic lighting system similar to solar-sensitive streetlights. The participant was asked to design a lighting system demo using a light sensor (TEMT6000), the evaluation platform, and the energy harvesters of their choice. The light was recycled from a commercial garden light product (i.e., an LED). All the participants were requested to make the system intelligent (i.e., act according to environmental conditions) and programmable (i.e., thresholds and logic can be adjusted later in the deployment). All signal processing and logic should operate locally on the MCU.

Task 2: *An interior door/window monitoring system.* The second task was to develop a monitoring system used in indoor environments. The system can be deployed on a door or window to alert users when an unexpected person opens it and enters the indoor environment. Specifically, the participant was asked to use a hall magnetic sensor (3144E) with a magnet. The two components were installed before the study began (i.e., the sensor on the door and the magnet on the wall).

Energy harvesters: We chose four types of energy harvesters (Figure 3), identified by their popularity in the research fields and maker/hobbit communities. If used properly, any harvester type could meet the energy needs for the two tasks, whether indoor or outdoor. The selected energy harvesters were as follows:

- Solar panel: Converting solar energy to electricity.
- Piezoelectric: Converting the mechanical stress to electricity.

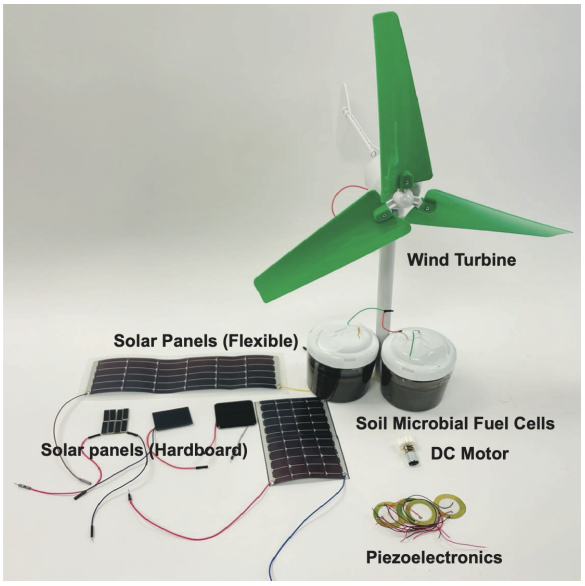

Figure 3: Energy harvesters provided in the mentored physical prototyping study.

- DC motor/Wind turbine: Converting the kinetic energy to electricity.
- Soil microbial fuel cell: Converting chemical energy in a microorganism into electricity through bacteria in the soil.

6 USER STUDY RESULTS

6.1 Findings from Observing the Study (OS)

To understand the workflow of our participants in their development of self-powered IoT systems, we encoded the video sequence (mostly based on conversations between the novice developers and the expert and their activities) into eight subtasks: 1) Test the sensor function, 2) Code, 3) Test the system function, 4) Measure the power consumption, 5) Measure the energy harvesters, 6) Calculate the capacitance and power, 7) Test the system connection, and 8) Improve robustness. Figure 4 visualizes the two requested tasks in this study, and Figure 5 shows the subtasks along the time axis.

Table 2 shows the quantitative measurements of the two tasks in terms of harvested energy, consumed energy, and the design choice of each participant. The participants had various design choices regarding system working time, energy harvesting time, and energy harvester type. For Task 1, the power measurement of the solar panel was conducted only in the indoor environment for ease of debugging and development. The participants differed in the degree of success of their program designs. For example, P8 and P6 were unsuccessful in both tasks. P1, P4, and P7 were successful in both tasks. P2 and P5 were successful only in Task 2. P3 was successful in Task 1.

6.1.1 Workflow. The participants generally followed the development paradigm of *components testing* (for sensor and energy harvester) - *system building* (realizing the system function) - *task debugging* (using energy harvesters to power the system).

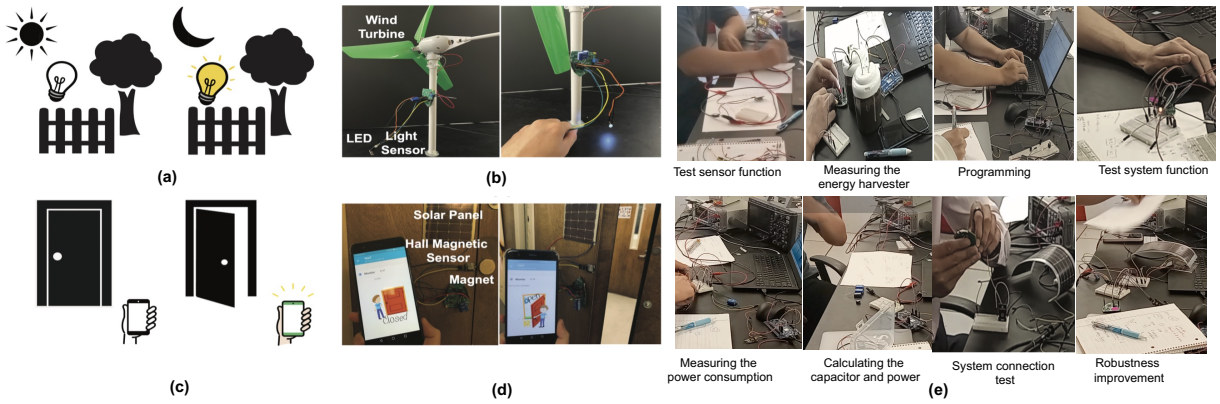

Figure 4: Two requested tasks in the mentored physical prototyping study: An automatic lighting system (a) with related demos of Task 1 of using a wind turbine as an energy harvester (b). An interior door and window monitoring system (c) with a related demo of Task 2 using a solar panel (d). Subplot (e) shows the participants' behaviors in programming, using the evaluation platform, and energy harvesters.

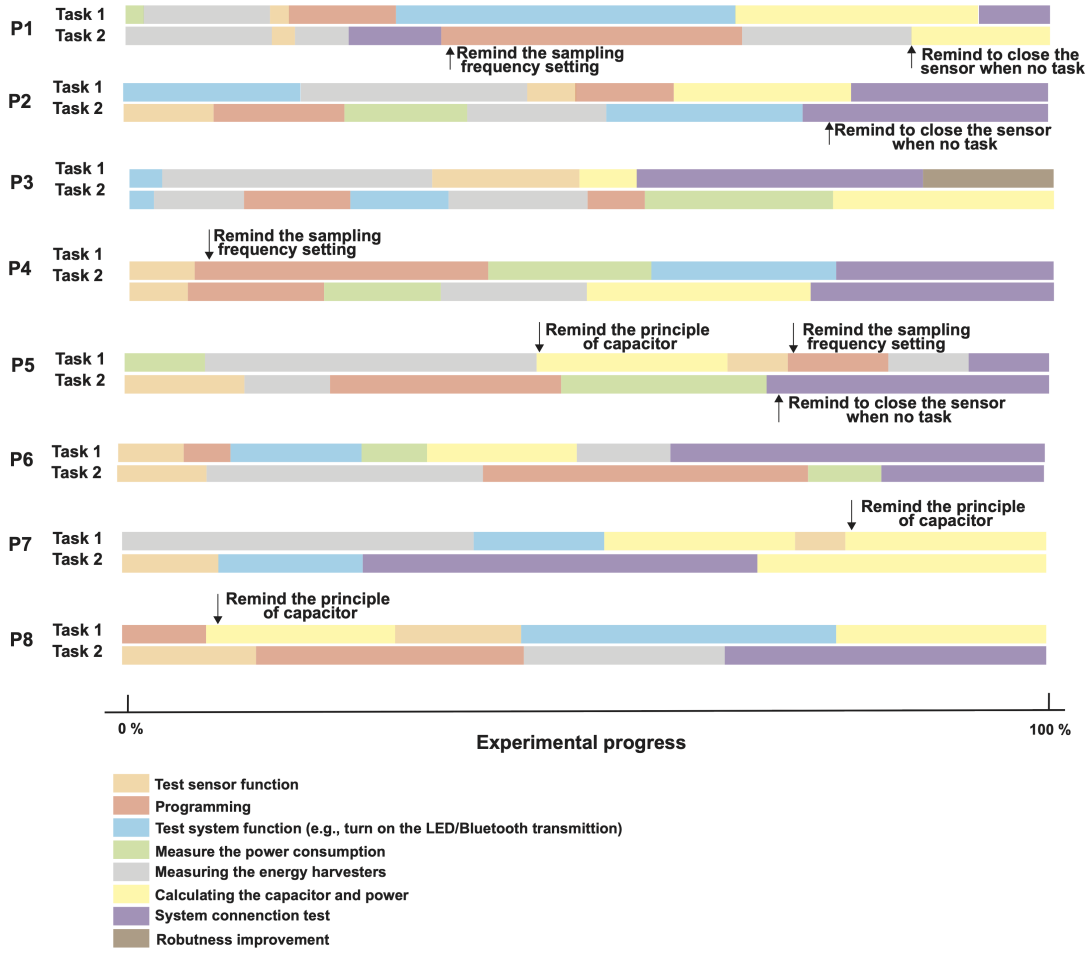

Figure 5: Participant time allocation during the mentored physical prototyping study.

Table 2: Quantitative measurements of the two tasks by all the participants in the mentored physical prototyping study.

Developer	Task 1			Task 2		
	Consumed Power	Selected Harversters	Harvested Power	Consumed Power	Selected Harversters	Harvested Power
P1	0.01W	Solar panel	0.4mW	0.3mW	Motor	2.6mW
P2	0.58mW	Solar panel	0.3 mW	0.27mW	Solar panel + Soil MFC	0.6mW
P3	1.56mW	Solar panel	3.68mW	0.3mW	/	/
P4	4mW	Solar panel	8mW	0.2mW	Wind Turbine	3.3 mW
P5	0.675mW	Wind Turbine	2.5mW	0.3mW	Motor + Soil MFC	0.5mW
P6	0.3mW	Solar panel	0.17mW	7.8mW	Motor + Solar panel	0.4mW
P7	1.25mW	Solar panel + Wind Turbine	12.6mW	0.1mW	Solar panel	0.3mW
P8	6mW	Solar panel	1.2mW	8.8mW	Solar panel	0.3mW

Components testing: As novice developers were more familiar with the concept of sensors or programming, most of the participants preferred testing the sensor function or getting familiar with the key parts of the system first. Thus, they implemented system functions using the DC power supply, which was later replaced by energy harvesters. Afterward, they started measuring the energy harvesters, which took a long time.

System building: In building systems, as we provided a demonstration programming, the developers did not put much effort into the function realization of the system. The time spent on this link was shorter.

Task debugging: The results showed that most of the participants understood the key part of developing such an energy harvesting system: matching the energy harvested to the system's consumption. They attempted to find a trade-off between energy consumption and energy harvesting, which was significant in developing an energy harvesting system.

For the developers who completed the task, the development normally took 1.5–2 h. Most of the participants spent comparatively longer on capacitor calculation, energy harvester measuring, and system connection tests, and less time on system and sensor function tests. They also needed to be reminded of how to calculate capacitance values and adjust the system's duty cycle. Based on the thematic analysis of the video recorded during the study and the observation notes of the expert, we identified two high-level themes: 1) gaps between concept and practice and 2) the need for assistive features. The findings for these themes are summarized as follows:

6.1.2 Gaps between concept and practice. a). Gaps between the ***power concept*** *and the* ***measurement of energy harvester***: The developers quickly understood the task's requirements. They typically started with the sensor's performance and performed basic tests to see how the sensor output relates to the input to determine the basic operational logic of the system. More questions were asked while powering the system with the energy harvesters. The frequently asked questions were:

> *"How do I measure the power of the energy harvester?"* (P2, P5, P7). *"What is the output of these [energy harvester])?"* (P3, P5).

Although the developers were briefed on how energy harvesters work and the output energy, it was clear that they had little idea how to convert this knowledge into practice. The basic concept of power is known to developers and can be applied in system design to some extent (power is needed to support the system's operation). However, they usually did not have the practical skills to measure power. For example, when P7 asked how to measure power, the expert needed to remind P7 of building a circuit loop using a resistor and measuring the current and voltage. However, when the expert asked what resistance value to choose, P7 did not give a specific reason (took one resistor randomly). The participants recalled the formula for power calculation, but it was difficult for them to determine which circuit configuration to use to measure the voltage and current, which are two constituent parameters in the formula. Specifically, they did not know the meaning of the output power, which should be determined by the load resistance in the circuit. To reach optimal output power, the impedance should be matched. This required elaboration and help from the expert, who assisted the novice developers in measuring the output power of the energy harvesters and the output power.

b). Gaps between the ***capacitor concept*** *and* ***deployment***: By using the energy harvesters, the participants quickly realized the characteristics of the unstable input of energy harvesters (e.g., using solar cells with power during the day and no power at night), and they were all aware that storage mechanisms could be used to mitigate the instability. However, when the capacitor was provided to the developers, at least one of the following questions would be raised by the participants:

> *"How does this [capacitor] work?"* (P5), *"How does the [capacitor] work in the circuit?"* (P5), *"How can I charge the [capacitor]?"* (P4, P5, P7), *"How fast does it charge?"* (P4), *"How do I increase the charging speed?"* (P2) *"How many volts will it be charged to?"* (P7), *"How does it discharge?"* (P5-P7), *"Is the charging current the same as*

the discharging current?" (P5, P7), *"How do I determine how much charge it holds?"* (P4, P5, P7).

The novice developers knew how a capacitor would be used in this project (i.e., to store energy to eliminate the effects of fluctuations in the environmental parameters). However, they required further guidance from the expert on the steps and design rules of thumb for capacitor use.

c). Gaps between the ***sensing concept*** *and* ***digital sampling***: Another question that was frequently raised was how to read the digital signal of the sensor during development. The participants understood the basic concepts of analog to-digital conversion and sampling but often needed to look into the setting of the sampling frequency to enable the system to operate in intermittent mode with the duty cycle. Therefore, it was often the case that the expert had to remind the participants about the steps to set the sampling frequency. Furthermore, the novice developers wondered how high/low should they set the sampling frequency. Nevertheless, they usually understood the intention of sampling sparingly, which was to preserve power:

> *"I would like to set the sampling frequency to [once every] 10 minutes because the outdoor light changes are relatively slow."* (P1). *"Let us set it to be 500ms first to test the system function, and then we can test the practical situation of opening the door to change it."* (P2).

6.1.3 Need for assistive features. a). Need for ***capacitor calculation***: More effort was put into the selection of the capacitor's value. Although the basic features of the capacitor (e.g., the capacitor can store and release energy) were known to novice developers, they were stuck in calculating the charging time and the required capacitor size after specifying the required energy. The expert often assisted the novice developers in using an online capacitor energy calculator to help them overcome the problem quickly. [2].

b). Need for ***optimization strategy***: Another theme from the observation notes was their strategies in handling the issue of the harvested energy not meeting the consumed energy. As the output power from the energy harvester is generally low, this is one of the most common problems developers must solve. The participants wanted to increase the harvested energy to balance out the consumed energy, which was often their strategy when first faced with the issue of an energy shortage. For example, in Task 1, P8 used only one solar panel for the electronic system at first and found that it did not work (out of the box). Thus, he added more solar panels to a series and tried to improve the voltage. As he did not consider the power issue, the energy harvesters still failed to meet the energy requirements of the system. For most developers, adding input energy harvesters to increase the amount of energy that could be collected was the most obvious solution, but the additional cost was also discussed:

> *"I can use many solar panels if the cost allows."* (P3)

Interestingly, compared with increasing the energy harvester count, it was not as obvious to our participants that they could also move the solar panels to a brighter environment, as they thought the brightness would not affect the output power much. The expert helped P3 place the solar panel near a window, and P3 found that the output current greatly increased.

> *"I didn't think the light would have such a big impact on the solar panel."* (P3).

Another interesting note is overlooking the key solutions. In this case, one commonly overlooked solution was to reduce energy consumption, especially under space and cost constraints, which limit the number of allowable energy harvesters. This reflected a lack of understanding of the sensor's energy consumption. Although the participants were aware that optimization could reduce energy consumption, it was unclear to them how much could be reduced and whether it was worth the trouble to compare with simpler solutions of having more energy harvesters. This unknown led to the neglect of this part of energy consumption. When the expert guided the developers in dealing with the energy consumption of sensors, the participants remarked the following:

> *"I didn't know that the hall sensor consumes so much energy."* (P2) *and "This is where a lot of the energy is consumed by the system."* (P5).

Another energy optimization strategy is diversifying the system's energy harvester profile to improve robustness against low-energy environmental conditions. At the end of Task 1, the expert asked questions, hoping to solicit the use of multiple types of energy harvesters. As the application of Task 1 was for outdoor-use scenarios, the environmental factors changed more significantly than the indoor scenarios (Task 2). However, this limitation also came with a richer set of energy sources that, if leveraged effectively, could be advantageous. Specifically, the expert asked, "If we choose to use solar cells, does the current design support lighting conditions in the wild, for example, when it rains for several days in a row or when there is not enough light?" Although it was not their first response, most of the developers eventually came up with the solution of leveraging multiple types of energy harvesters:

> *"Include the use of MFC or wind energy harvesters to make up for the missing energy amount."* (P5).

A hybrid harvesting scheme using different energy harvesters that complement each other to increase the input energy was the solution. However, this solution did not address (if not worsened) the issue of the uncertainty of energy profiles by introducing more variability from the environment. In addition, an increase in energy harvester types could lead to false confidence in the energy supply. To address this issue, the participants mentioned a solution for monitoring the harvested energy and adjusting the power consumption accordingly. According to P4:

> *"We can monitor the voltage in the capacitor to derive the energy, and preset an information table in the system to dynamically adjust and calculate the time to provide lighting in conjunction with the local sunrise and sunset times."* (P4)

Maintaining the energy match of the output and input and dynamically adjusting the energy at the consumption end is a more intelligent solution to improve the robustness of the energy harvesting system. Novice developers generally prefer optimizing the

[2]Online Capacitor calculator tool: https://www.omnicalculator.com/physics/capacitor-energy

energy supply by increasing the number and type of energy harvesters to meet demand (e.g., power consumption, robustness). However, there are more issues to consider when using multiple energy harvesters in conjunction with impedance matching, and energy harvester response. These were the issues that novice developers did not consider when developing systems using energy harvesters, constituting the "unknown unknowns," which we relied on our interview with the expert to uncover. For example, it was unknown to most novice developers that complementary dynamic adjustment and dynamic balancing solutions were possible. During the experiments, most of the novice developers needed to be reminded by the expert.

6.2 Interview with *Novice Developers* (IN)

After the two tasks were completed, we conducted a short interview with the novice developers, following a semi-structured interview scheme. Their responses were coded using thematic analysis and affinity diagramming (the method is introduced in Section A.2). The results also echoed the main themes identified in the observations.

6.2.1 Gaps between Concept and Practice. a). Gaps between ***energy harvesters*** *and* ***battery powering***: We asked the participants what they thought after completing the development of these two tasks and what they felt they had learned that they had not noticed before. We considered many comments regarding energy harvesters. Almost all of the developers indicated they had a more practical understanding of energy harvesting technology:

> *"I didn't expect the solar cell to produce such different outputs for different lighting conditions"* and *"I thought the soil MFC was the most reliable, but I didn't expect it to have such a low output voltage and power". "The solar cell was the most surprising to me. I thought it would be as stable as a battery [to provide a stable voltage/current output]."* (P2)

Similar comments on the power profiles of harvesters also emerged from the interviews with P1, P7, and P8. Although no developer thought of energy harvesters entirely as batteries, they all used energy harvesters with expectations and conventions inherited from their previous experience with batteries. For example, they expected them to be as easy as using battery power and that their power and output would be stable. The novice developers commented that they could be "naïve" in the use of energy harvesters, constantly trying to draw analogies between energy harvesters and batteries. This mindset could be a result of failure and frustration rather than thinking of energy harvesters as a function of power with input parameters of environments and system implementation, both of which were noted as elements of surprise by the participants in their interview.

6.2.2 Need for Assistive Features. a). Need for ***power simulation***: We then asked the novice developers about their willingness to use energy harvesters and, in particular, whether they would consider using them instead of batteries (e.g., power banks or button cells) when developing similar IoT systems. Overall, the participants indicated that they would use energy harvesters to develop systems if certain conditions were met. These conditions were all related to tedious calculations and the unknown considerations of critical design elements, both of which could be addressed with assistive features in future development tools. For example:

> *"I would choose to use an energy harvester if it was in outdoor conditions because the battery would need to be replaced."* (P1). *"I would trust it more if I could know exactly how much output power each energy harvester can have."* (P2). *"If I could be told directly how much power each energy harvester could produce and how much energy the system would consume, I could just connect them easily."* (P3). *"I would like to trust the energy harvester if I understand all of the [system] parts' power consumption."* (P4).

Thus, more explicit information about power is needed. The novice developers relied on the power harvester to tell them how much power could be generated so that they could be more comfortable in the design and debugging phases. This demonstrates the importance of exposing critical design elements as early as possible in the initial stages. Simulation can help with this. Power information should be provided to users as early as possible, and users should be allowed to design based on the simulation information, thus reducing actual trial and error.

6.3 Interview with *Expert* (IE)

We interviewed the expert to compensate for the participants' feedback, which could require completion or could be biased due to a lack of knowledge (effect of "unknown unknowns"). With the interview with the expert, we expected to validate the participants' findings and discover complementary new insights into the participants' needs. We navigated the interview with the expert using the following questions:

- Information exchange/sharing. Was there information asymmetry between expert and participants. If so, how did this affect participants' development of self-sustaining systems?
- Communication experiences. What were the experiences of exchanging information with participants in collaborative developments. Were there any obstacles that expert encountered in this process?
- Navigation strategy. How did expert solve problems or deal with difficulties in communicating with participants during the collaborative developments?

6.3.1 Gaps between concept and practice. The expert revealed that the novice developers needed to learn to put constituent concepts in energy harvesting into practice. These concepts include *capacitor*, *power*, and *duty cycle*. The first two findings generally agree with the observations of the study.

a). Gaps between the ***capacitor concept*** *and* ***charging and discharging***: The expert revealed that although all the participants knew that capacitors could store energy, few knew how to make or use the connections. The expert reported an example in which one participant asked, "Can a capacitor discharge when it is not fully charged?" Another even more troublesome example was when the participants expected their system to work only after the capacitor was fully charged without asking the question. In fact, discharging a capacitor does not need to wait until it is fully charged. The actual discharge of the capacitor is affected by the situation of the circuit

and its capacitance. This problem also revealed that the novice developers had only a conceptual understanding of capacitors and were not familiar with their practical use. The expert also found power to be another front in which knowledge of practice was largely missing.

> *"Participants knew P is calculated by multiplying U with I, but what they didn't know is that both voltage and current have to be supplied to get the expected power."* (Expert)

b). Gaps between ***power concept*** *and* ***load***: What the developers overlooked was that the output power of the harvester is decided by its load, which changes the output voltage and current in energy harvesting. Assuming either of them was constant only led to confusion and frustration. For example, the expert mentioned an example in which P8 tested an open-circuit voltage of 5 V from the solar panel. When P8 connected the solar panel to the system, he found that it could not supply power to the system and was confused about why the output voltage decreased rapidly.

c). Gaps between the ***sensing concept*** *and* ***duty cycle***: On the power consumption end, it was common for novice developers to overlook the option of reducing it by optimizing the sensing or actuation duty cycle. The expert observed that most of the participants immediately found ways to increase the power supply once they learned about its shortage in supplying the consumption of their applications while overlooking the other part—reducing the consumption on the LED (actuator) or the sensor.

Further communication between the expert and the novice developers revealed that the reasons for this mindset were because 1) the novice developers believed that power consumption was already very low and that improving it would not be worthwhile, 2) they did not know the existence of methods for reducing energy consumption, and 3) they thought the methods to reduce energy consumption were challenging. The expert reported that the novice developers tended to think that the sensors had very low power consumption and did not require merit optimization. However, hall effect sensors consume considerable power (3.5 mW) without optimization of the sampling rate or adoption of power gating approaches. Furthermore, even if the novices noticed the considerable power consumption of the sensors, they needed to learn how to correlate it with the sensor's sampling rate. We found that the gaps between practice and concept in the case of the LED were much better than those in the case of sensors. The expert observed that the novice developers had a better understanding of perceivable elements. The light from the LED was a constant reminder of "hey, I am consuming power, so do something about it," whereas the sampling rate of the sensors was abstract and thus "invisible" to them in their developments.

6.3.2 Need for assistive features. Most of the assistive features were reported in the previous sections, while the interview mostly focused on the "unknown unknowns"—knowledge that the novice developers themselves would not know that could help them. Findings regarding this were scattered throughout the interviews.

For example, depending on the design points at different stages, when using energy-consuming or power supply units, it is usually necessary to specify their power. When selecting energy storage elements, it is necessary to specify how much energy is to be collected, as the time of charging needs to be taken into account. The novice developers spent a significant amount of time converting power (needed for the LED/sensor) into energy (design factor for capacitance selection) and power (needed to be harvested) in the iteration of their system development. This calculation could be facilitated and even automated using built-in features on future development tools.

a). Need for ***capacitor selection recommendation***: The expert also noted that the size selection for capacitors could be a space in which support is needed. The novice developers relied on a trial-and-error approach to select capacitance for storing energy. For example, once the energy required was calculated, they would try a small capacitor (e.g., 0.1 F) to store it. If the energy harvested charged up the small capacitor too quickly, they would increase it to a larger capacitor (e.g., 1F) to improve the energy harvesting efficiency, as energy does not charge into the capacitor further after it reaches the maximum voltage. However, a capacitor that is too large can result in a longer energy harvesting time before reaching a usable voltage level. This trade-off posed difficulties in capacitor selection for all novice developers in the study.

b). Need for ***hints***: The expert reported a general problem in energy harvesting of having too many variables to consider in optimization. Guidelines, recommendations, or proactive hints compared with passive information feeds (e.g., status indications, logs, and visualizations) could help reduce the confusion caused mainly by novice developers needing to know where to start and what should be the plan of attack. This support is even more important given the ebbs and flows of energy supply and consumption due to varying environmental conditions and application needs. The novice developers had little clue about which variable to tune in response to the change in dynamism between energy supply and consumption.

6.4 Summary

We summarized the findings from observing the study and interviews with novice developers and the expert (Table 3). We observed that for the informational gap, novice developers need to bridge it with relevant hands-on experience in basic electronics, which could lead them to acquire the skills necessary for energy harvesting systems faster. These fundamental physical electronics informational gaps were previously dispersed in various applications in the electronics field, and the energy harvesting system integrated them. The problems specific to energy harvesting technology are mostly reflected in the assistive features needed to help novice developers. This also tells us what kind of skills they need to be educated to practice and what kind of help they need when introducing a novice to energy harvesting system development.

We further connect the relevant informational gaps and assistive features with issues found in the study (Figure 6). To some extent, the informational gaps can also be reflected in the need for assistive features, which are the potential functions of what a toolkit aims novice developers to have (Table 4). However, most solutions solve practical energy harvesting-related physical electronics problems through assistance methods. We believe that these existing tool-based solutions are insufficient for novice developers and that the

Table 3: Findings on the informational gaps and the need for assistive features from the user study. OS: observing the study; IN: interview with the novice developers; IE: interview with the expert.

Gaps between concept and practice	*OS-a & IE-b:* *Power concept and measurement of energy harvester/load* *OS-b & IE-a:* *Capacitor concept and deployment/charging and discharging* *OS-c & IE-c:* *Sensing concept and duty cycle* *IN-a:* *Energy harvesters and battery powering*
Need for assistive feature	*OS-a & IE-a:* *Capacitor calculation/selection recommendation* *OS-b & IE-b:* *Optimization strategy/Hints* *IN-a:* *Energy harvester power simulation*

Table 4: Summary of issues and existing solutions.

Issues	Findings	Category	Solutions in existing platforms/tools	Used in study
Physical electronics issues	Measurement of energy harvester/ power with load	Informational gaps	Power profile tool Voltmeter	✓
	Digital sampling/ duty cycle		Demo programming [23, 55]	✓
	Capacitor deployment/ charging and discharging		Online calculator	✓
	Capacitor selection recommendation	Assistive features	Online calculator	✓
Energy harvesting issues	Optimization strategy/hints		/	
	Energy harvester power simulation		Power simulation model [22, 32, 35, 55]	

problems are still open for better solutions, such as power profiles or logging [8]. To better reveal novice developers' needs, we did not provide energy harvester power simulation tools in the experiment. The existing platforms are well aware of and integrated with relevant functions for this part of the solution (e.g., Exergy [55] and Solacle [35]). An unsolved problem lies in the provision of power optimization strategies. The novice developers were often at a loss when faced with problems such as energy fluctuations, excessive power consumption, and insufficient harvested energy due to a lack of experience.

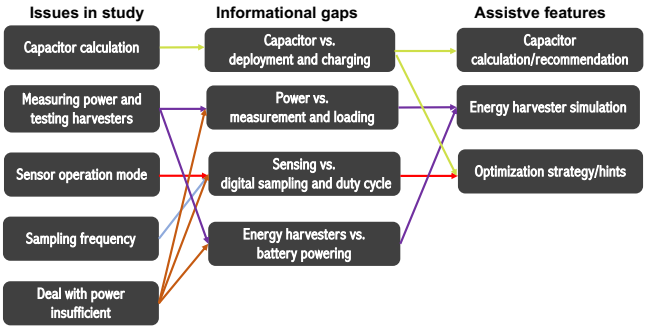

Figure 6: The relationship between informational gaps and assistive features found in the user study.

7 DISCUSSION AND IMPLICATIONS FOR DESIGN

7.1 Education Efforts

The experiments in this paper revealed many practical issues in basic physical electronics that might have nothing to do with the energy harvesting technology itself as a related underlying technology. However, we believe that, as a gap, these issues are what novice developers should overcome when developing energy harvesting systems. Related issues are also reflected in [55], which is an energy harvesting toolkit for novice developers. When promoting

the toolkit to novice developers, the authors conducted a workshop and tutorials for them before using the relevant Exergy toolkit. The novice developers needed to be familiar with the related concepts before they could start using the toolkit. This "educational tutorial" was intended to solve the relevant informational gaps of novice developers. Therefore, a toolkit intended for novice developers should consider the users' informational gaps between concept and practice more carefully. To fill this informational gap, educational efforts should be centered on the involvement of lab components in the curriculum, giving students hands-on experience in using harvesters in embedded system projects. As universities have long been integrating low-cost, novice-friendly hardware into their curricula [29], having an energy harvester could be incremental and not disruptive to the existing curriculum setup. Educators have shared their experiences in this domain, such as in a survey of energy harvesting techniques for engineering courses [11] and curriculum setup [53].

7.2 Functions to Bridge the Informational Gaps

Although education can address these informational gaps to some extent, we do think that it can also be integrated into a novice developer-oriented toolkit for energy harvesting. The toolkit needs to consider the practical issues of novices and provide more assistance. This kind of help should not only be about the use of skill-level tools (e.g., instrument/power profile). A solution is to consider more example demonstrations. These demos are similar to game guides or product manuals. They should "awaken" novice developers to make up for the practice of previous concepts in an easy-to-understand manner. For example, regarding the ***power concept*** and ***measurement/loading***, the toolkit can start with basic concepts that users are familiar with (e.g., the calculation formula of power) through example demos and guide users in applying power in practice (e.g., measuring a power source, clarifying the maximum power and output power). In addition, the example of the ***power concept*** further illustrates the comparison between ***energy harvesters*** and ***battery power***. By showing that the power output of an energy harvester depends on relevant environmental factors, it is possible to identify the differences between an energy harvester and a battery and their advantages (e.g., it can be seen as an unlimited source of energy) and disadvantages (e.g., it is subject to fluctuations in environmental factors). Similarly, in the programming process for ***sensing***, users are prompted to pay attention to programming related to "intermittent" work. When novice developers realize that energy storage units can be used to buffer energy fluctuations, the toolkit can show through simple examples how the ***capacitor*** should be used in the system and what the principles are. Closing these informational gaps can help users better discover the main knowledge of energy harvesting and open the key to enter its domain.

7.3 Logging for Optimization Strategy

An extensive amount of logging can help novice developers uncover problems in their development and come up with informed solutions. As what the participants desired in the study, the energy optimization strategy depends on information that can be straightforwardly acquired in logging. For the same purpose, a checklist approach can be used to help developers check whether their designs have been optimized, especially for obvious fronts on which energy could be saved (e.g., intermittent sensor turn-on and turn-off). Such methods of providing help have been widely used to help developers use complex software tools [2, 44].

7.4 Further Expand Simulation to Digital Twins

We also highlight the importance of using digital tools, such as simulations or even digital twin techniques, to move exploration from the physical to the digital realm. This point reflects the need for the simulation of energy harvesters during the study. Many previous systems have demonstrated simulations for energy-harvesting applications [23, 54]. The simulation of energy harvesters can lead to optimization that guides the selection of harvester parameters (e.g., size of a solar cell) and code-level implementation (e.g., sampling rate and duty cycle), as shown by Lizuka et al. [23]. Moreover, simulation tools may potentially turn into digital twins, with real-time monitoring capabilities possibly achieved using debugger types of adaptor boards or data exchange mechanisms in firmware. With rich data logged from deployments, including measurements that can be difficult to simulate (e.g., environmental factors), digital twins can facilitate the exploration of novice developers later in the development phase in cases in which they need to find alternative approaches.

These digital twin systems should be more modular than the status quo, allowing for more flexible explorations in a trial-and-error manner. The trial-and-error approach has been recognized as the preferred strategy for developers when faced with complex objects [34, 50]. These digital twin systems can be combined with calculation tools (e.g., a capacitor size calculator) and a software programming platform to collaborate when developing the required programs. For example, Weniner et al. [70] proposed a guided exploration to help novice developers create monitoring tools, highlighting the most important information to users and providing relevant suggestions.

7.5 Composability/Plug and Play

In the physical layer, it is also important to help developers make connections efficiently and test them, as we found that much time was spent on tasks regarding connectivity in our study (Figure 5 purple). The results from this study confirm the benefits of modular designs, which have shown promise in previous studies (e.g., [6, 20]) in facilitating the assembly and measurement of physical devices. The novice developers found one major difference between batteries and energy harvesters—the ability to "package" energy to be later used in modules. For example, batteries support physical connections, including series and parallel connections, achieving the desired voltage and current in a flexible manner enabled by the modular design of batteries, which is a plug-and-play experience. Aside from the fact that it is faster to make connections between modularized harvesters and the system, novice developers can also benefit from modularized harvesters in reducing the possibility of error. In this sense, supporting tools that can yield "just work", use experience can yield the same benefits. We argue that providing a robust "just work" experience can be a better focus for tool research in this space. Another benefit is the automatic match impedance

demonstrated by commercially available energy management ICs (e.g., [68]), further mitigating the source of errors while preserving novice developers' time for more design iterations.

7.6 Avoid "Creeping Featurism"

There is beauty in the fundamentals, as pointed out by pioneering research in creativity-supporting tools in the caveats for "creeping featurism" [57]. Guo argued that minimizing user options would be preferable in the long run to avoid overwhelming novice users and to lower the burden of maintenance [15]. In developing energy harvester-based IoT devices, developers need to put more effort into eliminating the instability caused by unstable environmental factors, such as real-time scheduling, dynamic load adjustment, data storage under checkpoints, and energy storage mitigation of monitoring. However, developing many optimizations and providing as many assistive tools as possible can pose the opposite effect, leaving developers with a dizzying array of features, unnecessarily increasing the number of "tools" and taking the focus away from the awareness of in-/outlets and variables developers need to tune.

In addition, featurism packages fundamentals into "black boxes" that hide important information from novice developers that could lead to the unveiling of potential problems or solutions. Existing popular IoT development platforms, such as Arduino, Raspberry Pi [3], and Micro: bit [4], have appealed to a wider range of developers. The platforms have a high degree of system encapsulation to lower the threshold of expertise for use. For example, users can easily plug and unplug electronic components and wires physically. To form an accessible path, they only need to recognize the appropriate pins and simple electronics knowledge (e.g., positive and negative electrodes). However, convenient development with a low knowledge threshold hides much information about the performance of physical electronics (e.g., supply voltage and power). On the software level, the demo code guides users to rewrite the core parts without requiring more knowledge of embedded system development. Thus, expertise such as digital sampling is disregarded. The over-concealment of specialized knowledge also makes existing platforms less suitable for novice developers of energy harvesting to get started. Such platforms, which are designed to facilitate development, hide the key elements of energy harvesting system development, such as power and sampling frequency, so that developers who have formed a development habit with these platforms will be exposed to various problems when facing energy harvesting system development.

Therefore, the struggle to have a low floor while keeping the ceiling high will likely continue to exist in the supporting tools, and we should be mindful of the risk of having too many features that skip, combine, or hide fundamentals from developers. We urge the community to focus on improving the awareness of and capability of developers in in-/outlets and variables for energy, as opposed to features in future supporting tools for building.

[3]https://www.raspberrypi.com/
[4]https://microbit.org/

8 LIMITATION

8.1 Lack of Investigation into True Intermittency

Although our evaluation platform features all the components needed to implement systems that face true intermittency, we decided to exclude it from the current study's scope to accommodate novice developers' lack of experience. In other words, advanced techniques in intermittent computing, such as timekeeping and checkpointing, still need to be implemented and tested. In reality, power failures can be frequent and difficult to predict. The target applications that the novice developers implemented included data processing, sensing, communication, and actuation, and we asked them to refrain from considering (implementing) scenarios in which power failures could disrupt these functions (e.g., clock resetting and lost volatile memory). Future efforts are needed to investigate novice developers' implementation of IoT systems with intermittent computing schemes.

8.2 Missing Data Points from Developers of Various Levels

Continuing from the previous limitation, we should have considered expert developers or developers with more embedded system development experience in the current study. We suspect that expert developers may have fewer gaps between concept and practice, while novice developers without experience in embedded system development need more help to bridge these gaps. However, the same assistive features obtained from the study are still required for all developers, regardless of their level. Future work continuing this line of research should look into informational gaps and assistive features for developers of various levels, including experts in developing IoT systems with energy harvesting and K12 students who have just begun to learn about embedded systems. Additionally, considering the diverse age and gender participants could help a more comprehensive understanding of the related IoT system development with energy harvesting.

8.3 Limited Variety of Evaluated Applications

Finally, the two target applications could not represent all applications of the IoT. For example, the two examples we selected were stationary, whereas self-powered IoT systems could also take wearable or mobile forms that have entirely different sources of energy (e.g., the body temperature or kinetic energy of a wearer) and constraints in optimization (e.g., size, weight, and flexibility). Even for stationary IoT systems, a diverse set of applications could be considered in the future, such as applications in the wild that are far from human activities (e.g., wildlife/wetland preservation). These applications demand different amounts of energy and levels of robustness. Other design factors weighing more than energy harvesting efficiency must also be considered. For example, self-powered IoT systems must not pollute the sensitive ecological environment in the wild. These applications were not covered in this study, and they merit further investigation in our future work.

9 CONCLUSION

We conducted the first user study to investigate the support needed for novice developers in creating self-powered IoT systems using energy harvesting. We recalled the represented existing works to facilitate the development of self-powered IoT systems with different developers' knowledge levels and created a customized energy harvesting platform for evaluation. We then conducted a two-day user study with this platform in a mentored physical prototyping paradigm, teaming up novice developers with an expert in the development of two target applications—an automatic lighting system and a home security system—both of which were required to be self-powered using the platform and a wide array of energy harvesters provided in the study. We performed a thematic analysis on video transcripts, notes from the experimenters, and interviews with both the novice developers and the expert. Our findings point to informational gaps and assistive features and call for future efforts on tools to support novice IoT developers' adoption of energy harvesting technology to improve the sustainability of computing and IoT applications. Overall, we believe that our study fills a gap in the literature and provides a foothold for future efforts.

ACKNOWLEDGMENTS

This is supported by Research Funds for the Central Universities Grant (XJSJ23109) and in part by National Science Foundation Grant IIS-2228982. The author would like to thank the support from Prof. Yang Zhang and the HiLab in UCLA.

REFERENCES

[1] Sara Al-Sodairi and Ridha Ouni. 2018. Reliable and energy-efficient multi-hop LEACH-based clustering protocol for wireless sensor networks. *Sustainable Computing: Informatics and Systems* 20 (2018), 1–13.

[2] Oscar D Andrade, Nathaniel Bean, and David G Novick. 2009. The macro-structure of use of help. In *Proceedings of the 27th ACM international conference on Design of communication*. ACM New York, NY, USA, 143–150.

[3] Abu Bakar, Alexander Ross, Kasim Sinan Yildirim, and Josiah Hester. 2021. REHASH: A Flexible, Developer Focused, Heuristic Adaptation Platform for Intermittently Powered Computing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 3 (Sept. 2021).

[4] David Benedetti, Chiara Petrioli, and Dora Spenza. 2013. GreenCastalia: An energy-harvesting-enabled framework for the Castalia simulator. In *Proceedings of the 1st International Workshop on Energy Neutral Sensing Systems*. 1–6.

[5] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[6] Bradford Campbell and Prabal Dutta. 2014. An energy-harvesting sensor architecture and toolkit for building monitoring and event detection. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*. ACM, New York, NY, USA, 100–109.

[7] Mehmet Celepkolu and Kristy Elizabeth Boyer. 2018. Thematic analysis of students' reflections on pair programming in cs1. In *Proceedings of the 49th ACM technical symposium on computer science education*. ACM, New York, NY, USA, 771–776.

[8] Alexei Colin and Brandon Lucia. 2016. Chain: Tasks and Channels for Reliable Intermittent Programs. In *Proc. OOPSLA* (Oct. 30 – Nov. 4). ACM, Amsterdam, The Netherlands, 514–530.

[9] Alexei Colin, Emily Ruppel, and Brandon Lucia. 2018. A Reconfigurable Energy Storage Architecture for Energy-harvesting Devices. In *Proc. ASPLOS* (March 24–28). ACM, Williamsburg, VA, USA, 767–781.

[10] Jasper De Winkel, Vito Kortbeek, Josiah Hester, and Przemysław Pawełczak. 2020. Battery-free game boy. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 3 (2020), 1–34.

[11] Eric C Dierks, Jason M Weaver, Kristin L Wood, Kendra Crider, and Daniel D Jensen. 2011. Energy harvesting for engineering educators. In *2011 ASEE Annual Conference & Exposition*. 22–565.

[12] EnOcean. 2022. EnOcean. https://www.enocean.com/en/.

[13] Graham Gobieski, Brandon Lucia, and Nathan Beckmann. 2019. Intelligence beyond the edge: Inference on intermittent embedded systems. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. 199–213.

[14] Alberto González-Pérez and Sven Casteleyn. 2022. Hypnos: A Hardware and Software Toolkit for Energy-Aware Sensing in Low-Cost IoT Nodes. *IEEE Internet of Things Journal* 9, 15 (2022), 13524–13541.

[15] Philip Guo. 2021. Ten Million Users and Ten Years Later: Python Tutor's Design Guidelines for Building Scalable and Sustainable Research Software in Academia. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. 1235–1251.

[16] Gunnar Harboe, Jonas Minke, Ioana Ilea, and Elaine M Huang. 2012. Computer support for collaborative data analysis: augmenting paper affinity diagrams. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*. 1179–1182.

[17] Josiah Hester, Timothy Scott, and Jacob Sorber. 2014. Ekho: Realistic and Repeatable Experimentation for Tiny Energy-harvesting Sensors. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems* (Memphis, Tennessee) *(SenSys '14)*. ACM, New York, NY, USA, 330–331. https://doi.org/10.1145/2668332.2668382

[18] Josiah Hester, Lanny Sitanayah, and Jacob Sorber. 2015. Tragedy of the coulombs: Federating energy storage for tiny, intermittently-powered sensors. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. ACM New York, NY, USA, 5–16.

[19] Josiah Hester and Jacob Sorber. 2017. Flicker: Rapid prototyping for the batteryless internet-of-things. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. ACM, New York, NY, USA, 1–13.

[20] Josiah Hester and Jacob Sorber. 2017. Flicker: Rapid Prototyping for the Batteryless Internet-of-Things. In *Proc. SenSys* (Nov. 6–8). ACM, Delft, The Netherlands, 19:1–19:13.

[21] Josiah Hester, Kevin Storer, and Jacob Sorber. 2017. Timely Execution on Intermittently Powered Batteryless Sensors. In *Proc. SenSys* (Nov. 6–8). ACM, Delft, The Netherlands, 17:1–17:13.

[22] Gerardo Hurtado Hurtado, Jose A Romero, and Carlos S López-Cajún. 2016. Energy harvesting simulator. In *2016 12th Congreso Internacional de Ingeniería (CONIIN)*. IEEE, 1–7.

[23] Tatsuya Iizuka, Yoshiaki Narusue, Yoshihiro Kawahara, and Tohru Asami. 2016. Planning simulation tool for designing energy harvesting applications. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*. ACM New York, NY, USA, 289–292.

[24] Natsuki Ikeda, Ryo Shigeta, Junichiro Shiomi, and Yoshihiro Kawahara. 2020. Soil-Monitoring Sensor Powered by Temperature Difference between Air and Shallow Underground Soil. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–22.

[25] Seiko Instruments. 2022. S-882Z. https://www.alldatasheet.com/datasheet-pdf/pdf/222154/SII/S-882Z.html.

[26] Texas Instruments. 2022. BQ25570. https://www.ti.com/product/BQ25570.

[27] Hamid Jabbar, Young S Song, and Taikyeong Ted Jeong. 2010. RF energy harvesting system and circuits for charging of mobile devices. *IEEE Transactions on Consumer Electronics* 56, 1 (2010), 247–253.

[28] Neal Jackson, Joshua Adkins, and Prabal Dutta. 2018. Reconsidering batteries in energy harvesting sensing. In *Proceedings of the 6th International Workshop on Energy Harvesting & Energy-Neutral Sensing Systems*. ACM New York, NY, USA, 14–18.

[29] Peter Jamieson and Jeff Herdtner. 2015. More missing the Boat—Arduino, Raspberry Pi, and small prototyping boards and engineering education needs them. In *2015 IEEE Frontiers in Education Conference (FIE)*. IEEE, 1–6.

[30] Hrishikesh Jayakumar, Arnab Raha, Woo Suk Lee, and Vijay Raghunathan. 2015. QuickRecall: A HW/SW approach for computing across power cycles in transiently powered computers. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 12, 1 (2015), 1–19.

[31] Pouya Kamalinejad, Chinmaya Mahapatra, Zhengguo Sheng, Shahriar Mirabbasi, Victor CM Leung, and Yong Liang Guan. 2015. Wireless energy harvesting for the Internet of Things. *IEEE Communications Magazine* 53, 6 (2015), 102–108.

[32] Tom J Kazmierski, Leran Wang, Geoff V Merrett, Bashir M Al-Hashimi, and Mansour Aloufi. 2013. Fast design space exploration of vibration-based energy harvesting wireless sensors. *IEEE Sensors Journal* 13, 11 (2013), 4393–4401.

[33] Muhammad Kamran Khan, Muhammad Shiraz, Kayhan Zrar Ghafoor, Suleman Khan, Ali Safaa Sadiq, and Ghufran Ahmed. 2018. EE-MRP: Energy-efficient multistage routing protocol for wireless sensor networks. *Wireless Communications and Mobile Computing* 2018 (2018).

[34] Kimia Kiani, George Cui, Andrea Bunt, Joanna McGrenere, and Parmit K Chilana. 2019. Beyond" One-Size-Fits-All" Understanding the Diversity in How Software Newcomers Discover and Make Use of Help Resources. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM New York, NY, USA, 1–14.

[35] Daeyong Kim, Junick Ahn, Jun Shin, and Hojung Cha. 2021. Ray tracing-based light energy prediction for indoor batteryless sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 1 (2021), 1–27.

[36] Vito Kortbeek, Abu Bakar, Stefany Cruz, Kasim Sinan Yildirim, Przemysław Pawełczak, and Josiah Hester. 2020. BFree: Enabling Battery-Free Sensor Prototyping with Python. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4, Article 135 (Dec. 2020), 39 pages. https://doi.org/10.1145/3432191

[37] Vito Kortbeek, Abu Bakar, Stefany Cruz, Kasim Sinan Yildirim, Przemysław Pawełczak, and Josiah Hester. 2020. Bfree: Enabling battery-free sensor prototyping with python. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 4 (2020), 1–39.

[38] Christopher Kraemer, Amy Guo, Saad Ahmed, and Josiah Hester. 2022. Battery-Free MakeCode: Accessible Programming for Intermittent Computing. *Proc. UbiComp* 6, 1, Article 18 (mar 2022), 35 pages. https://doi.org/10.1145/3517236

[39] Christopher Kraemer, Amy Guo, Saad Ahmed, and Josiah Hester. 2022. Battery-free MakeCode: Accessible Programming for Intermittent Computing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 1 (2022), 1–35.

[40] Seulki Lee, Bashima Islam, Yubo Luo, and Shahriar Nirjon. 2019. Intermittent Learning: On-Device Machine Learning on Intermittently Powered System. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 4 (Dec. 2019), 141:1–141:30.

[41] Yichen Li, Tianxing Li, Ruchir A Patel, Xing-Dong Yang, and Xia Zhou. 2018. Self-powered gesture recognition with ambient light. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, 595–608.

[42] Qi Lin, Weitao Xu, Guohao Lan, Yesheng Cui, Hong Jia, Wen Hu, Mahbub Hassan, and Aruna Seneviratne. 2020. KEHKey: Kinetic Energy Harvester-based Authentication and Key Generation for Body Area Network. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–26.

[43] Linear. 2022. LTC3599. https://www.analog.com/en/products/ltc3588-1.html.

[44] Frank Linton and Hans-Peter Schaefer. 2000. Recommender systems for learning: Building user and expert models through long-term observation of application use. *User Modeling and User-Adapted Interaction* 10, 2 (2000), 181–208.

[45] Yuwen Lu, Chengzhi Zhang, Iris Zhang, and Toby Jia-Jun Li. 2022. Bridging the Gap between UX Practitioners' work practices and AI-enabled design support tools. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*. ACM New York, NY, USA, 1–7.

[46] Dong Ma, Guohao Lan, Mahbub Hassan, Wen Hu, Mushfika B Upama, Ashraf Uddin, and Moustafa Youssef. 2019. Solargest: Ubiquitous and battery-free gesture recognition using solar cells. In *The 25th Annual International Conference on Mobile Computing and Networking*. ACM, New York, NY, USA, 1–15.

[47] Kiwan Maeng, Alexei Colin, and Brandon Lucia. 2017. Alpaca: Intermittent Execution without Checkpoints. In *Proc. OOPSLA* (Oct. 22–27). ACM, Vancouver, BC, Canada, 96:1–96:30.

[48] Kiwan Maeng and Brandon Lucia. 2018. Adaptive dynamic checkpointing for safe efficient intermittent computing. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*. ACM, New York, NY, USA, 129–144.

[49] Kiwan Maeng and Brandon Lucia. 2020. Adaptive Low-Overhead Scheduling for Periodic and Reactive Intermittent Execution. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation* (London, UK) *(PLDI 2020)*. Association for Computing Machinery, New York, NY, USA, 1005–1021. https://doi.org/10.1145/3385412.3385998

[50] Damien Masson, Jo Vermeulen, George Fitzmaurice, and Justin Matejka. 2022. Supercharging Trial-and-Error for Learning Complex Software Applications. In *CHI Conference on Human Factors in Computing Systems*. ACM New York, NY, USA, 1–13.

[51] Inc Miro. 2023. Miro. 9.https://miro.com/[Accessed:(2019)].

[52] Jim Philippe Neussl, Rebecca Zheng, George Hanson, and Boyin Yang. 2019. TechBuddies: Engaging Students to Teach Retirees about Technology. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM New York, NY, USA, 1–6.

[53] Omer Onar and Alireza Khaligh. 2009. An Energy Harvesting Curriculum Developed And Offered At The Illinois Institute Of Technology. In *2009 Annual Conference & Exposition*. 14–184.

[54] Jung Wook Park, Alishan Hassan, Tingyu Cheng, Rosa Arriaga, and Gregory Abowd. 2021. A Simulation and Prototyping Toolkit for Airflow Energy Harvesting in Vehicles. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. ACM, New York, NY, USA, 588–589.

[55] Jung Wook Park, Sienna Xin Sun, Tingyu Cheng, Dong Whi Yoo, Jiawei Zhou, Youngwook Do, Gregory D Abowd, and Rosa I Arriaga. 2023. Exergy: A Toolkit to Simplify Creative Applications of Wind Energy Harvesting. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 1 (2023), 1–28.

[56] Benjamin Ransford, Jacob Sorber, and Kevin Fu. 2011. Mementos: System support for long-running computation on RFID-scale devices. In *Proceedings of the sixteenth international conference on Architectural support for programming languages and operating systems*. ACM, New York, NY, USA, 159–170.

[57] Mitchel Resnick, Brad Myers, Kumiyo Nakakoji, Ben Shneiderman, Randy Pausch, Ted Selker, and Mike Eisenberg. 2005. Design principles for tools to support creative thinking. (2005).

[58] Kimiko Ryokai, Peiqi Su, Eungchan Kim, and Bob Rollins. 2014. Energybugs: Energy harvesting wearables for children. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 1039–1048.

[59] Jeffrey S Saltz and Ivan Shamshurin. 2019. Exploring pair programming beyond computer science: a case study in its use in data science/data engineering. *International Journal of Higher Education and Sustainability* 2, 4 (2019), 265–278.

[60] Nurani Saoda, Md Fazlay Rabbi Masum Billah, and Bradford Campbell. 2021. Designing a General Purpose Development Platform for Energy-harvesting Applications. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 581–583.

[61] Raymond Scupin. 1997. The KJ method: A technique for analyzing data derived from Japanese ethnology. *Human organization* 56, 2 (1997), 233–237.

[62] Himanshu Sharma, Ahteshamul Haque, and Zainul Abdin Jaffery. 2018. Modeling and optimisation of a solar energy harvesting system for wireless sensor network nodes. *Journal of sensor and Actuator Networks* 7, 3 (2018), 40.

[63] Jonathan A Smith. 2015. Qualitative psychology: A practical guide to research methods. *Qualitative psychology* (2015), 1–312.

[64] Jacob Sorber, Alexander Kostadinov, Matthew Garber, Matthew Brennan, Mark D. Corner, and Emery D. Berger. 2007. Eon: A Language and Runtime System for Perpetual Systems. In *Proc. of The 5th Int'l ACM Conf. on Embedded Networked Sensor Systems (SenSys)*. 161–174.

[65] Naomi Stricker, Yingzhao Lian, Yuning Jiang, Colin N Jones, and Lothar Thiele. 2021. Joint energy management for distributed energy harvesting systems. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. ACM New York, NY, USA, 575–577.

[66] Vamsi Talla, Bryce Kellogg, Shyamnath Gollakota, and Joshua R. Smith. 2017. Battery-Free Cellphone. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 2 (2017), 25:1–25:20. https://doi.org/10.1145/3090090

[67] Shan-Yuan Teng, KD Wu, Jacqueline Chen, and Pedro Lopes. 2022. Prolonging VR Haptic Experiences by Harvesting Kinetic Energy from the User. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. ACM New York, NY, USA, 1–18.

[68] Texas Instruments Inc. 2013. BQ25570 Ultra Low power Harvester power Management IC with Boost Charger, and Nanopower Buck Converter. https://www.ti.com/lit/ds/symlink/bq25570.pdf. Last accessed: Apr. 25, 2020.

[69] Samyukta Venkat, Marshall Clyburn, and Bradford Campbell. 2020. Energy Harvesting Systems Need an Operating System Too. In *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*. ACM New York, NY, USA, 15–21.

[70] Markus Weninger, Elias Gander, and Hanspeter Mössenböck. 2021. Guided Exploration: A Method for Guiding Novice Users in Interactive Memory Monitoring Tools. *Proceedings of the ACM on Human-Computer Interaction* 5, EICS (2021), 1–34.

[71] Linda L Werner, Brian Hanks, and Charlie McDowell. 2004. Pair-programming helps female computer science students. *Journal on Educational Resources in Computing (JERIC)* 4, 1 (2004), 4–es.

[72] Laurie Williams and Robert R Kessler. 2003. *Pair programming illuminated*. Addison-Wesley Professional.

[73] Chengshuo Xia, Daxing Zhang, Witold Pedrycz, Kangqi Fan, and Yongxian Guo. 2019. Human Body Heat Based Thermoelectric Harvester with Ultra-Low Input Power Management System for Wireless Sensors Powering. *Energies* 12, 20 (2019), 3942.

[74] Cheuk-Wang Yau, Tyrone Tai-On Kwok, and Yu-Kwong Kwok. 2017. Energy Gatekeeper Architecture for Enabling Rapid Development of Energy-Harvesting Internet of Things. In *Proceedings of the Fifth ACM International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*. 43–45.

[75] Nur Yildirim, James McCann, and John Zimmerman. 2020. Digital Fabrication Tools at Work: Probing Professionals' Current Needs and Desired Futures. In *Proceedings of the 2020 CHI conference on human factors in computing systems*. ACM New York, NY, USA, 1–13.

[76] Jianping Zeng, Jongouk Choi, Xinwei Fu, Ajay Paddayuru Shreepathi, Dongyoon Lee, Changwoo Min, and Changhee Jung. 2021. ReplayCache: Enabling Volatile Cachesfor Energy Harvesting Systems. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*. ACM New York, NY, USA, 170–182.

[77] Dingtian Zhang, Jung Wook Park, Yang Zhang, Yuhui Zhao, Yiyang Wang, Yunzhi Li, Tanvi Bhagwat, Wen-Fang Chou, Xiaojia Jia, Bernard Kippelen, et al. 2020. OptoSense: Towards Ubiquitous Self-Powered Ambient Light Sensing Surfaces. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 3 (2020), 1–27.

[78] Daxing Zhang, Fan Yang, Tsutomu Shimotori, Kuang-Ching Wang, and Yong Huang. 2012. Performance evaluation of power management systems in microbial fuel cell-based energy harvesting applications for driving small electronic devices. *Journal of Power Sources* 217 (2012), 65–71.

[79] Yang Zhang, Yasha Iravantchi, Haojian Jin, Swarun Kumar, and Chris Harrison. 2019. Sozu: Self-powered radio tags for building-scale activity sensing. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, 973–985.

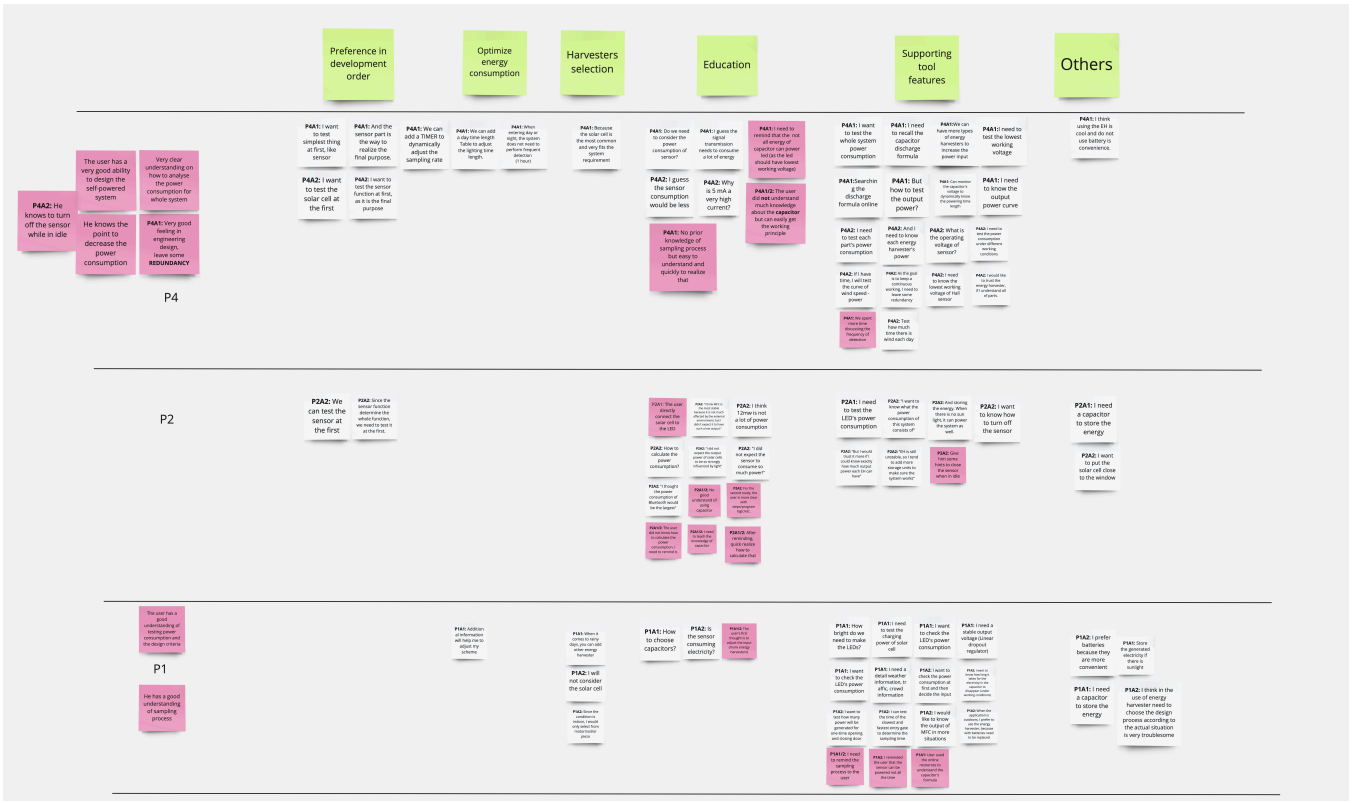

Figure 7: The thematic analysis example with affinity diagramming in the user study.

[80] Franz Zieris and Lutz Prechelt. 2020. Explaining pair programming session dynamics from knowledge gaps. In *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*. IEEE, IEEE, Long Beach, CA, USA, 421–432.

A DETAILED INFORMATION IN USER STUDY

A.1 Introduction of Energy Harvesting by the Expert

After presenting the main contents of the two-day mentored physical prototyping study, the expert describes the characteristics of each type of energy harvester since none of the novice developers involved in the study had experience developing energy harvesting systems. The expert only explained what kind of energy-to-electricity conversion the corresponding energy harvester accomplished (instead of the related detailed electrical characteristics, e.g., the voltage level, power, and so on) and the characteristics of the output energy (DC or AC). After confirming that each novice developer had a clear understanding of the characteristics of the provided energy harvesters and had no further questions, the main phase of experimentation began.

A.2 Affinity diagramming and Thematic Analysis for Observation Study and Interview

Both the observation study and interview study were analyzed by affinity diagramming for thematic analysis, which is the common way in HCI qualitative evaluation [45, 52, 75]. We transcribed the observation study and recorded the video and interview video. Two co-authors (including the expert) reviewed the video and transcripts via the Miro [51]. Each co-author textualized the key points of novice users' behavior, such as their questions, required assistance, and proposed tentative solutions, as well as the interview's main topics. Subsequently, two co-authors worked together to cluster the found topics and form the groups. Lastly, they collaboratively reviewed all topics and groups and summarized them in more specific and appropriate themes. Figure 7 presented our thematic analysis.