Abstract: IoT Based Human-Building Interaction

Our interactions with built environments are increasingly augmented with digital capabilities. Smart appliances coupled with Building Management Systems (BMS) promise to increase occupant comfort and to reduce energy consumption through automated control and personalized services. However, it is not clear how smart appliances, originally designed for smart homes, can be used in the context of non-residential buildings. For example: How can a user be given access to smart appliances in her vicinity? This simple question raises a number of issues in non-residential buildings: What are the smart appliances in the vicinity of a person (this requires building-wide metadata collection and maintenance)? How can a user access new smart appliances as she moves around without a cumbersome initialisation process, yet with security measures enforced? How to mediate between diverging smart appliances settings?

In this thesis, we study how Mark Weiser's original vision for ubiquitous computing can help us tackle such questions. Specifically, our contributions are the following:

(i) At the human-building interface, we design and implement a system for a building manager that enables building-wide visualization and control by transforming the physical building, with its structure, sensors and actuators, into a virtual reality system. For building occupants, we enable the ubiquitous use of appliances and sensors in vicinity by bridging existing off-the-shelf smart appliances to a common Bluetooth Low Energy (BLE) interface with room-level based authorization.

(ii) We explore different identification mechanisms to support metadata management. For existing BMS sensors and actuators, we introduce a crowd-sourced approach that incrementally builds up consistent metadata. For smart appliances in vicinity, we introduce acoustic based localization. We further design and implement a system that uses smartphone sensor based user feedback to automatically select appliance settings. We also mediate conflicts between users locally and building wide energy policies.

(iii) On a system level, we present an architecture for the decentralized integration of smart appliances into non-residential buildings that relies on user smartphones as opportunistic gateways and BLE for communication. We design and implement a distributed framework to evaluate BLE performance in such smartphone-peripheral systems. We perform a detailed evaluation of multiple smartphone models that shows that the native BLE stack fails to provide homogeneous abstractions for different implementations. We improve the default behavior, by the introduction of a dynamic, smartphone model dependent library, that adapts to the idiosyncrasies of specific BLE implementations.

Resumé: IoT baseret Menneske-Bygning Interaktion

Vores interaktioner med kunstigt konstruerede miljøer er i stigende grad supplerede med digitale muligheder. Smarte apparater kombineret med Building Management Systemer (BMS) lover at øge komforten og reducere energiforbruget gennem automatiseret kontrol og personlig service. Det er dog ikke klart, hvordan smarte apparater, som oprindeligt er designet til intelligente hjem, kan blive brugt i forbindelse med bygninger der ikke huser beboere. For eksempel: Hvordan kan en bruger få adgang til smarte apparater i hendes nærhed? Denne enkle spørgsmål rejser en række spørgsmål i ikke-beboelsesejendomme: Hvilke smarte apparater er i nærheden af en person (dette kræver bygning-dækkende indsamling og vedligeholdelse af metadata)? Hvordan kan en bruger få adgang til nye smarte apparater, mens hun bevæger sig rundt uden en besværlig initialiseringsproces uden at give afkald på håndhævelse af sikkerhedsforanstaltninger? Hvordan kan der mægles mellem forskellige brugeres præferencer i i forbindelse med smarte apparaters indstillinger?

I denne afhandling undersøger vi hvordan Mark Weisers oprindelige vision for ubiquitous computing kan hjælpe os til at tackle sådanne spørgsmål. Konkret er vores bidrag følgende:

(i) Ved menneske-bygningen interface, designer og implementerer vi et system til en bygning manager, der gør det muligt at visualisere og kontrollere hele bygningen ved at genskabe den fysiske bygning – med opbygning, sensorer og aktuatorer – i et virtual reality-system. For bygningens beboere, muliggør vi den allestedsnærværende brug af apparater og sensorer ved at bridge eksisterende off-the-shelf smarte apparater til et fælles Bluetooth Low Energy (BLE) interface med authorization på værelsets-niveau.

(ii) Vi udforsker forskellige mekanismer til identifikation for at støtte metadata vedligeholdelse. For eksisterende BMS sensorer og aktuatorer, introducerer vi en crowdsourcing tilgang, der trinvist opbygger konsekvente metadata. Fornære smarte apparater, indfører vi fysisk akustisk-baseret lokalisering. Vi yderligere designer og implementerer et system, der bruger brugerfeedback fra sensorerne i en smartphone til automatisk at vælge apparatets indstillinger. Vi formidler også konflikter mellem brugere lokalt og energipolitik på bygningsniveau.

(iii) På et system niveau præsenterer vi en arkitektur for den decentrale integration af smarte apparater til ikke-beboelsesejendomme, der bygger på brugernes smartphones som opportunistiske gateways og BLE for kommunikation. Vi designer og implementerer et distribueret framework for at evaluere BLE præstationer i sådanne smartphone-peripheral systemer. Vi udfører en detaljeret evaluering af flere smartphone modeller, der viser, at den Androids standard BLE stak ikke giver homogene abstraktioner for forskellige implementeringer. Vi forbedrer standard adfærden, ved at tilbyde et dynamisk, smartphone model-afhængigt bibliotek, der tilpasser sig specifikke BLE implementeringers idiosynkrasier.

To Vanesa,

Thanks for booking that flight in November.
Thanks for moving from Madrid to Copenhagen.
Thanks for staying in Copenhagen six months alone.
Thanks for bearing and supporting me the last three years.

Contents

Acknowledgements

The three years of my PhD have been an exciting ride and this thesis would not have been possible without the many outstanding people that I have met, discussed my research with, and collaborated with on various projects.

"The beginning is the most important part of the work—Plato". Our work on off-grid microgrids together with Nik Gawinowski, Sebastian Büttrich and Philippe Bonnet for my MSc. thesis was my first proper research project. I still have good memories of the days Nik and I spent in our blacked out, improvised laboratory, just outside the Software and Systems department, the trip with Sebastian to Zambia and my first academic paper presentation at GHTC 13' in San José. Thanks for this great time that directly led me into my PhD. Sebastian has continuously helped me throughout my PhD with his experience, his knowledge and also practically by providing me with tools and equipment out of PITLabs vast stack. Nik has been a great friend since our joined MSc. thesis.

Philippe Bonnet has been a brilliant advisor, not only for my MSc., but even more during my PhD. He has always found the right trade-off between giving me the freedom to explore different research directions, making my own mistakes and providing guidance and direction to align my work towards the bigger picture. His big academic network has enabled me to get in contact with some of the most recognized researchers in my field. His broad system knowledge of sensor networks and databases has been a unique asset during my PhD. Besides that, he always supported me on various trips to conferences and for the puropose of collaborations.

I am very happy that I could meet David Culler and Randy Katz during my stay abroad in their research group at UC Berkeley and discuss my work with them. The whole group of faculty and students of SDB has been a blast, and I have become friends with many of them. Thanks, Kaifei Chen, Gabe Fiero, Michael Anderson, Elizabeth Kim and Arka Bhattacharya for making my stay abroad so great. Thanks to Albert Goto for all the nice chats, for always offering to pick me up and drive me to the airport, for all the provided candy and late dinners and for all the other small things. The weekly hangout meetings with Kaifei Chen have been part for most of my PhD. Our discussions have helped me tremendously to see problems from another angle and Kaifei always has had a pointer to at least three related papers.

At the IT University of Copenhagen, I have had the pleasure to spend my days in the office with a bunch of great people. Thanks to Javier González, Aslak Johansen, Matias Bjørling, Niv Dayan, Mohammed Aljarrah, Ivan Picoli, Carla Villegas, Andrés Faiña and all the people I have forgotten to list here. Javier and I came both to Denmark six years ago, became friends and ended both up as Philippe's PhD students. Aslak has been a great help on embedded systems, Danish language (I am still not any better) and friend for several years. Thanks to the exceptional people from our administration, especially Freja Krab Koed Eriksen and Christina Rasmussen for always helping me with my questions regarding ECTS and travel reimbursements and the PhD school for being so generous with all my funding requests.

And of course, thanks to my family. Thanks to my parents for supporting me to start my Master studies in Denmark and continue with my PhD and to Simon and David for being the best brothers one could wish for. Thanks to all my friends that I have met on the way to here. I promise to be more frequently available for the occasional friday beers again—at least I will try. Finally, thanks to Vanesa for being always there for me during this journey. During the easier and more difficult moments. You have helped me to enjoy both of them.

Chapter 1

Introduction

1.1 Context

Twenty-five years ago, Mark Weiser proposed the original vision of ubiquitous computing in Scientific American [44]. He predicted that our environment would soon consist of hundreds of connected computers, and that we would interact with such smart environment using ubiquitous displays in various sizes. A few years later, in the late 1990s, researchers of the networking community postulated that sensors and actuators equipped with computation and communication capabilities would become widely deployed and require new networking and application structures, distinct from traditional computer networking [15].

Today, the work of Weiser and others at Xerox PARC has led to the research fields of ubiquitous and pervasive computing, and sensor networks has emerged from computer networking as an independent research field. Likewise, in our daily lives, the original visions of ubiquitous computing and sensor networks are about to become a reality. Environmental sensors, smart devices and appliances such as fitness trackers, thermostats, light bulbs, power plugs and locks, equipped with short range radios are available in retail stores. Smartphones have become a pervasive tool for most of us, through which we can interact with our environment. They provide an adjustable interface in form of a touchscreen, various local and global networking protocols and radios (e.g., Bluetooth, 802.11, 3G, LTE) and gigabytes of personal storage.

IoT. The catch word, which like no other term, captures this development beyond academia is *IoT, the Internet of Things.* In 1999, Kevin Ashton coined the term "Internet of Things" in a sales pitch to Procter & Gamble [2]. Today, IoT is the subject of mass media and economic politics. When we started our work in 2013, IoT was still not widely accepted as a serious research field by the academic community, but merely considered a marketing term by Cisco, Google et al. Over the course of the last three years, this has changed dramatically. University based IoT labs and projects rapidly emerge around the globe (e.g., at Carnegie Mellon University [34], Stanford [39], Bosch/ETHZ [5]), and academic, IoT centered conferences have been established by the sensor network and pervasive computing community (e.g., [23]).

Many commercial off-the-shelf IoT appliances aim to improve our personal comfort and our energy consumption by enhancing ordinary appliances with computation and networking connectivity. Examples are smart lighting, smart HVAC (using smart thermostats) or smart power plugs (power monitoring and switching). This digitalization enables new capabilities like predictive analysis or automated adaptive control.

Non-residential Buildings. Buildings are a prime platform for such IoT deployments because of their high share of energy consumption and strong human involvement. The average person spends around 90% of her time inside a building [41]. In Europe and in the U.S., buildings consume nearly half of the primary energy, $\sim 75\%$ of electrical energy and are the cause for $\sim 36\%$ of CO_2 emissions [16, 40]. The two main building categories are residential and non-residential. A large portion of primary energy consumption (19%) can be traced down to non-residential buildings [40]. Besides their impact on energy consumption and the availability of off-the-shelf smart appliances, this thesis focuses on non-residential buildings for the following reasons:

(i) Non-residential buildings often contain already automated building control in form of Building Management Systems (BMS). These systems are vertically integrated stovepipes. The long building life-cycles of 50–70 years and long replacement times of existing BMS infrastructure ([26]) make it necessary to deal with these systems for the coming years. Computer science has been naturally predesignated to break open these stovepipes and to provide new abstractions that transform existing BMS into an operating system aligned architecture, a *Building Operating System*. Much pioneering work has been done in David Culler's and Randy Katz's Local and Software Defined Building (SDB) group at UC Berkeley (e.g., [8, 27, 9]). Today, this work forms some of the basis for a variety of BMS based research. Researchers have built applications on top of BMS to improve energy consumption ([3]), find faults in BMS control loops ([33]) or improve the comfort of building occupants ([14]). This thesis, during several occasions, builds on and extends UC Berkeley's work.

(ii) Non-residential buildings provide an appealing combination of shared spaces and a dynamic set of occupants. Further, a non-residential building usually consists of multiple functional units and the building ownership is not aligned with its use. Such setting has different requirements for access and authorization and requires the mediation between conflicts of users locally and between local comfort requirements and global, building wide goals. The high human density makes it possible, and often necessary, to develop human centric systems, to take humans into the loop and to crowd-source the knowledge of individuals.

Our overall thesis for the system design of IoT based interaction of humans with the non-residential built environment is the following:

Thesis

A seamless augmentation of physical human-building interaction with digital capabilities through IoT off-the-shelf systems can be derived from Mark Weiser's vision of ubiquitous computing.

1.2 Problems

This thesis addresses two classes of problems: (1) Using off-the-shelf IoT systems for human-building interaction and (2) aligning this interaction with Mark Weiser's vision of ubiquitous computing.

1.2.1 Problems with Off-the-Shelf IoT Devices

The reality of available off-the-shelf IoT systems in the context of human-building interaction reveals several misalignments and problems:

Stove Pipes. Bad design decisions, that were made in building management systems twenty years ago, are being repeated. Instead of systems that are compatible across different manufacturers (with well defined interfaces and by relying on open protocols), IoT devices are characterized by vertical silos, manufacturer stove pipes and incompatibility. IoT in its current state is thus appropriately described as *"The CompuServe of Things"* [46, 31].[1]

Centralized Architecture. Instead of following a decentralized architecture, many IoT systems are centralized and cloud-based. Practically, this means that the data and control flow between IoT devices and the user's input device (e.g., her smartphone) occurs through the manufacturer cloud and not directly between devices. Such centralization has serious implications for fault-tolerance, security, privacy and scalability.

First, it constitutes a single point of failure. Users do not have much control over their own devices and their functioning depends on the manufacturer cloud services. There have been cases where devices on people's premises have been unavailable due to ordinary server outages (e.g., Nest thermostat outages [13]), or even made permanently unusable by the manufacturer (e.g., in April 2016, Nest announced that it would disable their Revolv smart home hub [12]; Philips only reversed the deactivation of previously compatible, 3rd party light bulbs after massive user protests [22]).

Second, a centralized architecture directly exposes appliances to the LAN or the Internet. A potential attacker can thus take over control over appliances remotely, even when the owner is not present. This problem is more severe considering the high number of security flaws and slow software fixes for IoT devices. This high number of security flaws made Kaspersky Lab refer to IoT as the *"Internet of Crappy Things"* [25].

Third, IoT devices invisibly communicate with the manufacturer server, possibly leaking data that users might not want to share. The centralization

[1]Compuserve provided, different siloed online services (e.g., forums, news, messaging) on its popular CompuServe Information Service to their customers during the 80s.

allows the combination and aggregation of otherwise non-critical user data to privacy critical data (e.g., through nonintrusive load monitoring techniques, or by combining sensor data with the web browsing preferences of a user).

And last, the growing number of connected sensors and actuators rises questions of scalability in a centralized architecture. In contrast to that, the networking community often approached scalability issues with the concept of local and P2P networks (e.g., [15, 36]).

No Intuitive Interaction. The current way in which we interact with IoT appliances is not intuitive for several reasons: For end users, centralized IoT silos have led to different types of IoT devices requiring different smartphone applications with different authentication and authorization mechanisms. User authentication and authorization are mostly password based. Recent academic systems are not better: In order to identify appliances in the physical world, users might need to describe the appliance in cumbersome ways following a strict syntax (see e.g., [8]). Other solutions require a high infrastructure overhead to provide identification through tags (e.g., using BLE [47], RFID [43], QR-codes [42] or infrared [48]).

Existing smart appliances and IoT systems are designed to be used by a single family in the restricted environment of a home. Such a coupling between physical locality and unit of administration and usage does not exist in non-residential buildings. The Wifi or LAN that implicitely constitutes a natural boundary for smart appliance accesses at home, might cover one or several non-residential buildings shared by various organisations.

The current interfaces of building managers are not better: Interfacing with a building requires programming, technical or expert knowledge, specific to the building and BMS. Based on our own practical experience at our campus building in Copenhagen and several buildings at the UC Berkeley campus, this knowledge is not always present. It might even require external consultants to make use of the systems.[2] Even if expertise is available, we noticed some reluctance towards implementing changes on the physical building. The reason is that such changes directly affect the building occupants, and their comfort and the building manager is held responsible.

[2]E.g., in our campus building, a change of the schedule for the automatic windows requires external consultants.

1.2.2 Mark Weiser's Core Issues

Let us briefly go back to Mark Weiser's original vision [44], and how it applies to human-building interaction and IoT systems in 2016.

Core Concepts and Issues. In Weiser's vision, computational elements, embedded in our environment, become as ubiquitous as literacy technology (e.g., the writing on a street sign). Because of their ubiquity, we do not even consciously notice their presence, but make use of them without effort. We interface with such environment using ubiquitous displays (tablets), that are as omnipresent as regular notes and paper. He used the term "embodied virtuality" to describe this phenomenon, which he considered diametrically opposed to the notion of Virtual Reality (VR). VR creates a simulation of the real or some imaginary world and is therefore suitable for the exploration of otherwise inaccessible regions [44].

According to Weiser, four core issues need to be solved to achieve ubiquitous computing:

- **Location Awareness.** Computers must be aware of their location.
- **Scalability.** Systems need to scale to hundreds of computers in a room.
- **Ubiquitous Networking.** Machines should be connected in an ubiquitous, local network, but also connected on a wider scale.
- **Privacy.** Privacy is a key social issue. Embedded devices might sense our movement, cameras might record us. Thus, users need to be protected from their superiors, marketing firms and their government.

Reality Check. Today, computers have clearly vanished into the background of non-residential buildings. Embedded sensors collect environmental data, while we are surrounded by multiple devices that contain a computational unit (e.g., TV, kitchen appliances, smart thermostats and lights). Such smart environment can endow building occupants with individualized and adaptive comfort according to personal preferences. Occupant comfort research has seen a shift from a centralized, static, model based building control (PMV model) to a user centered, adaptive control strategy. The adaptive model assumes that people are able to adapt themselves to different temperatures dependent on the season and outside temperature. An ideal adaptive building has natural ventilation, personal occupant control and a high energy efficiency [10].

The privacy issue of ubiquitous systems has been continuously emphasized over the years. E.g., Langheinrich discusses several privacy problems in the context of ubiquitous systems in non-residential buildings. Problems arise due to the ubiquity and invisibility of the computing infrastructure. He draws a scenario of a possible fulfillment of the "...frightening vision of an Orwellian nightmare-come-true". Some of the proposed countermeasures are limiting the number of communication hops any message can travel and introducing the concepts of proximity and locality [29]. In [28], Langheinrich further describes the idea of "privacy borders" and that the crossing of these borders needs to be prevented by designers of ubiquitous systems.

The ubiquitous tablets Weiser and others envisioned have found a partial realization in our personal smart devices (our smartphones and tablets).[3] People use these devices naturally to interact with others and with their environment. Manufacturers usually rely on smartphone apps as an interface to their smart appliances.

Virtual reality has become a hot topic for industry and academia (e.g., Google, Facebook, Microsoft, Samsung, HTC are all working on their respective implementations), and is applied with success in the gaming sector.[4] Related to human-building interaction, we believe VR can enable building-wide insight and control and help to scope with the output of a huge number of different sensors. Augmented reality might be a middle way between a fully virtual system and a purely embodied virtuality.

Overall, it becomes clear, that Weiser's twenty-five year old vision is more relevant now than ever. For IoT based human-building interaction to be a success, we believe that his core issues must be addressed in a non-residential building context: (i) we need to achieve location awareness, (ii) systems need to be scalable, (iii) local, ubiquitous networks need to be created and (iv) privacy of users needs to be respected.

[3]In July 2016, Apple announced it had sold over one billion of iPhones in less than a decade [1].

[4]E.g., In April 2016, Minecraft was released for Facebook's Oculus system [35].

1.3 Approach

The overall approach of this PhD is system oriented and rooted in Experimental Computer Science [11]. Over the course of this PhD, we have been iteratively defining problems, developing hypotheses, designing systems, implementing prototypes of these systems and evaluating their performance qualitatively and quantitatively to prove or disprove our hypotheses. In many cases, the system implementations itself have opened up new problems that were not visible a priori, and design decisions that seemed initially obvious, proofed to be inadequate.

1.4 Contributions

We derived specific requirements by adapting human-building interaction from Mark Weiser's original vision. These requirements have guided us during our exploration of the design space. We then designed and implemented systems that explore specific parts in this design space.

1.4.1 Requirements

Our requirements for systems of IoT-based human-building interactions are the following:

(1) User Focused and Adaptive Model. Smart appliances should play a crucial part in achieving an adaptive environment that is guided by user comfort preferences: (i) Smart appliances can provide individualization (e.g., by adjusting to a person's comfort preferences) and a more fine grained adaption to the individual needs (e.g., by keeping the brightness at some preferred level, independent of outside influences). (ii) They support optimizing a building for energy efficiency through (a), automatic adaption to the current environment state (e.g., by basing the heating and lighting setpoint on occupancy or by lowering the brightness, while there is sufficient sun light) and (b), through raising awareness among occupants (e.g., by providing insight on a person's energy consumption combined with building wide score boards).

(2) Ubiquitous Access. Following Weiser, we argue that access (authentication and authorization) to these smart appliances in non-residential buildings must not be an additional burden for the user. Interacting with smart appliances should not be more cumbersome than interacting with ordinary appliances. This interaction also requires identification strategies in the physical and digital world. Such identification should establish a link between the user, her location, a physical smart appliance and a digital control point.

(3) Transparency. Users should always understand the current environment state and be able to take control of it. Automated adaption needs to be initiated by the user and appliance functioning must be transparent to her as e.g., it has been shown in [32]. The device user/owner, and not the manufacturer, should be in full control over her appliances.

(4) Building Wide Interface. The building manager's interface should not require deep technical or expert knowledge. It should include an intuitive way of visualisation and allow to explore adequate building configurations playfully, without harming the physical building and its occupants. It must support overall building strategies and schedules, but the focus should be on the exploration and supervision of the building state (e.g., the choice of the heat and light setting in different rooms should be left to the current occupants).

(5) Smartphone Centered. Due to the universality of modern smartphones, they should play a central part in IoT-based human-building interaction. Smartphones enable the storage of individual preferences and are a control point for data-flow between local infrastructure and the cloud, while providing through their adjustable touchscreen one of the most flexible and accepted user interfaces. An interface, that comes close to Weiser's vision of ubiquitous tablets. Their increasing number of sensors and ongoing extension to smart watches and other smart wearables helps to measure comfort related variables close to the affected individual person.

(6) Privacy Aware. IoT based human-building interaction should respect the individual privacy requirements. In practice, e.g., restrict the use of the cloud to necessary use cases (e.g., due to computational requirements or application requirements). A non-residential environment poses different parties with different interests (e.g., the building occupant and the building owner, an employee and a manager). Such milieu makes the compliance with privacy borders more important than in a residential environment.

(7) Decentralized and Local Networks. If the goal is to achieve ubiquitous networks, then the requirement must be that IoT systems are compatible with each other and use well defined interfaces. The trade-off between computation locally and in the cloud must be taken with care. To achieve scalability and fault-tolerance, we believe that devices should communicate locally when possible. The cloud should only be used when necessary. Some computation might require the cloud due to its computational intensity (e.g., Computer Vision applications), or due to application requirements (e.g., data aggregation). IoT users must be able to switch their cloud-provider like they are able to switch their electric utility or Internet provider.

1.4.2 Design Space

We split the design space for systems supporting human-building interaction into three dimensions: (i) Human-building interfaces, (ii) Means of identification and (iii) System architecture. We now briefly discuss each dimension:[5]

Human-Building Interface. We differentiate human-building interaction in terms of: (i) The scope of interaction: In a local interaction, building users interact with appliances in their surroundings (e.g., [48, 47]). Building wide interaction is e.g., done in a BMS, where a building manager is in control for the whole building (examples of academic systems are [45, 9, 37]).

(ii) Systems with a strong manual component (e.g., [14]) and systems with a high grade of automation (e.g., HVAC is run based on schedule).

In this thesis, we choose to explore human-building interaction from a building wide and a local perspective. For building wide interaction, we investigated the following questions: How can we support the building manager with a tool that gives a visual overview over the current state of the building, opposed to columnized streams of time series data? How can we visualize the effect of different configurations prior to deploying them on the actual building?

For the building users, we investigated the following questions: How can we enable end-users to ubiquitously use the smart infrastructure in their building? How can conflicts between different user preferences be mediated? Are there more intuitive ways than current ways of interaction?

[5]More details can be found in the respective later chapters of this thesis.

Means of Identification. We are all able to identify a physical light switch in a room, and the electrical wiring ensures that this switch toggles the light. For smart appliances, this physical connector artefact is lost. IoT based human-building interaction thus poses different levels of identification:

(i) Identification of appliances in the physical world. This identification process can take different forms, from manual to automated. Identification might be supported by some form of localization (e.g., [30]), by computer vision based approaches (e.g., [7]), or automated through personal sensor feedback. (ii) Identification of appliances (and their sensor and actuator points) digitally, e.g., by means of metadata (e.g., [8]). The required metadata can be either based on expert (e.g., [4]) input or crowd-sourced from ordinary users. It can be derived from either point values or point names (e.g., [38]). The process can be manual, semi-automatic or fully automatic.

In this thesis, we explored the following questions: (1) How can users identify smart appliances in the physical world? (2) How can digital setpoints of appliances be identified, when metadata of IoT devices or BMS metadata are inconsistent? (3) How can we dynamically identify the logical human-appliance relations? This means, how can we identify appliances like smart heating or lighting based on their effect on the individual.

System Architecture. The underlying system architecture can either be fully centralized, decentralized or a trade-off between both. Likewise, we might have some global network and a local network of connected devices.

This thesis explores a decentralized system with local networks and localized algorithms that adhere to global objectives. We have chosen to focus on Bluetooth Low Energy (BLE) as networking protocol in this thesis, because: (i) BLE is available on all modern smartphones without modification. (ii) It allows direct smartphone-to-IoT device connectivity without requiring a gateway. (iii) Compared to 802.11, it provides lower power consumption with the design goal of an exchange of small amounts of data. This is a perfect fit for IoT. Specifically, the question we address is: Can the existing stovepipes of IoT appliances be broken open and centralized architectures be replaced by a decentralized system in where localized algorithms follow a global objective like energy efficiency?

1.4.3 Systems

The systems we designed and implemented cover three areas: (i) Novel Human-Building interfaces for building managers and users, (ii) Identification (through BMS metadata, physical and logical appliance-human relation) and (iii) A decentralized, user-focused BMS, based on off-the-shelf appliances and the coupling of authorization and use through physical locality:

i In *BUSICO 3D*, we developed a system that integrates building simulation and control with a virtual 3D environment, merging the physical and the virtual world. A building manager is able to browse through time series data by means of a visualization of the data being projected in real-time on a 3D model of the actual building. The effects of different control strategies and schedules can be simulated before they are applied on the physical building. We implemented BUSICO 3D using a modern game engine in combination with an existing Building Information Model (BIM) of the building, facilitating a semi-automated deployment of our system.

ii During our implementation of BUSICO 3D, it became clear that metadata of IoT devices and BMS sensors and actuators is crucial for any application that builds on top. BMS metadata is often inconsistent and incomplete. No common naming schema is followed [4]. In *Babel*, we therefore developed a method to identify BMS-connected appliances through crowd-sourcing the building occupants. Our system incrementally builds up the metadata that is necessary for identification.

iii After we had dealt with the existing naming and interface abstractions, we came back to one of the early visions of sensor networks: Decentralized systems, with localized algorithms that adhere to a global objective [15, 24]. In *BLEoT*, we developed a system for the integration of off-the-shelf smart-appliances into a non-residential building, using Bluetooth Low Energy (BLE). We align use and authorization using acoustic based localization on a room level granularity. Such decentralized system can run localized algorithms that still adhere to some global, building wide goal like energy efficiency.

Based on our experiences in BLEoT with the BLE stack of current mobile operating systems, we decided to investigate it closer. In *BLEva*, we developed a distributed benchmarking framework for BLE in smartphone-peripheral systems. We then performed a detailed evaluation on nine different smartphone models. Based on our results, we proposed model dependent abstractions opposed to the native Android implementation and implemented a prototype.

When implementing BLEoT and Babel, we discovered that room level localization and consistent metadata are often not descriptive enough. It is hard to represent logical human-appliance relations in metadata. We would like to answer queries like: *Increase the brightness of the light that affects me most at my current location.* The included paper *"A Practical Model for Human-Smart Appliances Interaction"* gives an answer to this problem. Like Babel, it also puts the human in the loop. It uses the personal sensors of a user (e.g., smartphone, smartwatch) and human input to identify logical relations between a user and an appliance.

1.5 Structure of the Thesis

This thesis is composed of a collection of five research papers produced during my three year PhD. I am the main author of all these papers.

At the beginning of my PhD, I spent six months in David Culler's and Randy Katz's Software Defined Building (SDB) group at UC Berkeley. This stay has been an essential part of my PhD. I have continued to work with students of SDB for three years. In particular my collaboration with Kaifei Chen has been productive. Kaifei is a co-author on three papers of this thesis. I am also a co-author on three additional papers ([6, 7] and one paper currently in submission), for which Kaifei is the main author. This work is not included in my thesis and will be part of Kaifei's thesis. In these other papers, we approach human-building interaction from a different perspective, using Computer Vision (CV). Our premise in that work is: *What you see is what you interact with.* That work thus provides an intuitive, computer vision based interface for building occupants. Users are able to interface with smart appliances naturally, by using their smartphone camera. Opposed to that, my work utilizes localization awareness and smartphone based sensor feedback to identify appliances.

Besides my collaboration with UC Berkeley, I have been co-supervising several student projects during my PhD. This process has helped me to explore more of my ideas and areas of the design space. A part of the outcome of this work has also found its way into some of the attached papers (e.g., [21, 19]).

The papers included in this thesis are structured into three parts. In the first part, we present our work on developing a new interface for building managers: *"BUSICO 3D—Building Simulation and Control in Unity 3D"*, which has been presented in form of a demo at SenSys'14, at the OpenBAS'14 workshop at CMU and at the U.S. Department of Energy's Building Technologies Office Peer Review 2014 [18].

The middle part of this thesis discusses BMS metadata and how we incrementally crowd-source this data in our system. The included paper *"Crowd-sourced BMS Point Matching and Metadata Maintenance with Babel"* has been presented at PerCom'16 workshops [20] and in form of a demo at BuildSys'15 [17] (demo paper not included in this thesis).

In the last part, we introduce our decentralized, BLE based system of IoT devices for non-residential buildings *"Leveraging Physical Locality to Integrate Smart Appliances in Non-Residential Buildings with Ultrasound and Bluetooth Low Energy"*, which has been presented at IoTDI'16 and at the Software Defined Buildings Winter Retreat 2016 [21]. We continue with *BLEva*, our benchmarking framework and detailed evaluation of BLE on different smartphone models *"Evaluating and Improving Bluetooth Low Energy Performance in the Wild"*.

We then extend the physical locality of BLEoT to a logical human-appliance relation with *"A Practical Model for Human-Smart Appliances Interaction"*. We finish with an overall conclusion and an outlook on future work.

References

[1] Aaron Tilley (Forbes). Apple Confirms It Has Now Sold One Billion iPhones. `http://www.forbes.com/sites/aarontilley/2016/07/27/apple-confirms-it-has-now-sold-one-billion-iphones`. 2016.

[2] K. Ashton. That 'Internet of Things' Thing. *RFiD Journal*, 22(7):97–114, 2009.

[3] B. Balaji, J. Xu, A. Nwokafor, R. Gupta, and Y. Agarwal. Sentinel: Occupancy based HVAC actuation using existing WiFi infrastructure within commercial buildings. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems.* ACM, 2013, p. 17.

[4] A. A. Bhattacharya, D. Hong, D. Culler, J. Ortiz, K. Whitehouse, and E. Wu. Automated metadata construction to support portable building applications. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments.* ACM, 2015, pp. 3–12.

[5] Bosch, ETHZ, University of St. Gallen. Bosch IoT Lab. `http://www.iot-lab.ch/`.

[6] K. Chen, J. Kolb, J. Fürst, D. Hong, D. E. Culler, and R. H. Katz. Intuitive Appliance Identification using Image Matching in Smart Buildings. 2015.

[7] K. Chen, J. Kolb, J. Fürst, D. Hong, and R. H. Katz. Poster Abstract: Intuitive Appliance Identification using Image Matching in Smart Buildings. In *Proceedings of the 2nd acm international conference on embedded systems for energy-efficient built environments.* ACM, 2015, pp. 103–104.

[8] S. Dawson-Haggerty, X. Jiang, G. Tolle, J. Ortiz, and D. Culler. sMAP: a simple measurement and actuation profile for physical information. In *SenSys'10.* ACM, 2010, pp. 197–210.

[9] S. Dawson-Haggerty, A. Krioukov, J. Taneja, S. Karandikar, G. Fierro, N. Kitaev, and D. Culler. BOSS: building operating system services. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, 2013, pp. 443–457.

[10] R. J. De Dear, G. S. Brager, J. Reardon, F. Nicol, et al. Developing an adaptive model of thermal comfort and preference. *ASHRAE transactions*, 104:145, 1998.

[11] P. J. Denning. ACM President's Letter: What is experimental computer science? *Communications of the ACM*, 23(10):543–544, 1980.

[12] Electronic Frontier Foundation. Nest Reminds Customers That Ownership Isn't What It Used to Be. `https://www.eff.org/deeplinks/2016/04/nest-reminds-customers-ownership-isnt-what-it-used-be`. 2016.

[13] Engadget. Nest outage cuts remote users off from Dropcams, thermostats. `https://www.engadget.com/2015/09/07/nest-outage/`. 2015.

[14] V. L. Erickson and A. E. Cerpa. Thermovote: participatory sensing for efficient building hvac conditioning. In *BuildSys'12*. ACM, 2012.

[15] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar. Next century challenges: Scalable coordination in sensor networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. ACM, 1999, pp. 263–270.

[16] Eurostat. Energy Statistics. `http://ec.europa.eu/eurostat/web/energy`. 2015.

[17] J. Fürst, K. Chen, R. H. Katz, and P. Bonnet. Demo Abstract: Human-in-the-loop BMS Point Matching and Metadata Labeling with Babel. In *Proceedings of the 2nd ACM Internation*. ACM, 2015, pp. 101–102.

[18] J. Fürst, G. Fierro, P. Bonnet, and D. E. Culler. BUSICO 3D: Building Simulation and Control in Unity 3D. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*. ACM, 2014, pp. 326–327.

[19] J. Fürst, A. Fruergaard, M. Høvinghof Johannesen, and P. Bonnet. A Practical Model for Human-Smart Appliances Interaction. Tech. rep. IT University of Copenhagen, 2016.

[20] J. Fürst, K. Chen, R. H. Katz, P. Bonnet, et al. Crowd-sourced BMS point matching and metadata maintenance with Babel. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. IEEE, 2016, pp. 1–6.

[21] J. Fürst, K. Chen, M. Aljarrah, P. Bonnet, et al. Leveraging Physical Locality to Integrate Smart Appliances in Non-Residential Buildings with Ultrasound and Bluetooth Low Energy. In *IoTDI'16*. IEEE, 2016.

[22] Home Assistant. Philips Hue blocks 3rd party lights. `https://home-assistant.io/blog/2015/12/12/philips-hue-blocks-3rd-party-bulbs/`. 2015.

[23] IEEE. International Conference on Internet-of-Things Design and Implementation (IoTDI 2016). `http://conferences.computer.org/IoTDI/`. 2016.

[24] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*. ACM, 2000, pp. 56–67.

[25] Kaspersky Lab. Internet of Crappy Things. `https://blog.kaspersky.com/internet-of-crappy-things/7667/`. 2015.

[26] M. M. Khasreen, P. F. Banfill, and G. F. Menzies. Life-cycle assessment and the environmental impact of buildings: a review. *Sustainability*, 1(3):674–701, 2009.

[27] A. Krioukov, G. Fierro, N. Kitaev, and D. Culler. Building Application Stack (BAS). In *BuildSys'12*. ACM, 2012, pp. 72–79.

[28] M. Langheinrich. A privacy awareness system for ubiquitous computing environments. In, *UbiComp 2002: Ubiquitous Computing*, pp. 237–245, 2002.

[29] M. Langheinrich. Privacy by designprinciples of privacy-aware ubiquitous systems. In *Ubicomp 2001: Ubiquitous Computing*. Springer, 2001, pp. 273–291.

[30] J. Lifton, M. Mittal, M. Lapinski, and J. A. Paradiso. Tricorder: A mobile sensor network browser. In *Proceedings of the ACM CHI 2007 Conference-Mobile Spatial Interaction Workshop*, 2007.

[31] A. McEwan. Risking a Compuserve of Things. `http://mcqn.com/posts/wuthering-bytes-slides-risking-a-compuserve-of-things/`. 2013.

[32] M. Mozer. Lessons from an adaptive house. PhD thesis. Architectural Engineering, 2004.

[33] B. Narayanaswamy, B. Balaji, R. Gupta, and Y. Agarwal. Data driven investigation of faults in HVAC systems with model, cluster and compare (MCC). In *BuildSys'14*. ACM, 2014, pp. 50–59.

[34] C. M. U. News. CMU Leads Google Expedition To Create Technology for Internet of Things. `http://www.cmu.edu/news/stories/archives/2015/july/google-internet-of-things.html`. 2015.

[35] Oculus VR. Minecraft Now Available on Oculus for Gear VR. `https://www3.oculus.com/en-us/blog/minecraft-now-available-on-oculus-for-gear-vr/`. 2016.

[36] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. *A scalable content-addressable network*. Vol. 31(4). ACM, 2001.

[37] A. Rowe, M. E. Berges, G. Bhatia, E. Goldman, R. Rajkumar, J. H. Garrett Jr, J. M. Moura, and L. Soibelman. Sensor Andrew: Large-scale campus-wide sensing and actuation. *IBM Journal of Research and Development*, 55(1.2):6–1, 2011.

[38] A. Schumann, J. Ploennigs, and B. Gorman. Towards automating the deployment of energy saving approaches in buildings. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*. ACM, 2014, pp. 164–167.

[39] Stanford. Secure Internet of Things Project. `http://iot.stanford.edu`. 2015.

[40] U.S. Environmental Protection Agency. Building Energy Data Book. 2011.

[41] U.S. EPA/Office of Air and Radiation. The Inside Story: A Guide to Indoor Air Quality. 1988.

[42] J.-t. Wang, C.-N. Shyi, T.-W. Hou, and C. Fong. Design and implementation of augmented reality system collaborating with QR code. In *ICS'10*. IEEE, 2010.

[43] R. Want, K. P. Fishkin, A. Gujar, and B. L. Harrison. Bridging physical and virtual worlds with electronic tags. In *SIGCHI'99*. ACM, 1999.

[44] M. Weiser. The computer for the 21st century. *Scientific American*, 265(3):94–104, 1991.

[45] T. Weng, A. Nwokafor, and Y. Agarwal. Buildingdepot 2.0: An integrated management system for building analysis and control. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*. ACM, 2013, pp. 1–8.

[46] P. Windley. The CompuServe of Things. `http://www.windley.com/archives/2014/04/the_compuserve_of_things.shtml`. 2014.

[47] H. Wirtz, J. Rüth, M. Serror, J. Á. Bitsch Link, and K. Wehrle. Opportunistic interaction in the challenged internet of things. In *MobiCom'14 Workshops*. ACM, 2014.

[48] B. Zhang, Y.-H. Chen, C. Tuna, A. Dave, Y. Li, E. Lee, and B. Hartmann. HOBS: head orientation-based selection in physical spaces. In *ACM symposium on Spatial user interaction*. ACM, 2014.

Chapter 2

The Building Manager Perspective

In this chapter, we explore the human-building interface from the perspective of a building manager. A building manager is usually in charge of monitoring and controlling automated building components that are exposed through a building management system (e.g., HVAC, lighting) in a non-residential building. In the context of IoT, the number of these components is increasing. This makes it harder for often untrained building managers to scope with the high quantity of different sensor and actuator streams through traditional data analytics tools. We design and implement a system that allows a building manager to visualize and control an entire building in an intuitive way. It fully matches Weiser's use case for virtual reality [21], by applying it to the exploration of otherwise inaccessible areas (for a building manager it is not impossible, but impractical to physically visit all relevant parts of a building in order to control its functions).

In the first year of my PhD, I spent 6 months at UC Berkeley in David Culler's and Randy Katz's Software Defined Buildings (SDB) group. BUSICO 3D is a direct result of this exchange. I was part of the OpenBAS project. A project, by the U.S. Department of Energy, that funded three research teams at different universities (UC Berkeley, Carnegie Mellon University and Virginia Tech) to develop an open Building Automation System (BAS) that especially targets small and midsize commercial buildings. Hence, we developed BUSICO with the requirement of a centrally controlled system, in which a possibly untrained building manager (e.g., an employee carrying out the job part-time) is in charge of the building. The attached demo abstract was presented at SenSys'14, the OpenBAS14 workshop at CMU and at the U.S. Department of Energys Building Technologies Office Peer Review 2014 [11].

Our main goal was to develop an interface in which the building manager naturally interacts with the building, both for defining the building's schedule and setpoints, and for understanding its current and past states. Commercial automation systems and other academic work usually abstract the building state to time series data and to two-dimensional plots of this data (e.g., [7, 8, 18, 22, 10]). In BUSICO, we visualize data directly on a 3D representation of the building. We allow the manager to intuitively change that virtual environment and thereby change the physical building environment. Further, the manager can visually browse past states in the virtualization.

Mis-configurations and changes can negatively impact building occupants and are only perceptible after some time delay (e.g., for heating and cooling). This is why BUSICO also includes a simulation mode in which different setpoints and schedules can be test run in accelerated form, before they are applied on the physical building.

Because the attached paper is only a short demo paper, we now briefly discuss our design decisions and their relation to related areas of Building Information Models (BIM) and building simulation.

2.1 From Building Information Models to Game Engines

We started BUSICO by investigating Building Information Models (BIM). BIM have gained importance over the last years.[1] Ideally, they represent a digital representation of physical and functional characteristics of a building and serve thereby as a shared knowledge resource for decision making over the lifetime of a building [19]. However, outside of BIM marketing, they often do not live up to this promise. BIM use proprietary formats and are manu-

[1]The two main commercial BIM applications are Autodesk Revit [3] and ArchiCAD [1].

facturer locked-in systems. Further, the static, structural model during the construction phase is in a strong contrast to a dynamically changing building during the operational phase. We found the possibilities to directly use a BIM to create a virtual building experience, where the building state is in real-time derived from physical sensors and actuators, to be limited. This is why we abandoned the idea of implementing our system natively in a BIM.

Despite their shortcomings, commercial BIM support several widely used and standardized data formats. One example are the Industry Foundation Classes (IFC). IFC describe building and construction industry data of a BIM. They are an open specification and supported by all the main BIM applications [6].[2] Filmbox (FBX) is another widely supported format, used to represent and interchange 3D content. Despite being proprietary, it is supported both in the BIM ecosystem and the 3D modeling world. Most importantly, it is supported by the major game engines (Unity 3D, Unreal Engine).

Game engines allow for a rapid development of games and other virtual worlds by providing generic core components like a rendering, physics, collision detection engine. Some of the major engines (Unity and Unreal) have both become free for non-commercial use during the last years. We thus choose to implement our system using Unity 3D,[3] and by exporting the existing BIM structural model through FBX to Unity. If the BIM contains sensors and actuators that use the same namespace as the interface to the physical sensors and actuators (e.g., via the BACnet protocol), our workflow is not only able to derive the structural model, but also automatically match BIM sensors and actuators to their physical counterpart. Some related work has been done at MIT Media Lab. DoppelLab provides visualization and sonification of sensor data using Unity 3D [9], but focuses on presentation and leaves out actuation.

2.2 Simulation

In order to support simulation, we surveyed the variety of (building) simulation systems. However, we discovered, that most of them focus on the design and

[2]In Denmark, the use of IFC has been made mandatory for public building projects since 2003.

[3]At the time of implementation, the Unreal engine was still not freely available.

construction phase of a building and not on the operational phase. Second, many simulation tools focus on a single aspect of simulation (e.g., solely energy, lighting, heating etc..., or are too generic (e.g., [16, 14]). Third, they usually apply a high focus on the precisions of simulation results. Due to the lack of actual sensor data, simulation models need detailed input of the physical properties to calculate their output. This is opposed to simulating a building during its operational phase, where real building feedback is available as model input. Examples for widely used simulation frameworks to generally model physical systems are Matlab/Simulink and Modelica or Energy Plus, focusing solely on modeling buildings.[4]

Some directly related simulation systems exist: MLE+ provides co-simulation using Energy Plus and Matlab through sensor feedback from a BACnet interface [5]. BCVTB supports similar functionality [23]. However, both solutions require a relative deep technical and expert knowledge to be used. They are built over pre-existing BMS technologies and provide a traditional data analysis interface. For our use case, we do not require accurate simulation, but understand simulation as a user interface tool. We want to enable the building manager to naturally explore the working of the building. We therefore use Unity's lighting engine for light and implement a basic heating, cooling and energy simulation that uses the state-effect pattern [20] and a simplified thermodynamics model (see Equation 2.1).

$$T_{in}(t+1) = T_{in}(t) + \frac{U_{in} - U_{out}}{p * V * C} \tag{2.1}$$

Where:

- C is the specific heat capacity ($\frac{Joules}{Kg*Kelvin}$)
- p is the density ($\frac{kg}{m^3}$)
- V is the volume (m^3)
- U_{in}/U_{out} is the added/removed energy.

[4]An extensive list of building energy tools can be found at `http://www.buildingenergysoftwaretools.com`

U_{in} is derived through:

$$\begin{aligned} U_{in} = & \sum(Lighting_{Power}(t)) + \sum(Equipment_{Power}(t)) \\ & + \sum(Person_{HeatOutput}(t)) + \sum(Heat_{Power}) \\ & - \sum(Cool_{Power}) \end{aligned} \tag{2.2}$$

U_{out} is derived through:

$$U_{out} = h * time_{step} * A * \frac{T_{in}(t) - T_{out}(t)}{wall_{thickness}} \tag{2.3}$$

Where:

- h is the heat transfer coefficient $(\frac{W}{kg*m})$
- A is the contact area (m^2)

We derive the wall area, wall material and room and building volumes automatically from the BIM structural model and choose the correct heat transfer coefficient (e.g., $0.5\frac{W}{kg*m}$ for brick material).

2.3 Looking Forward

Recently, IBM Research picked up some of our work in [17]. They explore ways to visualize real-time building data using virtual and augmented reality. We predict that both, virtual and augmented reality systems, will become more common in a building management context in the future. They allow to scope with an ever growing number of smart devices and sensors, while lowering the professional and technical background of users. VR is about to hit the mainstream with Google, Microsoft, Facebook, Sony and Samsung all pushing their platforms commercially. At the same time, the implementation efforts of such virtual worlds are steadily reduced by the increased availability of digital BIM models, but also by the availability of both hardware (e.g., [12, 15, 2]) and software and algorithms (e.g., [4, 13]) to construct digital representations of the physical world. We therefore believe, that building management will be one of the main applications of VR outside gaming in the near future.

As future work, we want to investigate the crowdsourced creation of such models using consumer hardware (e.g., smartphones). Besides the issues that are involved in the creation itself (e.g., wall and material detection, user assistance in creation process), other problems are the (semi-) automated matching of sensor and data points to the digital model.

References

[1] ArchiCAD. `http://www.graphisoft.com/archicad`.

[2] Asus. Xtion PRO LIVE. `https://www.asus.com/us/3D-Sensor/Xtion_PRO_LIVE/`.

[3] Autodesk Revit. `http://www.autodesk.com/products/revit-family/overview`.

[4] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*. Springer, 2006, pp. 404–417.

[5] W. Bernal, M. Behl, T. X. Nghiem, and R. Mangharam. MLE+: A Tool for Integrated Design and Deployment of Energy Efficient Building Controls. In *Proceedings of the fourth acm workshop on embedded sensing systems for energy-efficiency in buildings*, 2012, pp. 123–130.

[6] BuildingSMART. Open Standards 101. 2014. URL: `http://www.buildingsmart.org/standards/technical-vision/open-standards-101/`.

[7] S. Dawson-Haggerty, X. Jiang, G. Tolle, J. Ortiz, and D. Culler. sMAP: a simple measurement and actuation profile for physical information. In *SenSys'10*. ACM, 2010, pp. 197–210.

[8] S. Dawson-Haggerty, A. Krioukov, J. Taneja, S. Karandikar, G. Fierro, N. Kitaev, and D. Culler. BOSS: building operating system services. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, 2013, pp. 443–457.

[9] G. Dublon, L. Pardue, B. Mayton, N. Swartz, N. Joliat, P. Hurst, and J. Paradiso. DoppelLab: Tools for exploring and harnessing multimodal sensor network data. In *Sensors, 2011 ieee*, 2011, pp. 1612–1615.

[10] G. Fierro and D. E. Culler. XBOS: An Extensible Building Operating System. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. ACM, 2015, pp. 119–120.

[11] J. Fürst, G. Fierro, P. Bonnet, and D. E. Culler. BUSICO 3D: Building Simulation and Control in Unity 3D. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*. ACM, 2014, pp. 326–327.

[12] Google. Project Tango. `https://get.google.com/tango/`.

[13] M. Labbe and F. Michaud. Appearance-based loop closure detection for online large-scale and long-term operation. *IEEE Transactions on Robotics*, 29(3):734–745, 2013.

[14] MathWorks, Inc. Simulink. 2015. URL: `http://se.mathworks.com/products/simulink/`.

[15] Microsoft Kinect. `https://developer.microsoft.com/en-us/windows/kinect`.

[16] Modelica Association. Modelica and the Modelica Association. 2015. URL: `https://www.modelica.org`.

[17] J. Ploennigs, J. Clement, and B. Pietropaoli. Demo Abstract: The Immersive Reality of Building Data. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. ACM, 2015, pp. 99–100.

[18] A. Rowe, M. E. Berges, G. Bhatia, E. Goldman, R. Rajkumar, J. H. Garrett Jr, J. M. Moura, and L. Soibelman. Sensor Andrew: Large-scale campus-wide sensing and actuation. *IBM Journal of Research and Development*, 55(1.2):6–1, 2011.

[19] US National Building Information Model Standard Project Committee. Frequently Asked Questions About the National BIM Standard-United States. 2015. URL: `http://www.nationalbimstandard.org/faq.php#faq1`.

[20] G. Wang, M. V. Salles, B. Sowell, X. Wang, T. Cao, A. Demers, J. Gehrke, and W. White. Behavioral simulations in MapReduce. *Proceedings of the VLDB Endowment*, 3(1-2):952–963, 2010.

[21] M. Weiser. The computer for the 21st century. *Scientific American*, 265(3):94–104, 1991.

[22] T. Weng, A. Nwokafor, and Y. Agarwal. Buildingdepot 2.0: An integrated management system for building analysis and control. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*. ACM, 2013, pp. 1–8.

[23] M. Wetter. Co-simulation of building energy and control systems with the Building Controls Virtual Test Bed. *Journal of Building Performance Simulation*, 4(3):185–203, 2011.

Demo Abstract: BUSICO 3D - Building Simulation and Control in Unity 3D

Jonathan Fürst
IT University of Copenhagen
jonf@itu.dk

Gabe Fierro
UC Berkeley
gt.fierro@berkeley.edu

Philippe Bonnet
IT University of Copenhagen
phbo@itu.dk

David E. Culler
UC Berkeley
culler@berkeley.edu

Abstract

In this demonstration, we present a novel system of building control and simulation focused on the integration of the physical and virtual worlds. Actuations and schedules can be manifested either in a physical space or in a virtualization of that space, allowing for more natural interactions with simulations and easier transferring of schedules and configurations from the simulated virtual environment to a real-world deployment. We provide an implementation using a widely used game engine (Unity 3D) and sMAP (Simple Measurement and Actuation Profile), a developed time series database and metadata store.

1 Introduction

Buildings are increasingly turning into systems of human-computer-building interaction via software. People are beginning to use mobile apps to control their devices like lights or thermostats. The 'Internet of Things' has spawned many new devices that people can interact with, such as the Philips Hue LED bulb and Nest thermostat. At the same time, there have been research efforts towards making existing building instrumentation accessible in a uniform way and integrating it with this new generation of smart devices, e.g., [1], [2]. Many of these projects provide relatively mature solutions for abstracting physical sensors and actuators to a uniform interface, but the question of how inhabitants or building managers access this physical information is still not very well developed. Interfacing with a building often requires programming or technical experience specific to that system. Furthermore, access to a Building Automation System (BAS) is often restricted because running applications or experiments on a building or changing its configuration can be harmful.

SenSys'14, November 3–5, 2014, Memphis, TN, USA.
ACM 978-1-4503-3143-2/14/11.
http://dx.doi.org/10.1145/2668332.2668380

With BUSICO 3D, we present a system that provides: (i) a virtual representation of a building as a more natural interface for human interaction, and (ii) an integration of simulation and control that allows the testing of configurations without any harm on the physical building or personal environment (see Figure 1). BUSICO 3D uses sMAP as an abstraction layer for accessing and describing various sensors and actuators of the physical building. The structure of the physical building is derived from an imported BIM (Building information modeling) model. The system can be run in either simulation or control mode. In simulation mode, people can change settings, define rules or schedules, and simulate the effect of these on the building. Feedback on the building's state, including temperature and energy consumption, is provided through the virtual experience. In control mode, feedback of the simulation engine is replaced by feedback of the actual building. In the following, we describe our prototype design and implementation using a widely used game engine (Unity 3D) and existing platforms for modeling the physical structure of a building (Autodesk Revit).

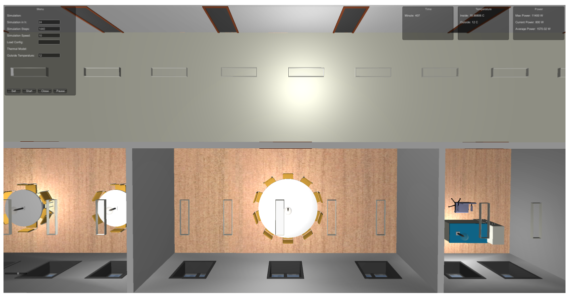


Figure 1. BUSICO 3D from birds-eye perspective.

2 Design and Implementation

Here we explain briefly the different components and working of our system. We finish by describing a typical interaction with our system. We have built a physical testbed for our system that resembles two rooms of a building (see Figure 2).

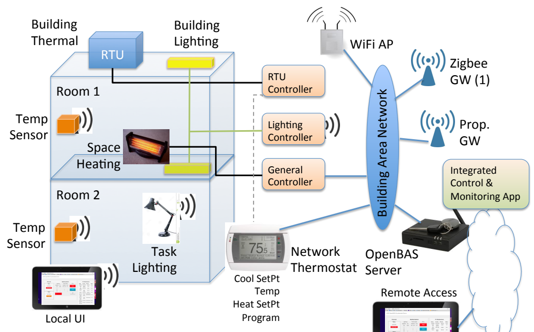


Figure 2. Schematic view of our testbed.

2.1 Components

BUSICO 3D uses Unity 3D as an execution environment. Unity is a game engine that allows a fast implementation of games and other virtualizations by providing a common set of features like graphics, lighting or physics engine. Unity allows for scripting of the underlying engine in several high level languages (C#, JavaScript and Boo). BUSICO 3D is entirely written in C#. It allows the import of a CAD architectural model of the building that is created in a modeling program of choice (Autodesk Revit, Graphisoft Archicad, Sketchup). sMAP acts as a hardware abstraction layer on top of various physical devices. sMAP drivers abstract away the particulars of sensors and actuators and expose a uniform REST interface [1]. BUSICO 3D reads this REST interface of sensors and actuators and maps it by means of sMAP metadata to the CAD model, creating automatically objects for each found sensor or actuator including GUI, programming logic and renderer.

2.2 Simulation and Control Engine

BUSICO 3D is built around an integrated simulation and control engine that follows the working of discrete event simulation. Every object (e.g., an object representing a thermostat) has a `Start()` method and an `Update()` method. `Start()` gets called once when the object is created, `Update()` gets called every frame by the Unity engine. Our system adds to this the state-effect pattern ([3]) by dividing calls to `Update()` in an 'effect' and 'state' phase. In the 'effect' phase, all objects are calculating effect values based on the state values of themselves and related objects. These effect values are then saved for the 'state' phase. Here, objects are accessing effect values to calculate their new states.

2.3 Schedules and Rules

BUSICO 3D includes an event-driven rule engine. A user may specify simple rules in the form of:

```
IF (condition) THEN (consequence);
```

An example could for instance be:

```
IF (window1==open)
    THEN (thermostat1_mode=OFF);
```

BUSICO 3D also includes a scheduler in which users can set different schedules for their lights or temperature in a building. Scheduling can also be event driven, in which case events are points in time.

```
IF (time==2014-07-27T07:48:42)
    THEN (thermostat1_temp=42);
```

2.4 User Interaction

A sMAP configuration file describes the components of a building; BUSICO 3D automaps the various devices therein to their correct locations. A user has two view options. She can observe each floor from a birds-eye-perspective, or she can switch to a first person view and walk through the building. Actuation can be done in two ways, either in pure simulation in which the drivers act as virtual drivers that include a thermal model and a model for power consumption, or it can be done by actually controlling the physical building. In a real scenario, a building manager first tries out different schedules and rules, runs the simulation and optimizes it over time. When she is sure about the result, the building configuration is applied to the real building. At any point, it is possible to walk through the building to have a visualization of different aspects (e.g light, heating, energy consumption). When discovering something to change there is no need to switch to a different view, the user can just naturally switch devices or set new setpoints like she would in a real building.

3 Conclusion and Future Work

BUSICO 3D provides a natural interface for building control and simulation that allows non-experts to interact with a building and configure it. Using a game engine for implementation allowed fast prototyping. We are currently exploring how BUSICO 3D can be tailored for various types of what-if analysis, and more generally as a means of experimenting with new forms of building operations. We are also designing serious games to gain insight on the social practices in a given building and the interactions that define how the building is actually used. We want to implement a more natural experience to the building by using a virtual-reality headset like Oculus Rift. On the simulation side we are currently investigating integration with a more sophisticated simulation engine like Energy Plus.

4 References

[1] S. Dawson-Haggerty, X. Jiang, G. Tolle, J. Ortiz, and D. Culler. sMAP: A Simple Measurement and Actuation Profile for Physical Information. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 197–210, New York, NY, USA, 2010. ACM.

[2] C. Dixon, R. Mahajan, S. Agarwal, A. B. Brush, B. Lee, S. Saroiu, and P. Bahl. An Operating System for the Home. In *NSDI*, volume 12, pages 337–352, 2012.

[3] G. Wang, M. V. Salles, B. Sowell, X. Wang, T. Cao, A. Demers, J. Gehrke, and W. White. Behavioral Simulations in MapReduce. *Proc. VLDB Endow.*, 3(1-2):952–963, Sept. 2010.

Chapter 3

Crowd-Sourced Metadata Management

Metadata helps to identify smart appliances for human-building interaction (e.g., is the appliance a light or thermostat? Which sensors and which actuators does it contain?). Cyber-physical application usually query sensors and actuators based on their metadata (e.g., [8, 9, 2]). This makes consistent metadata crucial to achieve compatibility between IoT devices and to support portable, cyber-physical applications. Building Management Systems (BMS) are existing (smart) environments of connected sensors and actuators, in which inconsistent and incomplete metadata is causing many problems. We thus specifically approach the issue of metadata collection and maintenance in the context of BMS.

Recent approaches towards consistent BMS metadata either try to detect patterns in metadata descriptors (e.g., [3, 23]) or use machine learning techniques, like a clustering based on sensor values (e.g. [13]). Our system, *Babel*, makes use of crowdsourcing to incrementally create a consistent metadata state for sensors and actuators in a BMS. We achieve a high accuracy and are further able to map an actuator or sensor to its physical location in a building. Such mapping is important to enable location context based applications (e.g., a smartphone application that allows to control lighting and heating in an office). Babel is the first system that applies crowdsourcing to BMS metadata, and it is likewise applicable to other emerging forms of IoT systems.

We presented a demo of Babel at BuildSys'15 and at the SDB winter retreat 2016 [10]. The included paper was presented at PerCom workshops 2016 [11]. To better place the problems that we address with Babel, we now briefly introduce the reader to BMS and their metadata representation. Further, we describe our motivation and future challenges more detailed than it is done in the included paper.

3.1 Building Management Systems

Most larger non-residential buildings contain a BMS. A BMS connects different controllers, sensors and actuators and allows for a central control of building functions such as HVAC and lighting. BMS follow usually a three-tier model of management, automation and field level (see Figure 3.1). At the lowest level, sensors and actuators communicate through a field bus (digital serial data bus) with each other and with control devices of the upper automation layer. Communication at the automation and management level usually takes place over LAN. Automation occurs locally, via a direct coupling of sensors and actuators (occupancy and light), on the automation level (via direct digital controllers [DDCs]) or on the management level (BMS machine) [16]. To retrieve digital values of set- and datapoints, a building manager can query BMS points on the management level.

3.1.1 BMS Metadata

The available metadata and its structure dependents on the specific BMS. BMS in the wild contain all, from proprietary and niche technologies, to standardized and widely used ones. Examples for widely used standardized systems and protocols are LonTalk, KNX, BACnet and Modbus. Interoperability issues across these systems are often faced problems, but even device interoperability within a single standard is not guaranteed. This lack of device and protocol interoperability translates itself to an inconsistent use of metadata. Metadata is usually applied manually during construction and commissioning. These are not atomic events, but processes that are almost always performed by several workers from different organizations and during different times. Further,

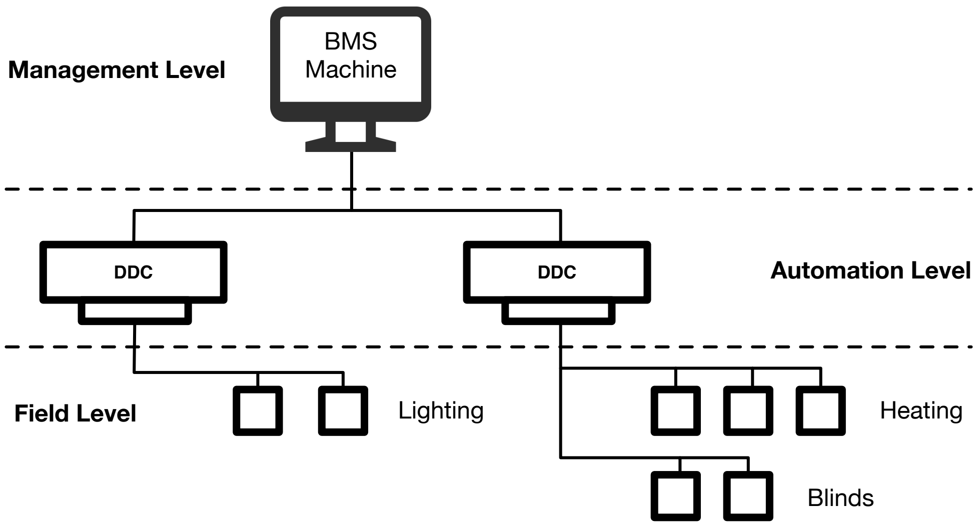


Figure 3.1: General Three-Tier BMS Architecture

Table 3.1: BMS Metadata from Soda and Davis Hall

BMS Label	**Decoded**
`SDH.AH1_RHC-9:HEAT.COOL`	Sutardja Dai Hall, Air Handler 1, Room 9, Heat Cool Setpoint.
`SDH.S1-04:AI 3`	Sutardja Dai Hall, Analog Input 3.
`DH.AH1B.SF_VFD:CMD STP.STRT`	Davis Hall, Air Handler 1B.
`WS86002.RELAY02`	Sutardja Dai Hall, Relay to switch lights on a hallway on 2nd floor.
`DH.AH1A_HCDAT`	Davis Hall, Air Handler 1A, Heating Coil Discharge Temperature.

over the lifetime of a building, devices get replaced and added on a regular basis and spaces change their original use—metadata however rarely follows through [12].

Most BMS systems support the labeling of data and set-points as a way of identification. To show a practical example, Table 3.1 shows some of the metadata that we found in our test-case building and a second building on campus. Common practices are the use of abbreviations for describing building component keywords (e.g., from the HVAC, lighting domain) and for specifying the location (building and room names). These encodings are however not consistent and not easily parseable for both machine and human.

One approach followed in industry and academia towards improving consistency in the long term, is the standardizing of name spaces:

- **The Industry Foundation Classes (IFC).** A standard for interoperability within the building and construction domain [5]. Due to its origin in the construction/architecture domain, it has a strong focus on the physical building structure and materials.

- **Project Haystack** is an ongoing open source initiative out of the BMS community to develop naming conventions and taxonomies for building equipment and operational data [22]. It therefore aims to be directly applicable to existing BMS data models that have as their lowest common denominator only the labeling of points.

- **CityGML** is a common XML based information model for representation, storage, and exchange of virtual 3D city and landscape models that originates in the geospatial research community [21]. Opposed to IFC, CityGML describe how buildings are observed or used [18]. It has not been updated since 2012.

- **ISO 16484–3:2005** specifies the requirements for the overall functionality and engineering services to achieve building automation and control systems. It defines a template for a BMS point list [14].

However, these naming standards are commonly not adhered to by building constructors and commissioners [23]. And second, it has recently been argued that none of them is descriptive enough to cover all building use cases [4].

3.2 Motivation

Consistent metadata supports two important goals: (i) Portable cyber-physical applications (e.g., our own, previously discussed interface for building managers, BUSICO 3D, depends on metadata), and (ii) a continuous building commissioning process. We describe both in the following.

3.2.1 Portable Cyber-physical Applications

Traditionally, BMS are custom fitted and isolated for a single building. They are intended to be accessed by a single, central machine that performs control and analytical tasks based on the data that is available from sensors and actuators and the expert input of the building manager.

Commonly, buildings are controlled using the PMV (Predicted Mean Vote) method. PMV defines comfort for air-conditioned buildings. It is based on a generic definition for an optimal environment. The model often contradicts the goal of low carbon buildings and disregards differences in individual comfort preferences and their seasonal dependency [6].

In contrast, the adaptive model is based on the fact that outdoor climate influences indoor comfort, because humans are able to adapt to different temperatures during different seasons of the year. It also accounts for differences in comfort preferences of individuals by aiming at an occupant controlled, naturally conditioned building. The access to environmental controls and thermal data influences building occupants' thermal expectations and preferences [19]. Many argue that ubiquitous computing and cyber-physical applications will play a key role in enabling an adaptive, energy saving model for buildings [6].

Establishing such an adaptive model requires, like any application outside the scope of the original BMS, consistent metadata. Therefore the suboptimal state of BMS metadata is a severe problem. It hinders the development of such applications, because they are not portable across buildings and need to be custom fit to every new deployment building.

3.2.2 Continuous Building Commissioning

Building commissioning is typically performed when a building is transferred from the constructor/architect to the actual owner and operator. The commissioning process assures that the building meets the requirements set by the building owner and is fully functioning. The functions that are commissioned are, e.g., HVAC, electrical and safety systems. The 'ASHRAE Guideline 0–2013: The Commissioning Process' defines building commissioning as follows [1]:

"*The Commissioning Process is a quality-focused process for enhancing the* *delivery of a project by verifying and documenting that the facility and all of its systems and assemblies are planned, designed, installed, tested, operated, and maintained to meet the Owner's Project Requirements.*"

If the building controls these functions through a BMS—as it is nearly always the case in recent buildings—, then also the working of the BMS is tested. Lately, continuous commissioning (or re-commissioning) is gaining importance as a quality process. It ensures that buildings continue to maintain their functionality throughout their life-cycle. Re-commissioning results in energy savings: e.g., a recent study has shown average savings of 0.29$/sqft per year for schools, hospitals and office buildings in the US [20]. A meta-analysis has come to median commissioning costs of $0.27/sqft, whole-building energy savings of 15%, and payback times of 0.7 years [17].

Despite these results, re-commissioning is not used on a regular basis. It is mainly hindered by the relative long duration (usually several months) and the high initial costs. Costs are mostly based on manual labor and a potential need for additional monitoring infrastructure (e.g., the installation of energy meters). Much of the manual labor is necessary because metadata is inconsistent, incomplete and missing. Thus, the commissioning experts manually need to visit most parts of the building.

We argue, that a continuous and crowdsourced labeling of BMS points is a way to greatly improve the process of re-commissioning. Inconsistent metadata is one of the reason that the re-commissioning process involves so much manual labor [17]. The crowdsourced approach can also be extended to other areas of the building (e.g., to report broken devices).

3.3 Looking Forward

Looking forward, we believe that the issue of metadata is an issue that needs to be addressed not only within the scope of a building, but for the greater vision of IoT, where a vast number of heterogeneous devices are connected. Future cyber-physical applications might require to access devices on the scope of a whole city by means of their metadata.

We find a promising approach towards this vision in the renaissance of the semantic web as Linked Data[15]. Linked Data could be described as a graph of machine-readable data, published in an open, shared document format. Documents are universally resolvable by a URI. They link to each other, forming a shared and structured graph of knowledge. Several open standards exist. The Resource Description Framework (RDF) is the standard model of the W3C for Linked Data exchange on the Web [24]. On the lowest level, RDF consists of `subject-predicate-object` triples. RDF has several serializations standards like XML or JSON-LD (see `http://www.w3.org/TR/json-ld/`).

We thus follow the reasons given by Curry et al. in [7] that the Linked Data approach should also be applied to the building domain, solving not only interoperability issues inside that domain, but also with other systems in the context of IoT. Human-in-the-loop feedback systems, like Babel, are one approach to keep the state of the Linked Data graph in synchronization with the state of the physical world. The incentive to participate in such systems can be achieved by introducing some playful, game-like factor. In the attached paper, we mention the augmented reality game Ingress as a successful application in this domain. Now, at the time of writing this thesis, Pokémon Go has by far outreached Ingress success in a few days. Google owns the majority of shares behind Niantic, the company behind Ingress and Pokémon Go.

References

[1] ASHRAE. Guideline 0-2013: The Commissioning Process. 2013.

[2] B. Balaji, J. Xu, A. Nwokafor, R. Gupta, and Y. Agarwal. Sentinel: Occupancy based HVAC actuation using existing WiFi infrastructure within commercial buildings. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2013, p. 17.

[3] A. A. Bhattacharya, D. Hong, D. Culler, J. Ortiz, K. Whitehouse, and E. Wu. Automated metadata construction to support portable building applications. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. ACM, 2015, pp. 3–12.

[4] A. Bhattacharya, J. Ploennigs, and D. Culler. Short Paper: Analyzing Metadata Schemas for Buildings: The Good, the Bad, and the Ugly. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. ACM, 2015, pp. 33–34.

[5] buildingSMART. IFC Standard. 2015. URL: `http://www.buildingsmart-tech.org/specifications/ifc-overview`.

[6] A. K. Clear, J. Morley, M. Hazas, A. Friday, and O. Bates. Understanding adaptive thermal comfort: new directions for UbiComp. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*. ACM, 2013, pp. 113–122.

[7] E. Curry, J. O'Donnell, E. Corry, S. Hasan, M. Keane, and S. O'Riain. Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics*, 27(2):206–219, 2013.

[8] S. Dawson-Haggerty, X. Jiang, G. Tolle, J. Ortiz, and D. Culler. sMAP: a simple measurement and actuation profile for physical information. In *SenSys'10*. ACM, 2010, pp. 197–210.

[9] V. L. Erickson and A. E. Cerpa. Thermovote: participatory sensing for efficient building hvac conditioning. In *BuildSys'12*. ACM, 2012.

[10] J. Fürst, K. Chen, R. H. Katz, and P. Bonnet. Demo Abstract: Human-in-the-loop BMS Point Matching and Metadata Labeling with Babel. In *Proceedings of the 2nd ACM Internation*. ACM, 2015, pp. 101–102.

[11] J. Fürst, K. Chen, R. H. Katz, P. Bonnet, et al. Crowd-sourced BMS point matching and metadata maintenance with Babel. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. IEEE, 2016, pp. 1–6.

[12] T. Haasl and T. Sharp. *A practical guide for commissioning existing buildings*. Oak Ridge National Laboratory, 1999.

[13] D. Hong, J. Ortiz, K. Whitehouse, and D. Culler. Towards automatic spatial verification of sensor placement in buildings. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*. ACM, 2013, pp. 1–8.

[14] International Organization for Standardization. ISO 16484-3:2005. 2005.

[15] Linked Data. `http://linkeddata.org/`.

[16] H. Merz, T. Hansemann, and C. Hübner. *Building Automation: Communication Systems with EIB/KNX, LON and BACnet*. Springer Science & Business Media, 2009.

[17] E. Mills. *The cost-effectiveness of commercial-buildings commissioning: A meta-analysis of energy and non-energy impacts in existing buildings and new construction in the United States*. Lawrence Berkeley National Laboratory, 2004.

[18] C. Nagel, A. Stadler, and T. H. Kolbe. Conceptual requirements for the automatic reconstruction of building information models from uninterpreted 3D models. *Proceedings of the International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*:46–53, 2009.

[19] J. Nicol and M. Humphreys. New standards for comfort and energy use in buildings. *Building Research & Information*, 37(1):68–73, 2009.

[20] S. Oh and D. Claridge. Implemented Continuous Commissioning Measures for Schools, Hospitals, and Office Buildings in the US, 2014.

[21] Open Geospatial Consortium. CityGML. 2012. URL: `http://www.opengeospatial.org/standards/citygml`.

[22] Project Haystack. `http://project-haystack.org/`. 2015.

[23] A. Schumann, J. Ploennigs, and B. Gorman. Towards automating the deployment of energy saving approaches in buildings. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*. ACM, 2014, pp. 164–167.

[24] World Wide Web Consortium (W3C). Resource Description Framework (RDF). 2014. URL: `http://www.w3.org/RDF/`.

Crowd-sourced BMS Point Matching and Metadata Maintenance with Babel

Jonathan Fürst*, Kaifei Chen†, Randy H. Katz† and Philippe Bonnet*
*IT University of Copenhagen, †UC Berkeley
jonf@itu.dk, {kaifei, randykatz}@berkeley.edu, phbo@itu.dk

***Abstract*—Cyber-physical applications, deployed on top of Building Management Systems (BMS), promise energy saving and comfort improvement in non-residential buildings. Such applications are so far mainly deployed as research prototypes. The main roadblock to widespread adoption is the low quality of BMS metadata. There is indeed a mismatch between (i) the anecdotal nature of metadata for legacy BMS – they are usually initialized when the BMS is commissioned and later neglected–, and (ii) the imperious need for consistent and up-to-date metadata for supporting building analytics or personalized control systems. Such applications access sensors and actuators through BMS metadata in form of point labels. The naming of labels is however often inconsistent and incomplete. To tackle this problem, we introduce Babel, a crowd-sourced approach to the creation and maintenance of BMS metadata. In our system, occupants provide physical and digital input in form of actuations (e.g., the turning on/off a light) and readings (e.g., reading room temperature of a thermostat) to Babel. Babel then matches this input to digital points in the BMS based on value equality. We have implemented a prototype of our system in a non-residential building. While our approach can not solve all metadata problems, we show that it is able to match end-user relevant points in a fast and precise manner.**

I. Introduction

Non-residential buildings are a prime platform for novel cyber-physical applications that can reduce energy consumption and improve occupant comfort. Reducing energy consumption in non-residential buildings is an important goal, considering that these buildings account for ca. 19% of primary energy consumption in the U.S. [1]. Occupant comfort is likewise important because we spend more than 90% of our time inside buildings [2].

Around half of non-residential buildings are already instrumented with a Building Management System (BMS). A BMS is a typically tightly coupled digital control and sensing system that performs automation and management tasks for a particular building (HVAC, lighting etc.) [1].

Lately, BMS have caught the interest of the Sensor Network community. Previous groundwork has made BMS sensors and actuators available to cyber-physical applications by abstracting them to a common interface (e.g., [3]). Early cyber-physical applications on top of BMS show great potential: Erickson and Cerpa develop a system that allows users to directly communicate their thermal preferences to the BMS, leading to energy savings of 10% and increased personal comfort [4]. Narayanaswamy et al. develop a system that optimizes energy usage and comfort by detecting anomalies in HVAC systems [5].

However, such cyber-physical applications are not yet deployed beyond building-specific research experiments. The problem is metadata. Traditionally, a BMS is commissioned and its functions are hardcoded to the specific characteristics of a building and needs of a building manager. After commissioning, the BMS becomes a legacy system and metadata is largely irrelevant. Devices get replaced and added on a regular basis while spaces get re-configured and change their use (e.g., a classroom is transformed into a lab). Metadata rarely follows such evolution [6].

BMS metadata is usually restricted to a single label that is set manually for each BMS point (e.g., a single sensor or actuator). Specific commissioning contractors are responsible for defining names for these BMS labels. Common naming practices rely on the use of abbreviations for describing building component keywords (e.g., from the HVAC, Lighting domain) and for specifying the location (building, room names). These encodings are not defined by a well-formed namespace. They are not easily parsable, even for humans. Here are some exemplary problems that we found while analyzing the BMS labels in three of our campus buildings:

- No strict naming scheme within a single building. E.g., `WS86007.RELAY12` and `SDH.PXCM-08SDH.S5-04:ROOM TEMP` represents a light switch and the room temperature of a thermostat at the same location.
- Different labels for the same semantic meaning (`RM` and `ROOM`, `TEMP` and `TMP`, `STPT` and `SP`). E.g., `SDH.CP.RESET.TMP.SP.MIN` and `SDH.CAC-3:ROOMTEMP` use different labels for temperature sensors.
- Incomplete metadata due to the restriction to point labels as data structure. For example `WS86007.RELAY12` completely misses information about the location of the light switch.

Inconsistent BMS metadata has been studied before (see Section II for an overview). What distinguishes our approach from previous work is that (i) we apply crowd-sourcing, relying on the building occupants, (ii) we consider both, actuators and sensors while (iii) we provide points with a mapping to their physical location. Our hypothesis is that much of the physical state of a building can be observed by humans and that building metadata maintenance should be based on human input.

In our system, *Babel*, occupants provide physical and digital input in form of actuations (e.g., the switching of a light) and readings (e.g., the reading of the room temperature of a thermostat). Babel then matches user input to points in the BMS by comparing point values to user provided input. For example, if the user notifies Babel that the temperature is 67^oF, we are able to reduce the qualifying points to all points with value 67. By further iterative refinement of crowd-sourced data (a user reports another value for the same temperature point), we can reduce the qualifying points eventually to a single match. Our intention is to enable occupants to set up the metadata for their own office space by providing input to Babel. After performing this setup process, they are then able to use a personal comfort application, e.g., on their smartphone. We consider BMS metadata maintenance as a good fit for crowd-sourcing for the following reasons:

- BMS metadata maintenance is parallelizable (there can be several parallel user inputs) and serializable (the order of user inputs does not matter).
- Many application relevant BMS points are visible to humans (e.g., thermostat setpoint, light state) and the user effort to report their state is small.
- It can be partitioned (e.g., metadata maintenance in one office is independent from maintenance in another).
- The introduction of our system can improve building operation and allow personal comfort applications. This helps to incentivize people to participate.

The core contribution of our work is the design and implementation of Babel, a system that allows for a incremental, crowd-sourced construction and maintenance of BMS metadata. We implement and evaluate Babel on an actual non-residential building in the U.S. to show the applicability, performance and accuracy of our system. To our knowledge, this is the first work that applies crowd-sourcing to the problem of inconsistent and incomplete BMS metadata.

The remainder of this paper is structured as follows: First, we present related work. We then follow with our design goals and the design of our system. Finally, we presents our deployment and evaluation in a non-residential building, concluded with an outlook on future work.

II. Related Work

Metadata population has been covered extensively for different content. For text, automated metadata generation has been done in various ways, e.g. through natural language processing [7]. Yang and Lee propose a machine learning approach to automatically generate metadata for the semantic web from the content of webpages. Rodriguez et al. use associate networks to transfer metadata from metadata-rich resources to metadata-poor resources. They evaluate their system using a bibliographic dataset [8].

Semi-automated, crowd-sourced metadata generation is very successful in the so-called folksonomies using e.g., community based tagging, where users tag a typically small fraction of documents [9]. An example is the tagging of YouTube videos. Projects like OpenStreetMap go further by collaboratively creating a free editable map of the world that has become comparable if not superior in quality than geo-data from commercial providers [10]. In the building domain, Rice and Woodman have successfully applied the concept of crowd-sourced construction of world models. They present a system that allows occupants to map the inside of a building [11].

In the sensor network community, Bhattacharya et al. proposes a system where sensor metadata is semi-automatically completed using regular expressions to detect common patterns in metadata descriptors using the expertise input of the building manager [12]. They achieve a correct matching of 70% of data points using relatively few manual sample matches. But they note that many labels have a very low frequency, making it very hard to qualify them automatically. Hong et al. apply spatial clustering to classify relative sensor locations with some initial success for 5 rooms and 15 sensors [13]. Finally, Schumann et al. present an approach for the semi-automated mapping of BMS and Energy Management System (EMS) labels. They propose semantic techniques for computing similarity values between BMS and EMS labels. These similarity values are subsequently used to reduce the number of points a user needs to consider when he/she matches labels manually. However they conclude that their approach only identifies the correct match in 16% of all cases and hence is not fit for automated labeling.

To solve the metadata problem in the first place, several long-term efforts to standardize naming exist. Most notable the ISO Industry Foundation Classes (IFC) [15], the open source initiative Project Haystack [16] and ISO 16484-3:2005 [17]. These standards are commonly not adhered to by building constructors and commissioners. Furthermore, a study recently came to the conclusion that none of these metadata schemes fully captures to model a building's sensors and actuators and their relationships [18].

III. Design Goals

We aim to solve the problem of inconsistent and incomplete BMS metadata through crowd-sourcing. Our design goals are the following:

- **Global namespace.** A global metadata namespace enables portable applications across different buildings. Our system should therefore map the current, loose namespace of a building to a global one.
- **Limited user involvement.** To achieve a wide adoption of a crowd-sourced system, user involvement should be limited. Our system should ideally rely on a localization system, so the location of the user can be automatically selected and she does not need to manually input it. Further, Babel should allow for the introduction of a reward system in terms of a salary bonus or internal competition. It should be possible to give users an immediate incentive by e.g., enabling a personal comfort application for them after points have been matched for their office.
- **Backward compatibility.** Buildings have long life-cycles of 50-70 years. This means that our system needs to be able to work with existing BMS and their characteristics (e.g., slow and unreliable data-polling). More recent buildings contain mostly one of the wider used,

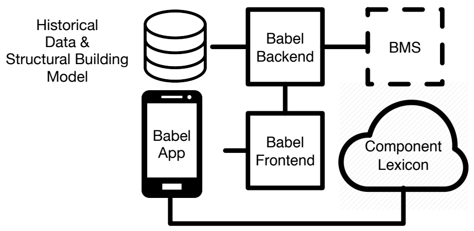
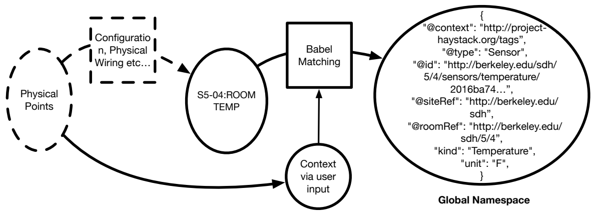


Fig. 1. Babel Architecture



```
{
"@context": "http://project-haystack.org/tags",
"@type": "Sensor",
"@id": "http://berkeley.edu/sdh/5/4/sensors/temperature/2016ba74…",
"@siteRef": "http://berkeley.edu/sdh",
"@roomRef": "http://berkeley.edu/sdh/5/4",
"kind": "Temperature",
"unit": "F",
}
```

Fig. 2. Mapping the loose BMS Namespace to a Global Namespace

standardized protocols like BACnet, KNX or LONWorks. Many open source and research projects have made it possible to integrate these (e.g., [3]). Our system should build on top of this work. Due to the slow data polling rate of BMS, our system should rely on asynchronous requests where possible.

- **Robustness.** Resulting matches and metadata need to be robust in regards to the human factor. We generally trust the users of our system. We assume that a user will not sabotage our system by providing wrong input. Users will be affiliated to the building as employees or students, which makes this assumption reasonable. Further, in a real deployment, countermeasures against faulty input can be the requirement of *n* correct matches for a point instead of *1* and to distinct between trustworthy and less trustworthy users. However, our system should handle small imperfection and imprecisions in the user input. For example, a user might report 64^oF, while the actual value is 64.3^oF, or she might report an observation with some delay to Babel.
- **Dynamicity.** To achieve dynamic metadata, our system should be able to verify its metadata at defined time-intervals. The time-interval depends on the requirement of the specific building.

In the following we describe our system design and implementation based on these design goals.

IV. System Design

Figure 1 depicts the general architecture of Babel. A smartphone application accesses a Web service in the cloud that contains a lexicon for different device types and points (e.g., a light on/off switch, a thermostat temperature point). The same lexicon is used across all buildings to enforce a global namespace. The smartphone application can further access the local Babel service. It provides (i) the specific, structural model of the building, adhering to a global naming scheme and (ii) an entry point for users to report new values from the physical world. When a user reports a value, it is compared against the current values in the BMS in order to find a match. The matching state is stored in a local database.

A. Metadata Model

Our system requires a data model that is able to represent the different BMS points and respective locations, while allowing to integrate elements outside of the building domain. A cyber-physical application might for instance retrieve local weather information to adjust the HVAC setpoints.

We follow Curry et al. [19] and apply the Linked Data approach to the building domain, solving interoperability issues not only inside that domain, but also with other systems in general. The Resource Description Framework (RDF) is the standard model of the W3C for Linked Data exchange on the Web [20]. RDF has several standardized serializations formats like XML or JSON. In Babel we use JSON-LD (http://www.w3.org/TR/json-ld/) to model the structure of a building and its different components as Linked Data. We principally follow the naming scheme from Project Haystack, which encompasses many points from the BMS domain (e.g., temperature, humidity and CO2 sensors and thermostat setpoints). Points that are not defined in Haystack are included by linking to other naming schemes (e.g., https://schema.org). Figure 2 shows by means of an example how the loose, existing namespace of BMS points is mapped to a global namespace by relying on user provided context. Sensors and actuators make up a majority of BMS points. They are physically wired to a controller, which connects to the BMS backend. Physical sensor and actuator states are then perceived by occupants and serve as parameter for Babel's point matching process. Matches are thus constructed incrementally over time and change as the building changes. This might be the case when the physical points change because part of the HVAC system gets replaced. In the example, a temperature sensor is mapped to a global namespace using the Haystack scheme.

B. Namespace Mapping and Point Matching

To match a single BMS point, we distinguish it from the set of all points through iterative refinement and enrich it with metadata. This is achieved by the simplified matching process seen in Figure 3. We need to eliminate other, unpredictable datapoint changes that might happen during an unfinished matching process. When receiving a new user input, Babel first reduces the points to consider for this request. It removes already successful matches and points whose "BACnet type" value does not fit the type of the point that should be matched.[1] Then it queries the remaining points and compares their value with the user provided value. A substantially reduced list of points is the result. This process is repeated when user input

[1]The BACnet standard defines 54 point types (e.g., Binary Input, Binary Output, Analog Input etc.) [21].

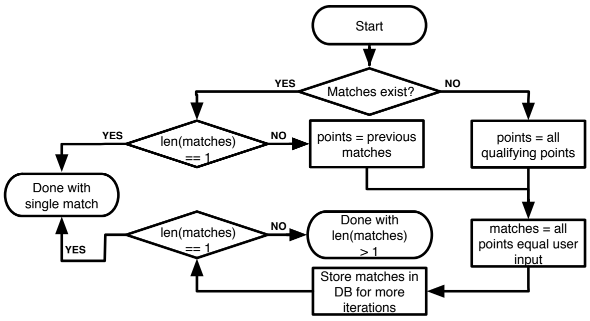

Start
Matches exist?
YES
NO
len(matches) == 1
points = previous matches
points = all qualifying points
Done with single match
len(matches) == 1
Done with len(matches) > 1
matches = all points equal user input
Store matches in DB for more iterations

Fig. 3. Babel Matching Process to Distinguish a Single Point.

for the same point is provided again until a single point is left. The point is enriched with metadata in the form of (i) the user selected location and (ii) information from the global component lexicon.

C. Implementation and Deployment

We use one of our campus buildings for implementation and deployment. The 141,000 s.f. 7-story facility contains mainly research spaces and classrooms. The building's BMS is connected to a PC which runs sMAP [3]. We implement a driver for sMAP that runs locally. Our driver uses pyBACnet to query the BMS and spawns a Web service for communicating with the main Babel component.

The main component (written in Go) offers a REST interface for smartphone clients. Most times the driver is idle. When Babel receives user input by a client, it dynamically creates a list of BACnet points that come into question for the user's input and fires up our driver with these points. This reduced list of points is based on BACnet type values and previously matched points. If the point is already part of an ongoing matching process, we continue the process with the already narrowed down points. We implement an Android smartphone client for Babel that connects to a structural model in the form of rooms and a lexicon of point descriptors following the Project Haystack naming convention and modeled in JSON-LD. A user of our app is able to select her location in the building and the type of point she wants to report (e.g., a light switch). She then only inputs the observed state (e.g., on).

V. Experimental Results

We evaluate our system with various micro- and macro-benchmarks. In the following, we present our experimental evaluation on our deployment building. The building is operated by Siemens Apogee, a proprietary BMS by Siemens. It provides a virtual BACnet interface that we interact with.

We scan for BACnet points on the building using pyBACnet. Our scan result shows that the building contains 7053 points. By observing the point names manually, we find that 365 of these points are part of the lighting system, while 6688 belong to HVAC. Analog Output and Binary Output points make for 87% of all points.

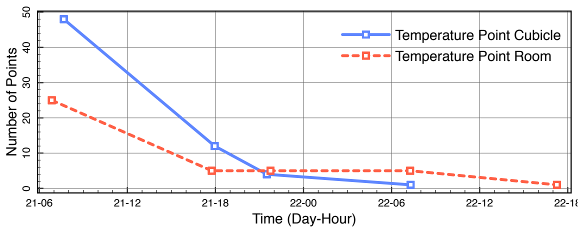


Fig. 4. Matching Progress for Two Thermostat Temperatures

We then sample all BMS points at different times of the day (9am, 1pm and 10pm) to be able to quantify how often the same value is shared among points. When many points have the same value, then a distinction based on the change of that value might require substantial more steps. Our experiments show that a value is shared on average among 4.8 other points. Few values make out the majority for all points (1 and 0 make up 42% of all points). The reason is that our system contains relatively many binary switched lights (365) and that HVAC systems use 1 and 0 to specify their heating and cooling mode.

A. Matching Process

We perform macro benchmarks for the point matching process by physically visiting several thermostats and light switches to perform user input to Babel over the period of one week. Figure 4 shows the result of this process for two thermostat temperature points. The points that need to be considered by Babel reduce over time, dependent on user input and the "grade of singularity" of that value. In depicted case, we are able to reduce the possible points to 48 and 25 with the first user input. In consecutive iterations, we only consider the remaining points. As can be seen for the room temperature (dashed line), the matching process might not be able to reduce the points if either (i) all other possible points have the same value or (ii) some of the points in the BMS do not respond to reading requests.

Figure 5 shows the same experiment for a point that can be directly influenced by users: a binary light switch. There are 365 such points in our deployment building. In contrast to the matching of the temperature setpoint, we can not reduce the number of points as significantly in the first iteration. But then we can observe the strength of our human-in-the-loop approach. By physically switching a light and reporting the new state we could always reduce to a single point during our experiments (100 tries for 20 different points).

To quantify the performance of our system at a larger scale, we have used historical data to simulate an ongoing matching process. The measurements have a 10s granularity. The dataset contains 463 different points. We use this data to perform a point matching in 10 minute time intervals. This means, that we assume that for all unmatched points, every 10 minutes, a user inputs her observations into our system. Figure 6 shows how the distribution of matched points increases over time. The result is a match of all points after 12

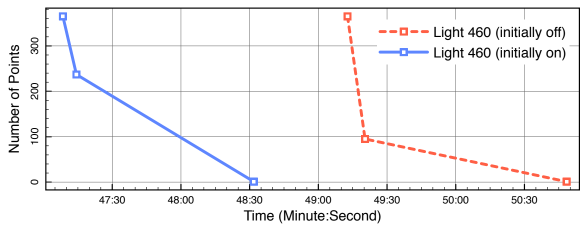


Fig. 5. Matching Progress for Two Light Switches

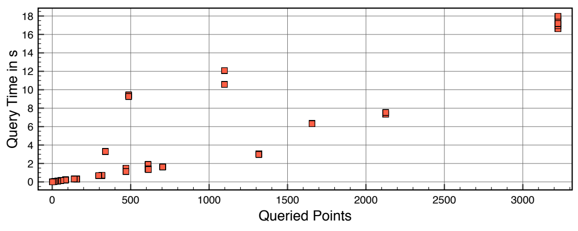


Fig. 7. BMS Query Latency

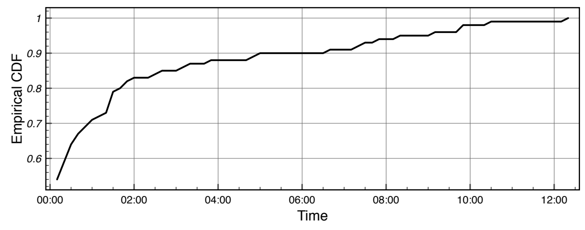


Fig. 6. Cumulative Distribution (CDF) of Successful Matches

hours. One must note that this experiment does not fully model a real deployment. First, this experiment does not contain any user initiated actuation (e.g., light switch, thermostat setpoint changes etc.) that will accelerate the matching process and second, we perform a matching every 10 minutes, which is not realistic for a real deployment. However, the results show that our approach leads to a quick point mapping when users participate sufficiently.

B. *End-to-end Latency*

We aim for an interactive system, where users get timely feedback on the completion of a matching process. First, low end-to-end latency can enable gamification as user incentive. Second, it enables users to engage in a sequence of rapid interactions (e.g., for lights). Letting users manually change the value of a point through a sequence of interactions with Babel can speed up the BMS metadata matching process.

The experienced end-to-end latency generally correlates with the number of points that need to be queried. Querying all 365 light points results in 5.9s of delay. The time reduces to 4.8 and 3.4s for 237 and 95 points respectively. We found the latency of the BMS the main dominating component for our system and perform therefore micro-benchmarks on it (see Figure 7). We are able to read a bulk of 3224 points in 17.3s (10 tries). The query time does not always correlate to the number of queried points. During our experiments, we found that the query time heavily depends on the physical BMS controller that connects the points. Sometimes querying more points takes less time. E.g., the controller that polls the 2100 points is significantly faster than the controller polling the 1100 points. Such unpredictability makes it challenging to implement an interactive system on top of a BMS.

C. *Accuracy*

Accuracy is crucial for cyber-physical applications and to overcome reservations by a building manager if a new, more user centric system is installed. If we assume that the state of physical readings by users does always correspond to the digital state in the BMS, then by universal instantiation, this must also be true for a single value that remains at the end:

$$\forall x\, P(x) \Rightarrow P(a/x) \tag{1}$$

This results in a 100% accuracy in an ideal system. However, in a real implementation, we need to consider (i) accidental or intentional wrong user input and (ii) that digital and physical states are not perfectly aligned in time and value dimension. Both can be dealt by building trust through multiple matches by several users. We leave extensive experiments with real occupants for future work, but we experienced that the physical state (e.g., of a light switch or temperature value) is not always equal to the value in the BMS. A thermostat was only able to display temperature as integer, while the value in the BMS was a two decimal floating point number. As a result, we convert values to integers, sacrificing some variance. We also found that physical artifacts do not always correspond one-to-one to BMS points. For instance we found a light switch that corresponded to two points in the BMS. Another problem occurs if BACnet points do not respond to reading requests. If this happens, we might remove the correct point from our list of possibilities. To avoid that, we keep points that do not respond in our list of possible matches.

D. *Lessons Learned*

The insight we gained over the course of our design and deployment towards the feasibility and performance of our approach is threefold. First, using a BMS as basis for user initiated interaction is feasible as our evaluation shows. However, we often face technological barriers. The BACnet read rate is limited. Considering that we are connecting to a virtual BACnet interface, we suspect a native BACnet to be even slower. Second, our approach has limitations when we deal with points that can not be observed directly by humans. The matching of internal setpoints (e.g., for the control flow of the HVAC system) is out of scope in our work, as these points are not directly relevant for many applications. Third, our matching process is considerably slowed down by a loss of variance due to different encodings of point values (e.g., on a thermostat display and on the BMS). We currently deal with this problem by using the lowest common denominator (e.g., convert to integer).

VI. Conclusion and Future Work

In this work, we developed and implemented the concept of crowd-sourced metadata maintenance through point matching for non-residential buildings. Inconsistent metadata in the building space is a problem that hinders the further development of cyber-physical applications that are portable across buildings. Inconsistent metadata is also a problem for traditional building operations. The introduction of a ESM is often expensive and requires manual work [14]. The traditional building commissioning is transforming into a continuous and iterative process that requires consistent metadata to stay cost-effective [22].

Our approach of introducing a human-link between the physical world and the digital point of a BMS works well in the experiments performed in our building. The current implementation is limited by the artifacts an occupant can observe, solving the most relevant metadata problems with regard to end-user applications. In addition, traditional appliances in residential homes are steadily replaced by smart appliances. It is inevitable that these appliances will reach the non-residential building market during the next years. This plethora of new devices will be much more interactive for humans and align itself well with our idea. Ultimately, we envision Babel as a general approach to metadata maintenance in the context of IoT.

Looking forward, we see several directions for future work. The obvious next step is to deploy Babel with actual occupants of a building to investigate to which extend our approach is an acceptable effort. Further, gamification is an ideal candidate for Babel. Ingress is a popular game (7 million players) by Niantic Labs (Google) where the goal of players is to conquer geographical areas in the real world by physically visiting these places (https://www.ingress.com). Combining Ingress's approach with metadata maintenance appears to be promising. Lastly, building commissioning is a process that often is only completed once when the building is constructed. A commissioning process that is driven by human input could propagate a continuous commissioning process where occupants keep the physical state of the building in sync with the digital.

Acknowledgment

This work is supported in part by the National Science Foundation under grant CPS-1239552 (SDB).

References

[1] U. E. P. Agency, "Building energy data book," 2011.

[2] U.S. EPA/Office of Air and Radiation, "The inside story: A guide to indoor air quality,," 1988.

[3] S. Dawson-Haggerty, X. Jiang, G. Tolle, J. Ortiz, and D. Culler, "sMAP: a simple measurement and actuation profile for physical information," in *SenSys'10*. ACM, 2010.

[4] V. L. Erickson and A. E. Cerpa, "Thermovote: participatory sensing for efficient building hvac conditioning," in *BuildSys'12*. ACM, 2012.

[5] B. Narayanaswamy, B. Balaji, R. Gupta, and Y. Agarwal, "Data driven investigation of faults in hvac systems with model, cluster and compare," in *BuildSys'14*. ACM, 2014.

[6] T. Haasl and T. Sharp, *A practical guide for commissioning existing buildings*. Oak Ridge National Laboratory, 1999.

[7] H.-C. Yang and C.-H. Lee, "Automatic metadata generation forweb pages using a text mining approach," in *WIRI'05*. IEEE, 2005.

[8] M. Rodriguez, J. Bollen, and H. Sompel, "Automatic metadata generation using associative networks," *TOIS'09*, 2009.

[9] A. Mathes, "Folksonomies-cooperative classification and communication through shared metadata," 2004.

[10] P. Neis, D. Zielstra, and A. Zipf, "The street network evolution of crowdsourced maps: Openstreetmap in germany 2007–2011," *Future Internet*, vol. 4, no. 1, pp. 1–21, 2011.

[11] A. Rice and O. Woodman, "Crowd-sourcing world models with openroommap," in *PERCOM'10*. IEEE, 2010.

[12] A. Bhattacharya, D. Hong, D. Culler, J. Ortiz, K. Whitehouse, and E. Wu, "Automated metadata construction to support portable building applications," in *BuildSys'15*. ACM, 2015.

[13] D. Hong, J. Ortiz, K. Whitehouse, and D. Culler, "Towards Automatic Spatial Verification of Sensor Placement in Buildings," *BuildSys'13*, 2013.

[14] A. Schumann, J. Ploennigs, and B. Gorman, "Towards automating the deployment of energy saving approaches in buildings," in *BuildSys'14*. ACM, 2014.

[15] buildingSMART, "IFC Standard," http://www.buildingsmart-tech.org/specifications/ifc-overview, 2015.

[16] Project Haystack, http://project-haystack.org/, 2015.

[17] ISO, "ISO 16484-3:2005," 2005.

[18] A. Bhattacharya, J. Ploennigs, and D. Culler, "Analyzing metadata schemas for buildings: The good, the bad, and the ugly," in *BuildSys'15*. ACM, 2015.

[19] E. Curry, J. O'Donnell, E. Corry, S. Hasan, M. Keane, and S. O'Riain, "Linking building data in the cloud: Integrating cross-domain building data using linked data," *Advanced Engineering Informatics*, 2013.

[20] W3C, "Resource Description Framework (RDF)," http://www.w3.org/RDF/, 2014.

[21] S. T. Bushby, "BACnet TM: a standard communication infrastructure for intelligent buildings," *Automation in Construction*, vol. 6, no. 5, pp. 529–540, 1997.

[22] E. Mills, *The cost-effectiveness of commercial-buildings commissioning: A meta-analysis of energy and non-energy impacts in existing buildings and new construction in the United States*. Lawrence Berkeley National Laboratory, 2004.

[23] J. Fürst, K. Chen, R. H. Katz, and P. Bonnet, "Demo abstract: Human-in-the-loop bms point matching and metadata labeling with babel," in *BuildSys'15*. ACM, 2015.

Chapter 4

Leveraging Physical Locality

In our first contribution, we developed a novel building interface for a building manager that relies on virtual reality. Our crowdsourced metadata maintenance system Babel allows to run such interface on top of existing building management systems by creating a consistent metadata state.

In this chapter, we switch our focus to the architectural issues that arise when off-the-shelf smart appliances are to be integrated in a non-residential building. We address authentication, authorization and networking issues. What guides our design throughout the following papers is Mark Weisser's vision of ubiquitous computing [15]. We aim to make such smart infrastructure truly ubiquitous by emphasizing on (i) an easy deployment by building management and users, and (ii) intuitive and effortless accessibility for current occupants. This chapter includes three papers.

4.1 Towards a Decentralized BMS

The first paper in this series is titled *"Leveraging Physical Locality to Integrate Smart Appliances in Non-Residential Buildings with Ultrasound and Bluetooth Low Energy"* and was presented at IoTDI'16 [6] and at the SDB winter retreat 2016. It describes the design, implementation and evaluation of BLEoT, a system that makes off-the-shelf smart appliances and sensors available to users in vicinity using Bluetooth Low Energy (BLE) for networking, and using acoustic communication to ensure a usable form of authorization. We use the user's smartphones as human-building interaction devices and as opportunistic gateways that enable local and remote networking. Ergo, with BLEoT, we explore a decentralized approach and user focused design of a BMS.

The main contributions of BLEoT lie in its decentralized system design, implementation and in the insight of coupling locality with usage. Our system is functional outside academia, by relying on off-the-shelf appliances and unmodified user smartphones.

To make smart appliances ubiquitous, we need to make interaction with them as simple as for traditional appliances. Authentication and authorization are often contrary to usability. Many smart appliances for the residential home rely on the implicit locality provided by the residential LAN. This locality is not guaranteed in a non-residential building. The LAN might cover multiple building parts or even multiple buildings. Further, non-residential buildings (like a university campus) are often characterized by a non-uniform, non-static set of occupants. Daily visitors are the norm and not the exception like in a residential home. Smart infrastructure should be available to all building users. Our insight is thus to explicitly reconstitute this locality by combining BLE with sound based authorization.

We decouple off-the-shelf smart appliances from their manufacturer cloud, restricting their communication to BLE through opportunistic gateways in form of user smartphones. Such a design gives the control and ownership back to the user of smart appliances. Current off-the-shelf smart appliances are siloed systems and coupled with the manufacturer's cloud server, often not for the benefit of the single user:

Such cloud-centric design does not allow the user to decide what data she wants to share and what data to keep private. Our architecture enables the user to keep control over collected data by restraining the flow of data on her smartphone. Apple's Health Kit [1] shows that such a design of user controlled flow of private data is already accepted for health related data in industry. We believe that this should not only be restricted to health data, but apply to all personal user data that is collected in the context of IoT.

The cloud-centric design also has serious implications to security. It means that many smart appliances will be constantly connected to the Internet. The variety of different Android smartphones is already not being updated frequently.[1] The update frequency and security state for the emerging IoT devices is even more severe (see [10] for an analysis). In our design, appliances are disconnected when occupants are not present.

BLEoT also shows that many of the factors that smart appliances promise to improve (e.g., personal comfort, energy consumption) can be solved at a local context. Device-to-cloud communication can thus in many cases be replaced by decentralized, local device-to-device connectivity. In future, we need to develop trade-offs for the necessary sharing and aggregation of data. Cryptography approaches like Homomorphic Encryption [7] might be a possible answer to this as systems like CryptDB have shown [11]. Differential privacy, a statistical approach, might be another approach to it [5].

4.2 Bluetooth Low Energy in the Wild

We choose Bluetooth Low Energy in BLEoT to connect appliances and sensors to the user's smartphone, because of its low energy consumption, but even more, because of its omnipresence on current smartphones.[2] This omnipresence, together with its single-hop, short-range characteristics, allow a user to directly interact with devices in her vicinity. Bluetooth 5 was announced recently and is expected to be released early 2017. It promises to quadruple the range, to double the speed and to increase data broadcasting capacity eightfold to further strengthen its applicability for IoT [4]. For these reasons, we expect that the importance of BLE for IoT is going to increase further in the near future.

However, during the implementation and deployment of BLEoT, we noticed erratic and nonuniform BLE behavior on current Android phones and tablets. As an example, when we implemented our first prototypes in 2014 and 2015, some smartphone models and Android OS versions required to turn 2.4GHz 802.11 off in order to achieve a stable BLE performance. Fortunately, many of the most severe issues have been improved with software fixes. Despite of these

[1]At the point of writing (June 2016), Android Marshmallow is only installed on 10.1% of all devices (see `https://developer.android.com/about/dashboards`).

[2]Apple introduced BLE with the iPhone 4S in 2011 [16]. Since then it has seen rapid growth and is in practice available on all current smartphones.

fixes, our experiences led us to investigate BLE performance in smartphone-peripheral systems.

Much work has been done in analysing and evaluating BLE as a communication protocol. Kamath and Lindh, Siekkinen et al. evaluate performance and energy consumption of BLE at chip level (SoCs) [8, 12]. Liu et al. analyse its discovery process [9] and Silva et al. focus on the interference with other protocols on the 2.4GHz band [13].

Surprisingly, there exists no "in the wild" evaluation of BLE in a smartphone-peripheral context. Modern smartphones are complex, multiprocessing machines. Their internals are at best only partly open (like on Android) or fully black-boxed (like on iOS). An application developer accesses their functions through multiple abstraction layers. Further, the actual process scheduling can only be influenced and not defined by the developer. In the end, the OS can interrupt, halt or even terminate a process according to available system resources or to reduce energy consumption. Another issue is the hardware variance. Manufacturers use different radio chips, different antenna and case designs.

If smartphones are to become an integral component of IoT and in opportunistic, ad-hoc sensor network designs, then the performance of actual BLE implementations needs to be better understood. This insight will allow an application to adapt itself dynamically to the oddities of the particular phones. Such dynamic adaption will make the BLE behavior on smartphones more deterministic and ultimately benefit embedded system designers, that are resource constrained with their designs. With BLEva, our second attached paper, we make a first step into this direction.

BLEva has several contributions: (i) We present a distributed evaluation framework for BLE on modern smartphones. (ii) We evaluate nine different smartphone models in our framework. (iii) We present an initial implementation of a dynamic BLE library that takes the characteristics of different smartphones into account.

Looking forward, we predict smartphones being progressively used as the user's personal tier between low power sensor and actuator infrastructure and the cloud. As such, the OS will need to provide more deterministic guarantees to help embedded designers to optimize energy consumption. An approach like we showed in BLEva, that adapts to the idiosyncrasies of specific BLE implementations can facilitate such behavior.

At the moment, we further work on establishing a simple methodology for cross-platform smartphone power monitoring applying the API only approach we choose in BLEva. Based on a survey of recent work of mobile based systems, we discovered that many researchers either externally measure power consumption at the battery terminals, or use non up-to-date and smartphone model dependent, software model based approaches. External monitoring is not applicable for many in the wild evaluations and increasingly unpractical due to closed case designs. This means, that many current power consumption results are not easily reproducible by other researchers. Thus, we believe that a reproducible methodology and toolchain for the power monitoring of continuous, smartphone based sensing and gateway applications is necessary.

4.3 From Physical to Logical Localization

The last paper of this thesis, titled *"A Practical Model for Human-Smart Appliances Interaction"*, combines personal smartphone sensors and human-input to dynamically detect logical relations between humans and smart appliances. As such, it extends the granularity of the physical, acoustic localization provided by BLEoT to dynamically enable relative queries of the form "increase the brightness at my current spot" and similar.

Our work builds on logical localization, taking our inspiration from SurroundSense's introduction of ambience fingerprinting [2] as alternative to RF based fingerprinting techniques. In contrast to SurroundSense and others, we apply logical localization in combination with a modifiable smart environment. We assume that some rough location context exists. This context could be provided in various ways: Through an acoustic, room-level based localization like in BLEoT, through WiFi-based localization (e.g., [3]) or merely through the physical signal propagation restrictions of BLE or similar RF protocols. In our system, this context then allows to read local values from sensors and modify actuators. By combining the possibility of environment modification with personal sensors (e.g., a smartphone's light sensor) and manual, user feedback, we are then able to grade human-smart-appliance relations according their impact on the individual. This allows for: (i) an optimization of energy efficiency as a global, building wide goal locally and (ii) for conflict mediation between disjoint comfort preferences of occupants.

As future work, we want to further investigate systems where humans are equipped with an increasing amount of different environmental and personal sensors (e.g., temperature, humidity, pulse). We believe that such personal sensors are a prerequisite to enable ubiquitous smart environments that adapt themselves to the individual comfort of the current occupants, eventually taking the human completely out of the loop. There are also many potential improvements in the networking and system architecture domain. Our current room-coordinator can be replaced by direct phone-to-phone communication. This will be further enabled through improvements in the upcoming Bluetooth 5 standard, like e.g., a much extended broadcasting capacity [4]. We want to investigate such local peer-to-peer networks and the trade-offs involved in the resulting distributed design.

Another interesting research direction is to model the different user comfort preferences and energy efficiency as multi-objective optimization. E.g., Sørensen et al. applied multi-objective optimization for conflicting requirements in the context of a greenhouse control system [14].

References

[1] Apple. HealthKit. `https://developer.apple.com/healthkit`. 2016.

[2] M. Azizyan, I. Constandache, and R. Roy Choudhury. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In *MobiCom'09*. ACM, 2009.

[3] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE.* Vol. 2. Ieee, 2000, pp. 775–784.

[4] Bluetooth SIG. Bluetooth 5 Coming Soon. `https://www.bluetooth.com/news/pressreleases/2016/06/16/-bluetooth5-quadruples-rangedoubles-speedincreases-data-broadcasting-capacity-by-800`. 2016.

[5] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[6] J. Fürst, K. Chen, M. Aljarrah, P. Bonnet, et al. Leveraging Physical Locality to Integrate Smart Appliances in Non-Residential Buildings with Ultrasound and Bluetooth Low Energy. In *IoTDI'16*. IEEE, 2016.

[7] C. Gentry. A fully homomorphic encryption scheme. PhD thesis. Stanford University, 2009.

[8] S. Kamath and J. Lindh. Measuring bluetooth low energy power consumption. *Texas instruments application note AN092, Dallas*, 2010.

[9] J. Liu, C. Chen, Y. Ma, and Y. Xu. Energy analysis of device discovery for bluetooth low energy. In *Vehicular Technology Conference (VTC Fall), 2013 IEEE 78th.* IEEE, 2013, pp. 1–5.

[10] K. Palani, E. Holt, and S. Smith. Invisible and forgotten: Zero-day blooms in the IoT. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops).* IEEE, 2016, pp. 1–6.

[11] R. A. Popa, C. Redfield, N. Zeldovich, and H. Balakrishnan. CryptDB: protecting confidentiality with encrypted query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*. ACM, 2011, pp. 85–100.

[12] M. Siekkinen, M. Hiienkari, J. K. Nurminen, and J. Nieminen. How low energy is bluetooth low energy? comparative measurements with zigbee/802.15.4. In *Wireless Communications and Networking Conference Workshops (WCNCW), 2012 IEEE*. IEEE, 2012, pp. 232–237.

[13] S. Silva, S. Soares, T. Fernandes, A. Valente, and A. Moreira. Coexistence and interference tests on a Bluetooth Low Energy front-end. In *Science and Information Conference (SAI), 2014*. IEEE, 2014, pp. 1014–1018.

[14] J. C. Sørensen, B. N. Jørgensen, M. Klein, and Y. Demazeau. An agent-based extensible climate control system for sustainable greenhouse production. In *International conference on principles and practice of multi-agent systems*. Springer, 2011, pp. 218–233.

[15] M. Weiser. The computer for the 21st century. *Scientific American*, 265(3):94–104, 1991.

[16] Wikipedia. iPhone 4S. `https://en.wikipedia.org/wiki/IPhone_4S`. 2016.

Leveraging Physical Locality to Integrate Smart Appliances in Non-Residential Buildings with Ultrasound and Bluetooth Low Energy

Jonathan Fürst*, Kaifei Chen†, Mohammed Aljarrah* and Philippe Bonnet*
*IT University of Copenhagen, †UC Berkeley
jonf@itu.dk, kaifei@berkeley.edu, {moal, phbo}@itu.dk

***Abstract*—Smart appliances and sensors have become widely available. We are deploying them in our homes to manage the level of comfort, energy consumption or security. While such smart appliances are becoming an integral part of modern home automation systems, their integration into non-residential buildings is problematic. Indeed, smart appliance vendors rely on the assumption that the Local Area Network (LAN) guarantees locality and a single unit of use/administration. This assumption is not met in non-residential buildings, where the LAN infrastructure might cover one or several buildings, and where several organizations or functional units are co-located. Worse, directly coupling smart appliances to the Internet opens up a range of security issues as device owners have very little control over the way their smart appliances interact with external services. In order to address these problems, we propose a solution that couples the use and management of smart appliances with physical locality. Put differently, we propose that smart appliances can be accessed via smartphones, but only from the room they are located in. Our solution combines opportunistic connectivity through local Bluetooth Low Energy (BLE) with an ultrasound-based method for room level isolation. We describe and evaluate a prototype system, deployed in 25 offices and 2 common spaces of an office building. This work opens up intriguing avenues for new research focused on the representation and utilization of physical locality for decentralized building management.**

***Index Terms*—IoT; buildings; middleware; Bluetooth; BLE; ultrasound;**

I. Introduction

Since the late 1990s, researchers have postulated that sensors and actuators equipped with computation and communication capabilities would become widely available [1]. Today, this vision has become a reality. Environmental sensors and smart appliances such as thermostats, light bulbs, power plugs and locks equipped with short range radios are available in retail stores. A large part of these smart devices is targeted at home automation systems. However, there is a case to be made for the deployment of such smart infrastructure in non-residential buildings.

Buildings account for roughly 40% of primary energy consumption in the US and in Europe while we spend more than 90% of our time inside them [2, 3]. Larger non-residential buildings are massively instrumented and equipped with a Building Management System (BMS) under the control of Facility Management (FM).[1] Direct influence of occupants is often limited to few, physical switches (e.g., light switches). However, various studies show that personal comfort and energy consumption can be greatly improved by giving occupants direct personal control and by raising their awareness of their environment (see e.g., [4]). Likewise, existing BMS could be greatly improved by gaining more insight through additional sensors and by receiving direct feedback from building occupants [5]. Medium and small commercial buildings on the other hand are normally run without a BMS [6]. Consequently, the deployment and federating of a smart infrastructure could be a cost-effective way to introduce user centric management in such buildings.

[1]The BMS centrally controls functions like HVAC or lighting.

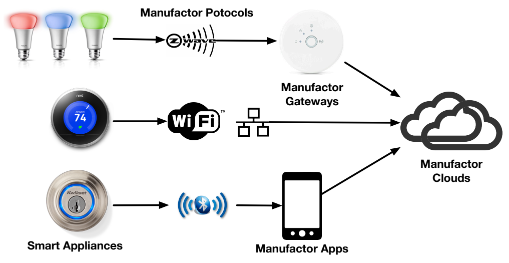


Fig. 1. Current Home Appliance IoT State

But the current IoT infrastructure does not align with the paradigms of non-residential buildings. First, despite being marketed the Internet of Things, the present ecosystem is characterized by manufacturer silos, with different protocols, different application gateways and different cloud infrastructures. Windley describes IoT as "The CompuServe of Things" [7]. We argue that this problem is even more severe when such devices are deployed in a business environment like a commercial building.

In principle, we can currently distinguish between three different IoT approaches to devices connectivity: (i) a low power protocol (e.g., ZigBee, ANT, Z-Wave) is integrated with the LAN infrastructure through a hardware manufacturer application gateway, (ii) the device is directly connected to the LAN (Ethernet/Wi-Fi), (iii) the device connects to a manufacturer application on the user's smartphone (see Figure 1).

A first consequence is that users do not have much control over appliances. Kaspersky Lab referred to IoT as the *"Inter-*

net of Crappy Things" [8]. Appliances invisibly communicate with the manufacturer server, possibly data that the users might not want to share. The point is that with current IoT approaches, users have no control over the sharing of data which might be personal.

Second, the connectivity options presented above directly expose appliances to the LAN or the Internet. A potential attacker might thus take over control over appliances remotely, even when the owner is not present.

Third, the owner is responsible of keeping the software on all devices up to date, while she must rely on the manufacturer to provide timely updates for security flaws. Such a system would become unmanageable in a non-residential building.

Fourth, the setup, deployment and access patterns of current IoT devices are an ill fit for (semi-) public and shared spaces like non-residential buildings. After deployment in the LAN, the user needs to establish a bond (authorization) with an application—usually on the users' smartphone—and a cloud service by the smart device manufacturer. This introduces the following problems:

- In a residential domain, the LAN guarantees locality. In a non-residential building, the LAN might cover the whole building, or even several buildings. The notion of physical locality is thus lost.
- At home, there is a uniform user domain (residents and guests). In a non-residential building, the LAN covers multiple functional units within an organization, and possibly multiple organizations. Above described setup process does not allow to configure who has access to a given smart device in a non-residential building.
- While there might be social tensions within a family to control the home automation systems, a home constitutes a single unit of administration. In a non-residential building, facility management, the IT department and the organizations occupying a given building constitute multiple overlapping units of administrations.

Put differently, the integration of smart devices relies on the amalgamation of physical locality and unit of use/administration. This coupling is implicitly defined by the LAN domain at home, and explicitly by a password based authentication. There is no implicit coupling in non-residential buildings, and the explicit coupling is cumbersome to achieve.

In order to address these problems, we must thus:

1) decouple smart appliances from the LAN or Internet. We propose to abstract the silo-ed cloud communication from traditional IoT solutions behind a generic and opportunistic gateway device: the user's smartphone.
2) define an infrastructure that explicitly reconstitutes the coupling between physical locality and unit of use/administration. Our insight is therefore to leverage Bluetooth Low Energy together with ultrasound based data communication to achieve authorization at the granularity of a room within a non-residential building.

In this paper, we present the design and implementation of a decentralized system of smart devices (sensor and actuator nodes) that are only intermittently connected through smartphones equipped with Bluetooth Low Energy: *BLEoT–The Bluetooth Low Energy of Things*. Our solutions makes smart appliances directly accessible for all users within BLE-range via smartphone. Authentication and authorization is achieved through the transmission of a key from smart devices to nearby smartphones using sound signals above the human hearing range that work well with off-the-shelf smartphones. Our solution does not require any changes in the buildings' IT infrastructure and does not necessitate an extended authorization process. Instead, it requires a simple deployment procedure managed by facility management and relies on authentication/authorization that is based on physical locality.

We studied how our design enables the deployment of smart devices with a prototype system deployed in 25 offices and two common spaces. Our experiments show that our solution was easy to deploy, easy to manage and that it opens up intriguing avenues for new research focused on the representation and utilization of physical locality in the Internet of Things.

In summary, the main contributions of our work are:

- The design of a solution to the problem of deploying smart appliances in non-residential building, that combines (i) the use of sound signals above the hearing range to achieve a secure access of smart infrastructure for close-by smartphones, and (ii) opportunistic communication via user smartphones equipped with Bluetooth Low Energy, that serve as generic links between local smart infrastructure and the cloud in the context of IoT.
- The implementation and evaluation of such a system in the scope of 25 offices and two common spaces.

The remainder of this paper is structured as follows. First we present background and related work. We then describe the design principles we followed, and overall system architecture before moving on to the detailed presentation of the two core components of our system: Acoustic channel and Bluetooth-based opportunistic gateways. Finally we discuss implementation and evaluation of our deployment in a non-residential building.

II. Background and Related Work

The two central components of our solution to connect smart appliances in a non-residential building are (i) opportunistic communication via Bluetooth Low Energy (BLE) and (ii) ultrasound communication. In this section, we present related work on smart appliances connectivity. We give a short BLE primer and discuss the use of Bluetooth for opportunistic communications. Finally, we discuss existing work where ultrasound is used for managing locality and data transmission.

A. Smart Appliances Connectivity

There has been a number of approaches from industry and academia focused on connecting smart appliances to home automation systems. Google is leading a consortium to develop a networking protocol for IoT called 'Thread' [9]. Thread builds on 6LoWPAN and creates a mesh network of up to 250 devices. Thread is targeting the home market and provides IP connectivity to all nodes. Apple recently introduced HomeKit, a framework for communicating with and controlling smart appliances [10]. HomeKit is built for the residential market, performing authentication based on the user's Apple ID. Our system distinguishes itself from these approaches by targeting non-residential buildings, where shared spaces require explicit coupling between use/administration domain and physical locality. We do not base our system on a fixed infrastructure like in Thread or Homekit, which does not scale to non-residential building, but on opportunistic gateways and self sustaining nodes.

The Californian start-up Zuli recently introduced BLE enabled smart-plugs, promoting a direct smart-plug to phone connectivity without the detour over a Wi-Fi network [11]. However their design follows a typical vertical vendor integration and smart plugs do not communicate their state with each other.

Zachariah et al. presented the idea of using mobile phones as routers for smart devices in [12]. They present the idea of using a participatory approach to bringing IPv6 to devices or to proxy their Bluetooth profiles to the Internet, but do not follow further with an implementation. With BLEoT, we have implemented a device bridge that creates Bluetooth profiles for REST APIs of several off-the-shelf smart appliances and opportunistic Internet gateways that enable communication between Bluetooth peripherals and between peripherals and Internet services.

Many prototypes and demonstrations in academia have been proposed for smart homes, including [13, 14, 15]. Mozer states in [14], that deployed smart systems need to inform users about their behavior. Functioning of devices need to be transparent to the user. Key challenges for home automation have further been addressed in [16], where Brush et al. list the following barriers: high cost of ownership, inflexibility, poor manageability, and difficulty achieving security. Brush also name convenience as a primary factor of user acceptance. With BLEoT, we use these observations as starting points for our design. Much work remains to be done to identify and address barriers to adoption in non-residential buildings.

B. Bluetooth Low Energy

We now describe briefly some of the key aspects and limitations of Bluetooth Low Energy (BLE) as they are important to understand our subsequent design decisions.

In BLE, data is exchanged asynchronously and limited to a single-hop in the 2.4 GHz band. BLE has two types of channels, advertising and data channels. If a device needs to only broadcast data, it can use the advertising channels to achieve a $1 : n$ unidirectional communication. Bidirectional communication takes place on the data channels between a central and a peripheral device, where peripherals usually provide services (server role) and centrals access these services (client role). Services are structured into characteristics that a client can read from or write to. This is managed by the Generic Attribute Profile (GATT). Following the publish-subscribe pattern, clients can get notified of value changes (notifications/indications). This can be used to push updated sensor values to clients for instance. The value that can be read and transmitted from a characteristic at a time is commonly limited to 20 bytes. Transmitting more than 20 bytes requires a split into multiple packages and possibly multiple read requests. A read/write request can only be initiated by a central device.

Park and Heidemann have shown that Bluetooth can be efficiently used as a support for data muling. During their deployment in several environments, they note that office spaces are a specially good fit for opportunistic mobility due to their dense sensors and long human loiter times [17]. They base their system on Bluetooth 2.0, using small embedded PCs and USB dongles. In contrast, our system utilizes custom battery powered sensor nodes, as well as commodity appliances and smartphones that communicate using Bluetooth Low Energy.

The recently announced Bluetooth 4.2 has the goal of establishing BLE as the wireless standard for IoT. It introduces a new profile enabling IPv6 for Bluetooth [18]. Nordic Semiconductor reacted by providing IPv6 over a Bluetooth protocol stack as well as a prototype of a IPv6 router, implemented on a Raspberry Pi [19]. Our implementation is not based on IPv6, instead we are proxying the existing APIs of smart appliances to native Bluetooth services.

C. Sound for Locality and Data Transmission

Sound signals have been used to achieve both, (i) localization/proximity and (ii) data communication.

Madhavapeddy et al. emphasize the adequacy of sound as a means to manage localization. Especially in buildings, walls prevent audio signals to be propagate outside a room, enabling room level localization [20]. Priyantha et al. have established the use of concurrent radio and sound signals to infer distance [21]. Borriello et al. then uses a combination of sound modulation and Wi-Fi networking (as communication channel) to achieve room level localization [22]. Finally, Lazik and Rowe use chirps in the ultrasound range to achieve localization [23].

Sound modulation for data communication has been studied extensively. Madhavapeddy et al. give an overview of using acoustic communication for both long range (telephone line) and short range (3m). For inaudible sounds (OOK on 21.2kHz carrier), they achieve 8bps [24]. Gerasimov and Bender explore different data encoding schemas for acoustic data transmission (echo coding, PSK, FSK and impulse coding) both in audible (5.5 kHz) and inaudible (18.4 kHz) frequency ranges. Their implementation works well for transmitter-receiver distances of up to 2m. Their maximum data rate is 3.4 Kbps when using multiple-level B-FSK in the 18.4 kHz range [25]. Nandakumar et al. propose a system to replace NFC communication with a secure, short range, sound based protocol that works in a range up to 20cm. Their operating bandwidth is 1kHz in the range of 6-7kHz. Their system relies on orthogonal frequency division multiplex (OFDM) with binary and quadrature PSK modulation for digital modulation and achieves 2.4Kbps [26]. Lee et al. develop an aerial acoustic communication by adopting chirp signals for digital modulation [27]. Their system works from 19.5-22kHz and achieves 16bps on a range up to 25m, while relying on a backend server to overcome the low data rate. Lopes and Aguiar develop a modulation schema that works in audible frequencies, but is more pleasant for humans by emulating sounds that humans are used to, like the ones of birds. They mention a data rate of 100-1000bps, but do not discuss the maximum range of their design [28].

In BLEoT, we are building on the experiences developed in aforementioned work. We achieve room locality through sound signals in the ultrasound range. We also use these signals to transmit an alternating key to close-by smartphones. We thus directly couple device access with physical access.

III. Design Principles

In this section, we discuss the principles that underlie the design of the system that we call BLEoT. These principles are articulated around three key requirements when deploying smart appliances in a non-residential building: security, ease of use and decentralized control.

A. Security Model

Our security goals are twofold. First, we want to ensure that only valid users are able to access smart devices. Second, we want to limit the possibility of a remote attack on the smart infrastructure. The assumption in our security model is that we trust the smartphone devices.

To ensure that only valid users can access devices, we couple their authorization with physical locality. The local coupling of acoustic data transmission and authorization maps thus physical authorization with digital authorization. Put differently, if a person has physical access to a room, we assume that she also has access to devices in that room. This is analogous to a person being able to switch the light in a room using a physical switch. This assumption is consistent with the security model of the physical space.

B. Deployment

Heavy deployment overhead has been one of the major challenges for smart systems and home automation in the past [16]. A key motivation of our work is to enable a quick and possibly temporary deployment of sensors and smart appliances in a large number of shared spaces. Ideally, neither facility management nor the IT department are involved in the deployment of smart appliances. Users should be able to (i) deploy smart appliances and make them available, and (ii) access any smart appliances they get close to.

A key aspect of our system is the definition of a common interface that abstracts existing devices. Devices are only able to communicate via that interface through the user's smartphone.

C. Decentralized Control

Deployed smart appliances and sensors need to follow a decentralized control logic, where operation does not break down with a connection loss to a central control unit.

Traditional Building Management Systems are usually following a three-tier model of management, automation and field level. At the lowest level, sensors and actuators communicate through a field bus (digital serial data bus) with each other and with control devices of the upper automation layer. Communication at the automation and management level usually takes place over LAN. Automation can happen locally via a direct coupling of sensors and actuators (occupancy and light), on the automation level (via direct digital controllers (DDCs)) or on the management level (building automation control computer) [29].

These layers of communication and control enable a building to be operated even when communication between local controllers and the central BMS computer breaks down. E.g., a direct coupling of a temperature sensor and a thermostat will be operational even in the absence of the centralized BMS. This approach to fault tolerance has been proven successful during several decades of building automation. A system of smart appliances and sensors should also be based on local control capabilities that do not require permanent connectivity with a central server.

Our design, based on opportunistic connectivity, pushes this logic and reverses the assumption: sensors and actuators are not connected to the upper automation layer unless a user with a smartphone equipped with the appropriate app enters the room where they are located.

IV. Design Overview

BLEoT aims to provide secure and usable access to smart appliances in non-residential buildings. BLEoT replaces existing manufacturer stovepipes with a gateway design. Edge-to-cloud data exchange becomes more visible and controllable by the user, via their smartphone.

A. Architecture

BLEoT consists of a loosely coupled three-tier architecture that uses opportunistic gateways to connect IoT devices with each other and with the cloud (see Figure 2). Locally, nodes act as BLE bridges for sensors and smart appliances. They advertise their service and state periodically via BLE. Nodes that bridge actuators contain a second, acoustic communication channel in the form of ultrasound. This channel is used to transmit a periodically changing key (e.g., every hour) that is required to access actuator capabilities on a node (see Figure 3). As such, we achieve a room level authorization based on the attenuation of sound waves by walls and doors. Nodes do not directly connect to each other or to remote services (e.g., the manufacturer cloud web service), but rely on opportunistic gateways—in form of smartphones. Using their BLE-enabled smartphone running BLEoT as gateways, users can interact with the environment (e.g., switching the light) in a room they enter and to read local sensor information (e.g., temperature).

All nodes are equipped with a Bluetooth radio, and some computation and storage capabilities. Nodes that connect to actuators are also equipped with a speaker. Nodes can connect

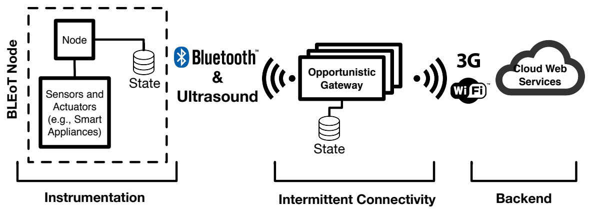


Fig. 2. Overall Architecture of BLEoT

to any sensor and actuator. We have for instance implemented several prototypes of native sensor and actuator nodes (see §VII). However, most importantly, our design allows to integrate various off-the-shelf smart appliances by bridging them from their native technologies and protocols to BLE services.

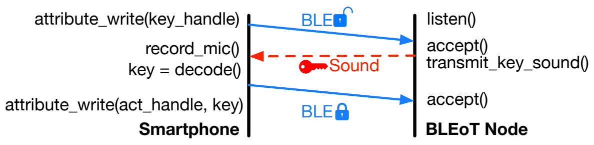


Fig. 3. Sound Based Authorization Process

V. BLE-Based Smartphone: A Gateway to Rule them All

This section discusses the design of our BLE based opportunistic gateway infrastructure. We describe how we abstract off-the shelf devices to a BLE interface and how we define the protocol that allows node-to-node and node-to-cloud communication.

A. *Bridging IoT Devices*

We abstract off-the-shelf smart appliances through a bridge to a BLEoT Node. Current off-the-shelf smart appliances provide different technologies, physical layers and APIs for access and are vertically integrated. Further, they do not always provide open and documented APIs.

Luckily, this problem of integrating several incompatible technologies, physical layers and (closed) APIs has been recognized and investigated earlier. Research projects as well as efforts originating in the open source community deal with their integration (e.g., [30, 31, 32]). BLEoT is building on the domain knowledge and technical insight gained in previous work (namely sMAP [30] and FHEM [31]). Both projects provide a rich repository of drivers for several existing IoT devices and the capability to write new ones. Such a device driver implements the specific protocol of the device and maps the device's functions to a local REST interface. Finally, this REST interface can then be accessed by applications independent of the device manufacturer interface (e.g., openHAB provides a UI application for residential homes [32]).

With BLEoT, we extend the current practice of REST integration by adding the capability of making that interface available locally via BLE to clients in proximity.

TABLE I
BLEoT Local Sensor Service

Characteristic	Access	Example	Description
Requested value	Write	`ed0000ff`	Value of requested local sensor value.

Taking the limitations of BLE into consideration, we have implemented two different approaches: (i) We simply tunnel the REST interface though BLE and (ii) we create native BLE services for each smart appliance.

While the latter approach (ii) seems like the better choice – because it is free of the HTTP overhead–, it causes difficulties when the set of smart applications connected to a bridge evolves in time. The commonly used Bluegiga BLED112 dongle requires for instance a reprogramming via USB DFU using a proprietary Windows-only update utility that writes a licence key on the chip. So we prefer the former approach (i). By tunneling the rest interface over a RX/TX pipe which we implement as a service on the BLE dongle, we achieve a generic interface. Standardized metadata that exists on the local REST interfaces becomes available on the smartphone client.

B. *Common Interface*

The interface between nodes and gateways consists of data communication taking place in pure advertisement state as well as communication taking place when a gateway is connected to a node.

1) Gateway Services: Each node exports BLEoT gateway services. Each service has a number of BLE read and write characteristics. Gateways access these services when a node forwards a request through its advertising message.

Table I shows the our service that allows basic node-to-node state transfer. A peripheral requests a sensor value from a defined set of abstracted types (e.g., temperature, humidity, CO_2 etc.). Opportunistic gateways serving that request will then directly write that value to the *Requested Value* characteristic.

Table II shows these characteristics for the BLEoT HTTP Service, while demonstrating a typical data offloading task that we implemented in our deployment. A URL characteristic provides the resource in form of a URL, while the HTTP method and payload are provided by other characteristics. Our specific payload for offloading sensor data consists of the sensor value (using IEEE 754 floats) and the sequence number of the measurement. Two writeable characteristics allow opportunistic gateways to acknowledge success in form of HTTP response codes and possibly write back the result in form of a HTTP message.

2) Advertisement Protocol: BLEoT nodes advertise their state, as well as requests for gateway services, regularly using a defined format on the advertising channels. This allows a $1 : n$ unidirectional communication between a node and close-by gateways. The data structure of the BLEoT advertisement can be seen in Figure 4. All data, necessary to populate a User Interface (UI) on the smartphone is encoded in the payload

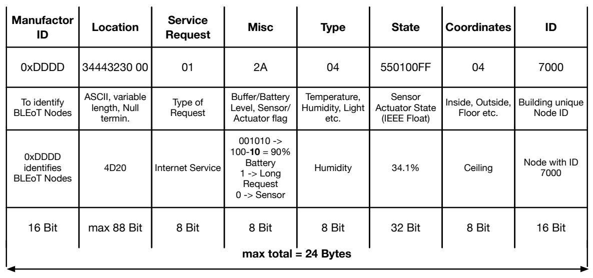
TABLE II
BLEoT HTTP Service

Characteristic	Access	Example	Description
URL	Read	130.226.142. 195/api/bleot/ addreadings	URL for the HTTP request.
HTTP method	Read	POST	HTTP method to use.
Payload	Read	`ed0000ff81ff` ... `81ff0100`	Payload for the HTTP request.
HTTP response	Write	201	HTTP response codes.
Message body	Write	not used	HTTP message (e.g., new configuration).

together with state and service request information for the gateway service:

- The **Manufacturer ID** allows to distinguish BLEoT Nodes from other BLE devices.
- The **Human Readable Location** enables a direct display of the node's location in a UI without requiring any remote connection. This makes each of our smartphone applications usable on every building.
- The **Service Request** allows a node to request of gateways to perform a HTTP service or local service in its behalf.
- A **Misc** flag encodes several information in binary: battery level, sensor/actuator and the expected length of a service request.
- The current **State** of the node is encoded as a IEEE floating point number (e.g., current sensor value).
- The **Coordinates** flag abstracts common placements of nodes in a room (e.g., `ceiling`, `outside` etc.).
- **ID** gives a node a unique ID inside a building scope.

The Bluetooth standard restricts BLE advertisement packets to 24 Bytes. This limits the maximum length of the dynamic, human readable location string to 11 Bytes in our protocol. But it is sufficient for representing a typical "building, room number encoding" (e.g., `ITU4D21`, `SODA410` to the user.



Manufactor ID	Location	Service Request	Misc	Type	State	Coordinates	ID
0xDDDD	34443230 00	01	2A	04	550100FF	04	7000
To identify BLEoT Nodes	ASCII, variable length, Null termin.	Type of Request	Buffer/Battery Level, Sensor/ Actuator flag	Temperature, Humidity, Light etc.	Sensor Actuator State (IEEE Float)	Inside, Outside, Floor etc.	Building unique Node ID
0xDDDD identifies BLEoT Nodes	4D20	Internet Service	001010 -> 100-**10** = 90% Battery 1 -> Long Request 0 -> Sensor	Humidity	34.1%	Ceiling	Node with ID 7000
16 Bit	max 88 Bit	8 Bit	8 Bit	8 Bit	32 Bit	8 Bit	16 Bit

Fig. 4. BLEoT payload

The example payload in Figure 4 shows how we use the protocol described above for one-to-many communication. The depicted node is located in the ceiling of room "4D20". The location is ASCII encoded, which allows direct display in a UI application. The node is currently requesting Internet services from gateways, its battery level is at 90% and the service request it has will take relatively long time for the gateway to complete. We have divided service requests in long and short lasting requests. Currently, only data-offloading is a long lasting request. This allows a gateway to utilize its inbuilt location service to decide if it should accept such a request (person is sitting at his/her office and phone does not move) or not (person is walking the hallway). Lastly, the exampled node is advertising a humidity sensor value of 34.1%.

C. Deployment Processes and Configuration

We expect that BLEoT Nodes will be mostly deployed by building users. Hence we can not expect that deployment requires a technical background with a deep understanding of Computer Science or the operation of a BMS. Our current implementation of the deployment and configuration process reflects this:

1) A user places the BLEoT node at the deployment location and pushes the configuration button on the node.
2) In the configuration tab of our smartphone app, the node is now shown ready to be configured. When connecting to the node we establish a bond, storing a long term AES-CCM 128bit key on the node and the smartphone. This key is necessary when a node needs to be reconfigured.
3) The user defines location (building, floor no. and room no.) and further narrows it down by choosing from predefined coordinates associated with that location (ceiling, floor, outside...). She then configures each single sensor and actuator of that node. For sensors, she can enable data-offloading, set sample, advertising frequency, buffer threshold and signal strength or leave them at their default. For actuators, she is able to define a generic control in the form of a default schedule for the space. In practice these are things like defining actuator set points for daytime and night-time, for weekdays and weekends. If there is no user in the space, then these settings will define the behavior of the space. To adjust the acoustic channel to the size of the space, we implement an adaption procedure, where the user moves at the room border and sound volume is adjusted to the quality of the received signal—now the node is operational.

The setup of off-the-shelf, smart appliances is slightly more complex. Before the above process, a BLEoT bridge is configured that connects all these devices and forms a BLE access point in form of a BLEoT node. Only then can the node be configured as usual.

Our design allows for an update of node configurations through opportunistic gateways. The node advertises on a regular basis its request (in the form of a HTTP service request) to get a possible new configuration. The gateway gets the configuration from the provided remote server and writes it back to the node. The message is encrypted server side using the AES-CCM 128bit keys that have been generated during the first bond with the smartphone of a trusted deployment person.

VI. Acoustic Channel

The acoustic channel between smartphone devices and BLEoT nodes implements room level authorization. As a requirement, our system must be capable to operate in indoor, (semi-) public spaces. BLEoT's correct operation depends on the acoustic channel as a physical medium. First, indoor spaces contain some level of ambient noise that causes interference to any acoustic communication. Second, indoor environments cause sound signals to echo from walls and other obstacles. The resulting multipath propagation represents another challenge. Finally, our requirement is to operate above the human hearing range and with off-the-shelf devices (smartphones).

A. Ambient Noise

Ambient noise differs on the type of indoor location. We categorize spaces into (i) smaller, delimited spaces like offices, (ii) spaces in which people pass through, like hallways and (iii) spaces of gathering (meeting rooms, cafeteria), where the ambient noise is expected to be higher. To qualify their noise level, we have taken sound samples on different periods of the day.

As our requirement is to work above the human hearing range, we pay special attention to ambient noise in higher frequencies. These might be due to the influence of high frequency sounds like those commonly emitted by switched-mode-power supplies. In Figure 5 we plot the Power Spectral Density (PSD) for different locations on our campus. The data was recorded using a portable USB condenser Microphone (Samson Go Mic). As can be seen, regardless of the environment, high spectral power can especially be observed up to around 6000kHz. High frequencies are not critically affected by ambient noise.

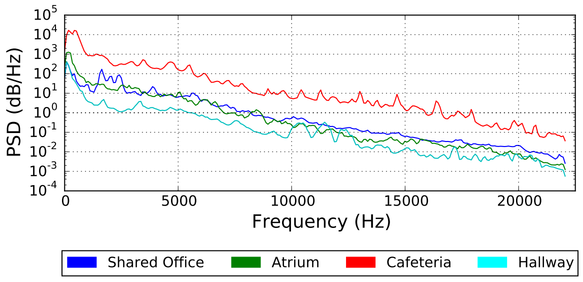


Fig. 5. Indoor Ambient Noise

B. Frequency Selectiveness

The human hearing range differs from person to person and depends on the applied audio pressure. Previous studies come to different results, reaching from a maximum of 19-20kHz ([33]) to 18kHz ([34]). Modern smartphones usually have an acoustic sampling rate of up to 44.1kHz, leading to a maximum frequency of 22kHz in theory. However, smartphone microphones are usually quite frequency selective and optimized for human voice frequencies (see e.g., [26]). We conducted experiments with several commonly used smartphones by playing a wide, linear chirp over the full frequency range (up to 22kHz). To minimize side effects of a non-flat speaker frequency responses, we used a studio monitor speaker (Audioengine 5+). Figure 6 shows our result for calculating PSD after Welchs method for several smartphones and tablets. As can be seen, the frequency selectiveness depends on device type and frequency. Higher frequencies contain a wider variance, but most devices are capable to record up to 21-22kHz. The result corresponds to the results obtained in [23] and [27].

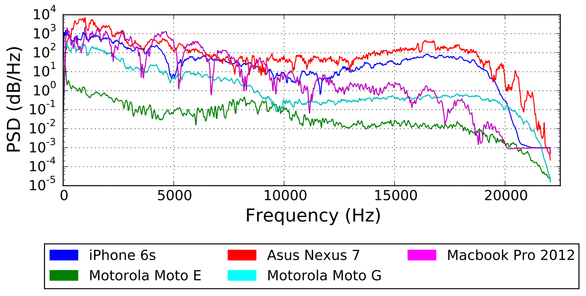


Fig. 6. Frequency Selectiveness

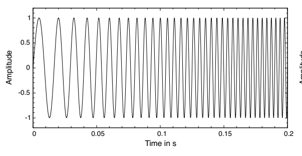
C. Generating the Signal

Our ambient noise experiments (see VI-A) have shown that high frequency ranges show little to no disturbance by ambient sounds. The sensitivity of smartphone microphones show however a high variance between models (see VI-B). Due to the high frequency selectiveness of smartphone microphones, we disregard any Frequency Shift Keying (FSK) based modulation schemas. Phase Shift Keying (PSK) schemas on the other hand perform bad in time-varying, fading channels like the acoustic channel [35]. We therefore follow the insight gained in [23] and [27] to use chirp signals for binary acoustic communication.

Compared to the time-invariance of FSK and PSK, a chirp signal varies its frequency over time. A chirp is a signal that linearly increases (up-chirp) or decreases (down-chirp) its frequency between two frequency ranges (see Figure 7). Chirp signals have been used extensively in sonar and radar applications. In the context of frequency selective microphones, a chirp signal has the advantage to use the same frequency range for up- and down-chirps. This means that both chirps are affected in a symmetric way.

Linear, up- and down-chirps are defined as follows:

$$s_{1,2}(t) = \sin\left[\phi_0 + 2\pi\left(f_0 t + \frac{f_1 - f_0}{2T}t^2\right)\right] \qquad (1)$$


(a) Up-Chirp

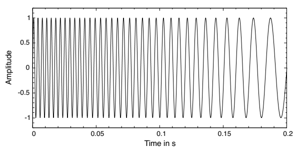

(b) Down-Chirp

Fig. 7. Up- and down-chirp signals between 50 and 300Hz and over a time interval of 0.2s.

Where:

- f_0 is the starting frequency.
- f_1 is the final frequency.
- T is symbol duration.
- ϕ_0 is the initial signal phase at $t = 0$.

Applying Equation 1 directly for modulating the signal results in a human perceivable clicking noise at the begin and end of the signal. This is due to the instant shift into high frequency ranges [23]. To remove this clicking effect we simply apply each chirp signal with a Hann window. Error detection is achieved by transmitting a parity bit at the end of the message.

To avoid multipath interferences, we add 50ms guard interval between symbols. We set our symbol duration to 100ms as our system is not expected to transmit longer messages via sound, but a small secret to allow clients to authenticate themselves with the BLEoT node. This results in a data rate of 7bps.

Acoustic power decreases by the square of the distance from the transmitter. This makes a one-fits-all power setting difficult. Ideally, the signal power changes with the receiver distance in a room. We therefore make use of the Bluetooth channel to achieve a more adaptive sound transmission by using RSSI at the transmitter as a coarse estimator.

RSSI roughly relates to distance as follows [36]:

$$RSSI = -10n \times log_{10}(d) - A \tag{2}$$

Where:

- n is the signal propagation constant.
- A is received signal strength at 1m distance (dBm).
- d is the distance between sender and receiver (m).

We experimentally define n and A for our combination of radios and indoor environment by taking RSSI measurements every meter from 1-25m. With $A = -59.947$, we then calculate n by inserting A in Equation 2 using the values from each experiment sample. This resulted in an averaged value of $n = 2.772$.

In the transmitter, we then multiply the signal power with the squared distance estimation.

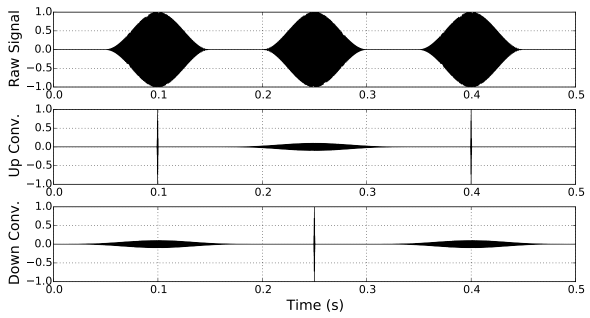


Fig. 8. Decoding `101` signal by convolving with the time reversed chirps.

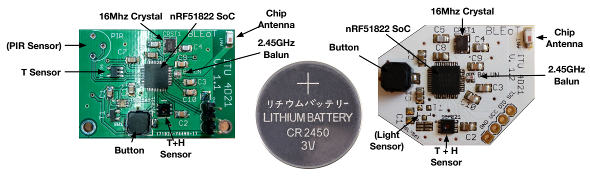


Fig. 9. BLEoT Sensor Nodes

D. Receiving the Signal

To decode the signal at the receiver, we apply a combination of convolution and peak detection. We generate both, up- and down-chirp signals, time reverse them and convolve them with the received audio signal. We pad the signal with 0s to achieve a complexity of $N * logN$ for FFT. After converting back to the time domain of the signal, we then perform peak detection on both results (received signal convolved with up-chirp and down-chirp). By merging both sets of detected peaks, we can then sort the values by time and thus decode the message.

Figure 8 shows a signal of `101` and the result of its convolution with the reversed chirp signals.

VII. System Implementation

We design and build custom nodes that connect sensors and actuators as well as a node that simply provide a Bluetooth bridge for off-the-shelf smart appliances.

The custom BLEoT nodes are implemented using the October 2014 revision of Nordic Semiconductor's nRF51822 SoC. It features a 32-bit ARM Cortex M0, 256kB of flash storage (of which the Bluetooth stack requires 80kB), 32kB of RAM and a 31-pin GPIO mapping scheme for analog and digital in- and output. We have designed and implemented several types of sensor (motion, temperature, humidity, light) and actuator (AC relay with Hall effect current sensor) boards using the nRF51822 (Figure 9 show our sensor nodes). The bulk of sensor nodes runs from a single 3V coin cell battery of type CR2450 which comes with a capacity of 620mAh.

Our IoT bridge is implemented using a Raspberry Pi Model B with a Bluegiga BLED112 dongle and a single, active speaker. The programmable dongle makes it a connectable peripheral device. We have experimented with several smart appliances. Our current deployment relies on Philips Hue light bulbs as personal desk lights, HomeMatic radiator thermostats to allow thermal changes and HomeMatic for switching arbitrary appliances. The Philips Hue lights rely on Z-Wave as the communication protocol, while the HomeMatic appliances use a proprietary protocol on the 868MHz band (BidCos).

A. *Gateways*

We implement opportunistic gateways in the form of an Android background service that is bundled with a UI application allowing users to control and sense their local space.

Once the application is started, the background service bootstraps necessary functionalities and maintains a global state. It relies on a fixed thread pool executor that will drive a number of internal threads concurrently. A "packet interceptor" thread is cyclically scanning for BLE advertisement packets, which then are handled by a "packet handler" thread that notifies observers (the UI) in case a valid BLEoT packet with new data is available. If the packet contains a service request flag, the thread will create a special "Service-Thread" and add it to the pool. This thread will indicate its accept of the service request by establishing a connection to the node and initiating the demanded request (e.g., data offloading its historic values). A "packet cleaner" thread is further removing dated packets (currently after 90s) and notifying the observers. This makes sure that possible occupant movement is reflected in the UI.

B. *Cloud Web Services*

We implement several Web services that nodes can access via opportunistic gateways: (i) data-offloading that allows nodes to offload their buffered sensor data, (ii) configuration retrieval to update a node's configuration. Both are implemented in Go (https://golang.org). The off-loading service takes POST requests that contain as their payload a node's buffered sensor data. Sensor data includes a sequence number for each sample that enables the logic on server side to calculate time stamps backwards, based on the known sample frequency. We plot historical sample data and make it accessible for other applications. The configuration retrieval service provides a new configuration for the node after a GET request is issued. Both services are protected by per node unique symmetric keys that are created when a node is deployed for the first time.

VIII. Results

We deployed our system in 25 (mostly shared) offices, two hallways/meeting rooms and a kitchen area at our university. The facilities consist of a long hallway with offices to the left and right. Meeting rooms are integrated in the hallway itself (see Figure 10).

We conducted experiments with BLEoT, as described in §VII, with various Android devices. Performance results varied. In the following, we show the results obtained with two devices that illustrate the breadth of the performance spectrum:

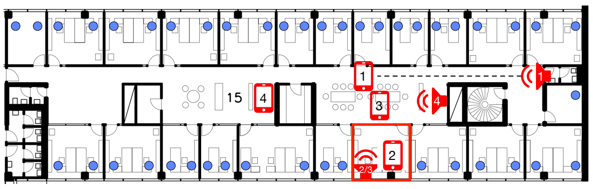


Fig. 10. Deployment with BLEoT nodes (circles) and experiments setup

(i) the Motorola Moto E smartphone and (ii) the Google Nexus 7 tablet.[2]

The most important performance measures are latency (for acoustic transmission and BLE), the performance of opportunistic gateway services (we evaluate data offloading as an example service), energy consumption (on battery powered nodes and smartphones) and finally the capability of our design to achieve a sound based room isolation. We conclude with a discussion on security aspects and a qualitative placement and discussion of our work.

A. *BLE Channel*

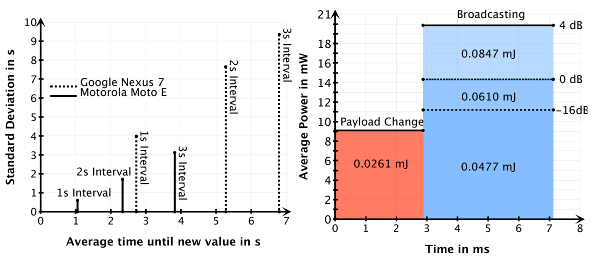
1) Latency: Several latency metrics are relevant for our work. Latency of actuation and the retrieval of local sensor values (state) is important to end-users (humans are able to perceive switching delays greater than 100ms [37]). The latency of opportunistic gateway services is ultimately important for the working of our system: If the state of a node drifts too far away from the state of its environment, its data might become irrelevant; if it is not able to offload its data, then data might be lost.

Our experiments have shown that latency of state transmission through advertisements depends heavily on the specific smartphone and on the advertising interval that has been chosen on the nodes. Figure 11a shows a growing latency as a result of a growing advertising interval. The standard deviation for the Nexus 7 is much higher than the Moto E.[3] Due to these erratic results between different devices, we suggest to set the interval to a maximum of 3s, which in our tests has been sufficient to keep the maximum latency between advertisements below 8s while still preserving battery.

Further, we measured the experienced latency when actuating bridged, off-the-shelf smart appliances. It took on average 0.6s to discover a new BLEoT node (with advertising interval set to 30ms). It then takes 3.74s to receive a 16bit key via the acoustic channel and the services of the node. Now, actuation can be done relatively quick. We measured the overhead introduced by our system in the range of 70 to 130ms. Node discovery and service retrieval only needs to be performed when a user enters a space for the first time. Afterwards, actuation occurs timely.

[2]See http://www.motorola.com/us/smartphones/moto-e-2nd-gen/moto-e-2nd-gen.html and http://en.wikipedia.org/wiki/Nexus_7_(2013_version)

[3]The unpredictability of the Nexus 7 seems to be a known problem in the Android community without a fix (https://code.google.com/p/android/issues/detail?id=65863).



(a) Receiver Latency (b) Energy Consumption

Fig. 11. Energy Consumption and Receiver Latency for Advertising.

The latency introduced by a opportunistic gateway (node-to-cloud and node-to-node) depends mainly on its availability. We have not yet performed an evaluation with an extended user base ($n > 10$). However we argue that the need for a gateway is highest when people are actually occupying a space. If a space is not used, it can be run in a default, unoccupied schedule. Data offloading can become critical during times of holidays. The maximum available flash storage of 156kB on our implemented sensor nodes allows for the storage of 194500 sensor values (8 bytes each). A node with a single sensor and 1 minute sample period will be able to persist values over a period of more than 16 days. This is in many cases sufficient for longer unoccupied periods.

2) Data Offloading: Data offloading is a data intensive service. We conducted experiments both, right next to the offloaded sensor node and at a distance of approx. 20m. 3258 sensor values were offloaded for 10 re-runs, taking 82msec/value on average for the close offloading and 100msec/value on average for the 20m distant sensor node. Moving the gateway further away from the sensor node resulted in connection dropouts.

These results fit well with our deployment in an office space in a university building. It takes around 5 minutes to continuously offload 3258 values. This seems much, but it does not worsen general operation due to the long loiter times in office spaces. When deploying our system in another context, buffer threshold needs to be adjusted to the specific environment and its occupation model.

B. Energy

We evaluate energy consumption for our battery powered sensor nodes and for the opportunistic gateway services. Energy consumption is mostly relevant for BLEoT sensor nodes that run from battery. Nodes that bridge off-the-shelf smart appliances are connected to the mains.

Our test equipment consists of a digital oscilloscope (Rigol DS1054Z) that we combine with an op-amp circuit to amplify a voltage signal at a small burden resistor. Due to the spread in consumption of BLE (from a few μA during sleep to $15mA$ during peak current when transmitting), we are using different burden resistor sizes (1Ω for higher currents and 1000Ω for sleep currents). Together with our amplification circuit of factor 100, this gives us a range of $0-20$mA and $0-20\mu$A for respective resistors.

We have measured consumption for the main operational events of our nodes. Figure 11b shows averaged results of 100 broadcasting and payload changes. The energy spent on payload changes is independent on TX power. Energy consumption for sensing is mostly defined by the specific sensor type and we thus omit it here. However persisting a value to flash is independent of the chosen sensor. We have measured that storing a single measurement requires 5.9μJ. An important event for our opportunistic design is node data collection by a gateway. In such a case nodes are transferring relatively big amounts of data in chunks of 22 bytes BLE data packets. A single connection event takes on average 3.05s with a consumption of 1.11mJ/s. This adds up to a total consumption of 3.39mJ for establishing a connection. The consumption for offloading the data depends on how full the buffer is. Our offloading mechanism uses Bluetooth indications, which allow GATT servers (peripherals) to push new values to the client (central device) without the need of poll requests by the client. We have measured 0.21mJ per indication. As one data packet in an indication fits two of our measurement data structures (8 bytes each), we end up with around 0.1mJ per transmitted measurement. For transmitting n measurements the consumption model thus becomes: $3.39\text{mJ} + n \cdot 0.1\text{mJ}$. A sensor node with a single sensor, a sample frequency of 1min and advertising frequency of 3s will thus run well over a year from a single CR2450 battery.

To evaluate the impact on smartphone energy consumption, we use "GSam Battery Monitor". We run our gateway service for a period of one week on the Motorola Moto E. This resulted in a battery impact of 14.8% on the phone's battery. During this time, the phone was offloading 68424 sensor values to the backend. We expect that the battery impact of such applications will drop in the near future, as Bluetooth connected devices become ubiquitous and hardware and OS support is being optimized for it. Further, to incentivize the use of our gateway service, we allow users to decide if the service should only be active when the UI application is in forefront. Our experience has shown that the power consumption is then mainly determined by the display.

C. Acoustic Channel

We now look at the performance characteristics of the acoustic channel.

1) PRR and Distance: To evaluate the Packet Reception Ratio (PRR), we transmit a message of 52bits from different distances (every 1m, from 1 to 10m) at different locations. Figure 12 shows the result doing this experiment in our section hallway (see Figure 10, No. 1) an open space (our university's atrium) and a shared office (No. 2). In open space we reach 30m before PRR drops substantially due to signal overlappings introduced by multipath (see Figure 13). In the hallway, these overlappings occur much earlier (around 16m). We can cover the $18.5m^2$ office fully with our signal. Extending the range is possible by increasing the guard interval between signals.

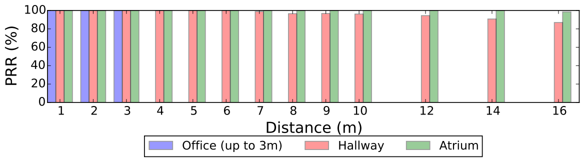


Fig. 12. PRR for different locations and as a function of distance.

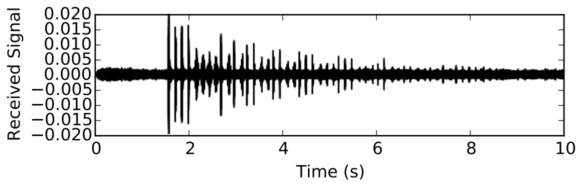


Fig. 13. Multipath effect on a signal at 40m distance.

Initial experiments when doubling the interval show that our approach can reach $> 40m$, which is more than the authors in [27] have achieved.

2) Room Isolation and Multipath: The main purpose in our design of using an acoustic transmission channel is to achieve isolation which we then map to device authorization. To measure the effectiveness of this isolation, we perform experiments with different pathways between transmitter and receiver (see Figure 10: No. 2, 3 and 4). Our experiments show that the acoustic channel easily covers the whole room, while a device listening at the wall and door, outside of the room is not able to pick up the signal. We successfully tested room isolation with different materials commonly used in buildings (concrete, wood, glass and polymer). An open door however, makes the signal available. To minimize this effect node configuration can be introduced to adjust the signal power to the room size. An attacker with a device with a high gain amplifier is however still able to pick up the signal.

D. Security Analysis

We use the concept of physical locality to authorize local actuation and access to sensor data. By basing access on locality in the physical world, we move the challenge of achieving a secure system out of a pure software implementation and into the physical security of a space. Security is therefore mainly dependent on how access to that space is controlled. Acoustic waves do not stop at open doors or windows. This means that our system further depends on the structure and location of a space.[4] Security also depends on the social structure that is prevalent. Are occupants likely to fiddle with other's instrumentation? During our deployment, one individual was occasionally actuating other's instrumentation as a friendly joke. Studying such scenarios in the context of a larger deployment is future work.

[4]E.g., A ground office is different than an office on the top floor.

We use symmetric encryption to exchange data between nodes and Web services, via smartphone gateways. Keys are created and exchanged when a node is deployed. During the deployment phase, local sniffing attacks might be conducted. After keys are exchanged, we depend on the security of the AES encryption. Mitigating the attack window during deployment is difficult. Nodes would need to have extended human input capabilities to enter a code directly on the node. Because of the short attack window and the exclusively local scope of the attack, we consider it a manageable threat.

IX. Conclusion

The BLEoT infrastructure is a first step towards integrating smart appliances in the context of non-residential buildings. Using ultrasound as a means to establish physical locality, BLEoT makes it possible to deploy and access smart appliances within non-residential buildings. Our results show that this approach is technically viable. Much work remains to be done to explore how BLEoT could complement an existing Building Management System (and possibly replace BMS in small buildings). More specifically, the key issue left as future work is the study of centralized building operation based on instrumented spaces that are only available when occupied by users with smartphone gateways.

In future work, we want to install our BLEoT sensors around the campus to perform larger user tests, to experiment with the acceptance of our system and to measure actual data loss due to the uncertainty in the system. We further want to experiment with the (temporary) deployment of sensors to improve the actual BMS of our campus.

Another interesting research area is to study how people will behave when they have the possibility to control smart appliances via smartphones. Having a larger installation, we will be able to evaluate to which extend an adaptive environment actually helps to increase comfort and reduce energy consumption. We want to compare an adaptive, user controlled setting with a central, model based system in terms of consumption and comfort.

References

[1] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. ACM, 1999, pp. 263–270.

[2] L. Pérez-Lombard, J. Ortiz, and C. Pout, "A review on buildings energy consumption information," *Energy and buildings*, vol. 40, no. 3, pp. 394–398, 2008.

[3] U.S. EPA/Office of Air and Radiation and Consumer Product Safety Commission, "The inside story: A guide to indoor air quality,," 1988.

[4] G. Brager, G. Paliaga, and R. De Dear, "Operable windows, personal control and occupant comfort." *Center for the Built Environment*, 2004.

[5] A. Piette, K. Kinney, and P. Haves, "Analysis of an information monitoring and diagnostic system to improve building operations," *Energy and Buildings*, vol. 33, no. 8, 2001.

[6] S. Katipamula, R. M. Underhill, J. K. Goddard, D. Taasevigen, M. Piette, J. Granderson, R. E. Brown, S. M. Lanzisera, and

T. Kuruganti, “Small-and medium-sized commercial building monitoring and controls needs: A scoping study,” *PNNL, Richland, WA (US), Tech. Rep*, 2012.

[7] P. Windley, “The CompuServe of Things,” http://www.windley.com/archives/2014/04/the_compuserve_of_things.shtml, 2014.

[8] Kaspersky Lab, “Internet of crappy things,” https://blog.kaspersky.com/internet-of-crappy-things/7667/, 2015.

[9] Thread Group, http://threadgroup.org, 2015.

[10] Apple HomeKit, https://developer.apple.com/homekit/, 2005.

[11] Zuli Smartplug, https://zuli.io, 2015.

[12] T. Zachariah, N. Klugman, B. Campbell, J. Adkins, N. Jackson, and P. Dutta, “The internet of things has a gateway problem,” in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*. ACM, 2015, pp. 27–32.

[13] S. S. Intille, “Designing a home of the future,” *IEEE pervasive computing*, vol. 1, no. 2, pp. 76–82, 2002.

[14] M. Mozer, “Lessons from an adaptive house,” Ph.D. dissertation, Architectural Engineering, 2004.

[15] C. D. Kidd, R. Orr, G. D. Abowd, C. G. Atkeson, I. A. Essa, B. MacIntyre, E. Mynatt, T. E. Starner, and W. Newstetter, “The aware home: A living laboratory for ubiquitous computing research,” in *Cooperative buildings*. Springer, 1999, pp. 191–198.

[16] A. Brush, B. Lee, R. Mahajan, S. Agarwal, S. Saroiu, and C. Dixon, “Home automation in the wild: challenges and opportunities,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2011, pp. 2115–2124.

[17] U. Park and J. Heidemann, “Data muling with mobile phones for sensornets,” in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, 2011, pp. 162–175.

[18] Bluetooth Core Specification 4.2, 2014. [Online]. Available: https://www.bluetooth.org/en-us/specification/adopted-specifications

[19] Nordic Semiconductor, “IPv6 over Bluetooth Smart,” http://www.nordicsemi.com/eng/News/News-releases/Product-Related-News/Nordic-Semiconductor-IPv6-over-Bluetooth-Smart-protocol-stack-for-nRF51-Series-SoCs-enables-small-low-cost-ultra-low-power-Internet-of-Things-applications, 2014.

[20] A. Madhavapeddy, D. Scott, and R. Sharp, “Context-aware computing with sound,” in *UbiComp 2003: Ubiquitous Computing*. Springer, 2003, pp. 315–332.

[21] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, “The cricket location-support system,” in *Proceedings of the 6th annual international conference on Mobile computing and networking*. ACM, 2000, pp. 32–43.

[22] G. Borriello, A. Liu, T. Offer, C. Palistrant, and R. Sharp, “Walrus: wireless acoustic location with room-level resolution using ultrasound,” in *Proceedings of the 3rd international conference on Mobile systems, applications, and services*. ACM, 2005, pp. 191–203.

[23] P. Lazik and A. Rowe, “Indoor pseudo-ranging of mobile devices using ultrasonic chirps,” in *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*. ACM, 2012, pp. 99–112.

[24] A. Madhavapeddy, R. Sharp, D. Scott, and A. Tse, “Audio networking: the forgotten wireless technology,” *Pervasive Computing, IEEE*, vol. 4, no. 3, pp. 55–60, 2005.

[25] V. Gerasimov and W. Bender, “Things that talk: using sound for device-to-device and device-to-human communication,” *IBM Systems Journal*, vol. 39, no. 3.4, pp. 530–546, 2000.

[26] R. Nandakumar, K. K. Chintalapudi, V. Padmanabhan, and R. Venkatesan, “Dhwani: secure peer-to-peer acoustic nfc,” in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 63–74.

[27] H. Lee, T. H. Kim, J. W. Choi, and S. Choi, “Chirp signal-based aerial acoustic communication for smart devices,” in *Proc. of IEEE Conf. on Computer Communications (INFOCOM), Hong Kong SAR, PRC*, 2015.

[28] C. V. Lopes and P. M. Aguiar, “Aerial acoustic communications,” in *Applications of Signal Processing to Audio and Acoustics*. IEEE, 2001, pp. 219–222.

[29] H. Merz, T. Hansemann, and C. Hübner, *Building Automation: Communication Systems with EIB/KNX, LON and BACnet*. Springer Science & Business Media, 2009.

[30] S. Dawson-Haggerty, X. Jiang, G. Tolle, J. Ortiz, and D. Culler, “sMAP: a simple measurement and actuation profile for physical information,” in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2010, pp. 197–210.

[31] Fhem, http://fhem.de, 2015.

[32] OpenHab, http://www.openhab.org, 2015.

[33] C. J. Plack, *The sense of hearing*. Psychology Press, 2013.

[34] B. C. Moore, *An introduction to the psychology of hearing*. Brill, 2012.

[35] A. J. Berni and W. D. Gregg, “On the utility of chirp modulation for digital signaling,” *Communications, IEEE Transactions on*, vol. 21, no. 6, pp. 748–751, 1973.

[36] Q. Dong and W. Dargie, “Evaluation of the reliability of rssi for indoor localization,” in *Wireless Communications in Unusual and Confined Areas (ICWCUCA), 2012 International Conference on*. IEEE, 2012, pp. 1–6.

[37] B. Shneiderman, *Designing the user interface-strategies for effective human-computer interaction*. Pearson, 1986.

Evaluating and Improving Bluetooth Low Energy Performance in the Wild

Jonathan Fürst
IT University of Copenhagen
jonf@itu.dk

Kaifei Chen
UC Berkeley
kaifei@berkeley.edu

Philippe Bonnet
IT University of Copenhagen
phbo@itu.dk

ABSTRACT

Despite its growing significance as a short-range, single-hop protocol for phone-to-peripheral communication in the context of IoT, the system performance characteristics of Bluetooth Low Energy (BLE) are not well understood. Existing performance studies focus on BLE peripherals, but not on overall smartphone-peripheral systems. Unlike peripherals, smartphones are complex, multiprocessing systems with different radios and IO capabilities. Their architecture and implementation is at best partially open, like on Android OS. Application developers can only access low-level functionalities through multiple layers of OS and hardware abstractions. Therefore, we designed and implemented a simple, portable and scalable benchmarking framework for evaluating BLE performance on smartphones. We propose several microbenchmarks derived from two common BLE applications, for which we perform an extensive evaluation on a variety of modern smartphones. Our evaluation characterizes existing devices and gives new insight about peripheral parameters settings. We derive the need for a dynamic, model-dependent setting of BLE parameter patterns on smartphones. Our initial implementation of such a library shows much improved results over the default BLE implementation on Android, improving PRR of advertisements up to 20-fold for some models and mean iBeacon accuracy by 2 m.

1. INTRODUCTION

Marc Weiser's vision of ubiquitous computing, formulated twenty years ago, has finally become a reality. We can all use our smartphone to interact wirelessly with smart devices in our surroundings. A core enabler technology for the realization fo this vision is Bluetooth Low Energy (BLE). BLE allows an energy efficient, local data exchange between smartphones and power, computationally constrained devices, enabling new, user-centric, cyber-physical applications.

BLE is a single hop, short range communication protocol, adopted into the Bluetooth Core Specification 4.0 in 2010. Since then, it has experienced rapid growth due to its tight bond with the fast growing smartphone market. Today, most smartphones support BLE and make it available to developers through high level SDKs. There are now billions of devices with BLE connectivity in application areas such as proximity sensing (iBeacons), fitness/sport tracking (e.g. Fitbit, smart watches), healthcare and smart home [5].

Despite its wide adoption, the performance characteristics of BLE systems are still largely undocumented. This leads to erratic application behaviours and underutlisation of resources. Existing research has focused on (i) the performance and energy consumption characteristics of BLE at chip level (SoCs) (e.g., [25, 35]), (ii) a detailed analysis of the Bluetooth discovery process (e.g., [28]) and (iii) interference with other protocols on the 2.4GHz band (e.g., [36]). However, all this work does not evaluate BLE in its main implementations: smartphone centric systems. In order to understand the performance characteristics of BLE on smartphones, a more holistic, system approach is required. In contrast to peripherals (e.g., sensors and actuators) that are increasingly implemented using a single SoC, smartphones non deterministic. They are powerful, multiprocessing machines that in their great majority run a Unix based OS (Android OS and iOS). They include a variety of closed hard- and software implementations and these implementations are often manufacturer specific. As a result, it is unreasonable to expect that complex smartphones behave like the SoC they contain.

We particularly motivate our work with the experience gained from our own BLE based deployment (see [13]). In this deployment, we made use of smartphones as opportunistic gateways between BLE-connected sensors, actuators and cloud services. We noticed, that the provided OS abstractions often fail to generalize certain model dependent performance characteristics. This results in varying and erratic performance for different models. E.g., a wide latency variance, failures to establish a connection, and even the impossibility to use any BLE functions at all without completely disabling 802.11 on the smartphone. This lack of performance guarantees across models is a problem for applications.

In this work, we develop a benchmarking framework for Android in combination with BLED112 dongles by

Silicon Labs (TI CC2540 based) to create a reproducible baseline for different BLE hardware and software implementations on smartphones. Based on our past experience with BLE and the emerging iBeacon use case, we design microbenchmarks and experimentally evaluate BLE performance on 9 smartphones models.

The main contributions of this paper are the following: (i) We analyse the working of BLE on Android by tracing high-level API calls down the OS stack. (ii) We design and implement a distributed benchmarking framework to easily perform BLE related benchmarks on Android. (iii) We design a set of benchmarks that model common usage scenarios of BLE. (iv) We perform an extensive evaluation on a variety of smartphones. (v) We abstract our results to a set of best BLE practices for different smartphone models that we make available for application developers. This improves PRR for some models 20-fold and iBeacon accuracy by a mean of 2 m for all models.

The remainder of the paper is structured as follows. First, we motivate our work by providing background on BLE and its hardware and software implementation on Android. We then present design and implementation of our framework and benchmarks in Section 3. Section 4 describes our results on 9 different smartphone models. We abstract these results to an adaptive BLE library in Section 5 that improves PRR and RSSI based distance estimation accuracy. Section 6 presents related work and Section 7 concludes our paper.

2. BACKGROUND AND ANALYSIS

In this section, we briefly summarize the working of BLE and its implementation on Android. We then describe our two motivational application areas.

2.1 Bluetooth Low Energy

BLE was designed as a RF standard with low power consumption, low cost, low bandwidth and low complexity. In BLE, data is exchanged asynchronously and limited to a single-hop in the 2.4 GHz band. The band is divided into 40 channels of which 3 are used for establishing connections, advertising and broadcasting (advertising channels) and 37 are used for connection data (data channels). To minimize interference by other signal sources (802.11, 802.15.4, classic Bluetooth), BLE applies time synchronized, adaptive frequency hopping spread spectrum (AFH) in its connection-oriented communication. The signal itself is modulated using Gaussian Frequency Shift Keying (GFSK) with a modulation rate of 1 Mbit/s.

The Generic Access Protocol (GAP) of BLE defines four important device roles: (i) The *Broadcaster* role, in which a device solely broadcasts. (ii) The *Observer* role, in which a device solely scans and listens to these broadcasts. (iv) The *Peripheral* role, in which a device advertises and is allowing connections to it. (v) The *Central* role, in which a device scans and connects to Peripheral devices.

If a device needs to only broadcast data, it can use the advertising channels to achieve a 1 : n unidirectional communication. Each advertising packet has a maximum payload of 31 bytes. The advertising channels are stateless, resulting in the fact that advertisements can be missed by a scanner. The probability of successful reception depends on the advertising and scanning intervals and the differential in their start time. For example, an advertiser might advertise sequentially on all three channels every 200ms. Then a scanner that only scans with a scan window of 25ms at an interval of 100ms on single channel might potentially miss most of the advertisements if advertising and scanning intervals are not coincidentally aligned.

Bidirectional communication takes place on the data channels between a central and a peripheral device, where peripherals usually provide services (server role) and centrals access these services (client role). Services are structured into characteristics that a client can read from or write to. This structure is managed by the Generic Attribute Profile (GATT). Following the publish-subscribe pattern, clients can get notified of value changes (BLE terms: notifications/indications), e.g., push updated sensor values to clients. The value that can be read and transmitted from a characteristic at a time is commonly limited to 20 bytes.[1] Transmitting more than 20 bytes requires a split into multiple packages and possibly multiple read requests. A read-/write request can only be initiated by a central device. We now look at two common applications of BLE.

2.1.1 iBeacon

iBeacon is a protocol by Apple, developed on top of BLE to allow proximity and localization applications [2]. An iBeacon system is based on Broadcaster devices that are placed at known locations. These beacons purely broadcast a unique ID with a reference signal strength at 1m distance. Close-by smartphones can then listen to these broadcasting beacons and derive a coarse distance estimation based on their Rx power.

The originally intended use of iBeacons was to only provide coarse location context (far, near, immediate) to applications [1]. For example a store can install iBeacons that enhance an applications functionality when the user is close by. This is because RF technologies on the 2.4GHz band are error prone due to interferences through Wi-Fi activities, human bodies or metal objects as e.g., Jiang et al show in [24]. Despite that, there are several research projects that push iBeacons towards traditional indoor localization problems. In [10], iBea-

[1]This value is dependent on the protocols further up the stack. The actual data payload size of BLE is 27 bytes.

cons are used to implement an occupancy detection system in a non-residential building. The authors in [29] use iBeacons for localization and achieve an average estimation error of 0.53m. Recently Biehl et al presented a system that applies iBeacons to trusted indoor location estimation [3].

Performance wise, the main criteria for an iBeacon system are distance accuracy and a small latency of received broadcasting packages on the phone to quickly estimate proximity or location and provide it to an application. The preferred walking speed of humans is considered to be around 1.4 m/s [27]. Hence, applications need to update user location according to their requirements: $f = \frac{1.4\,\mathrm{m/s}}{a}$, where f is the update frequency, a is the accuracy requirement in m. E.g., if an application requires 1 m accuracy, it would need to update its location every 0.7 s. Further, iBeacon devices are commonly battery powered, which makes the choice of broadcasting frequency a trade-off between latency and battery lifetime.

2.1.2 Personal Smart Devices

Besides of iBeacon based localization, BLE is extensively used to enable smartphone centric, cyber-physical applications by wirelessly connecting sensors and actuators. The Bluetooth standard defines several GATT profiles for commonly used applications (e.g., alert notification, heart rate, weight scale, environmental sensing etc. [6]) that can be implemented on any device to allow interoperability between different manufacturers.

The main performance criteria for these type of applications are (i) a timely discovery process of devices and their services and (ii) a timely exchange of data packets to communicate state changes from peripheral to phone or vice versa. For example, a temperature sensor might communicates state changes to a phone or a phone might communicate a new state to an actuator (e.g., a light switch). Research has shown that humans are usually able to perceive switching delays greater than 100ms [34].

2.2 Android BLE Stack

The Android architecture divides OS functionalities into five layers from bottom to top: (i) Linux kernel, (ii) Hardware Abstraction Layer (HAL), (iii) Android runtime and libraries, (iv) Application framework and (v) Application layer (see Figure 1).

Android leverages the drivers from the Linux kernel for several hardware components like memory and power management, audio, video, Wi-Fi and also Bluetooth. However, an application developer is not able to access that native Bluetooth stack (`system/bt`) on non rooted, off-the-shelf devices directly. Further, device manufacturers are able to make extensions to the default stack. Application developers access BLE through

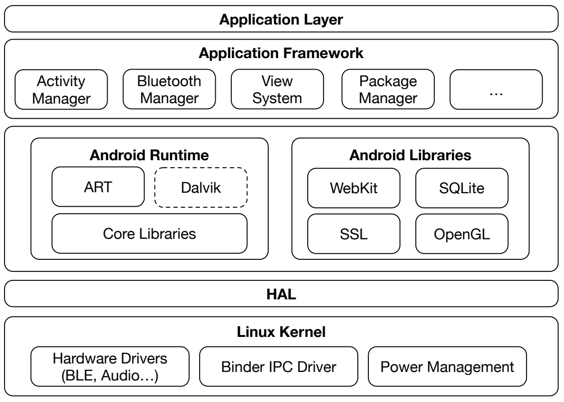


Figure 1: Simplified Android Architecture (based on [17, 37] and Android source code at [18]).

several abstraction levels by calling the Bluetooth application framework APIs (`android.bluetooth`). Internally this code then calls a single, running Bluetooth process (`packages/apps/Bluetooth`) through interprocess communication (Binder IPC). The Bluetooth process then coordinates the different requests by possibly multiple applications to the HAL layer using the Java Native Interface (JNI). Bluetooth events are communicated back using callbacks (e.g., an advertisement that matches a filter is discovered). The Android source code is available at `https://android.googlesource.com/`. Despite that, manufacturer specific extensions are closed source and Bluetooth hardware implementations are undocumented. As e.g., Moazzami et al show in [30], an application process may be blocked by the operating system anytime for up to 110ms. Besides that, Android OS versions in use are highly segregated. BLE was introduced into Android in 2013 (API level 18). Since then, it has evolved, making many API calls >= 21 behave differently and incompatible.

2.3 Hardware Implementation

BLE hardware (antenna, chip) and case design are implemented manufacturer and model specific. Their implementation details (e.g., antenna gain) are mostly not available on a smartphone's data sheet. However, many "tear-down" websites make some implementation internals, like the used chips, public. We looked at available information for the most popular models of the last three years according to [21] (see Table 1). The main insight is that all recent models implement BLE through a multipurpose SoC that handles 802.11, Bluetooth and possibly FM.[2] The main chip manufacturers are Qualcomm and Broadcom. Broadcom chips are becoming

[2]Front-end, power and low noise amplifier will still be model dependent in most cases.

Table 1: **BLE Chipset Hardware Implementations on Popular Smartphone Models between 2013–2015**

Phones	Chip	Type
iPhone 4S	Broadcom BCM4330	WiFi/BT
iPhone 5/5c/5s	Broadcom BCM4334/2	WiFi/BT
iPhone 6	Murata 339S0228	WiFi/BT
Samsung S4, Note 3, HTC One	Broadcom BCM4335	WiFi/BT
LG Nexus 5, LG G3	Broadcom BCM4339	WiFi/BT
Samsung S6, Nexus 6P	Broadcom BCM4358	WiFi/BT
Samsung Galaxy S5	Broadcom BCM4354	WiFi/BT
Motorola Moto G, LG Nexus 4	Qualcomm WCN3620	WiFi/BT
Motorola Moto X	Qualcomm WCN3680	WiFi/BT

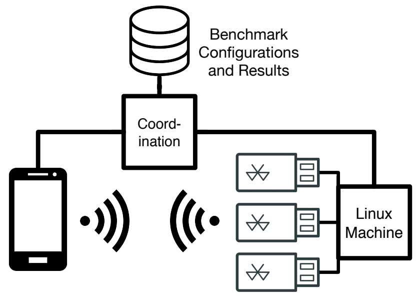


Figure 2: Overall Benchmarking Framework Architecture with Four Main Components: Smartphone, Repository, Coordinator, BLE Test Stand

more widespread recently.

3. FRAMEWORK AND BENCHMARK DESIGN

Our framework should enable a BLE performance evaluation on different, off-the-shelf smartphone models to understand the specific characteristics of single models. We first describe the design of our framework, before we discuss the rationale behind our experiments design.

3.1 Framework Design

In order to perform a wide scope of benchmarks on a variety of smartphones, we design a simple, portable and scalable distributed benchmarking framework:

- **Simple** It needs enable a simple creation of new types of benchmarks.
- **Flexible** It needs to enable performance tests for connection-oriented and connectionless operations.
- **Portable** It should be portable across Android versions, but also towards other platforms (iOS, Windows Mobile).
- **Reproducible** Performed experiments need to be reproducible by others and on other devices.
- **Scalable** It needs to scale to multiple peripherals and phones.

These requirements lead to our resulting system architecture in Figure 2. It consists of four main components: A repository (i) aggregates benchmarking results and stores their configurations. Configurations define the setup and different steps of a benchmark for phone and Peripheral(s) in JSON. They can be modified or created at will. A phone application (ii) accesses these configurations and coordinates the starting point of benchmarks through a coordinator (iii) with the commodity machine that connects multiple Peripherals via virtual serial ports (iv).

Our design is based on off-the-shelf hardware to allow for reproducibility of results and to create a common baseline for inter-smartphone comparisons. We implement our system using the widely available BLED112 USB dongle. It uses TI's CC2540 chip ([22]) and provides all Bluetooth 4.0 features through a virtual serial port to the host while providing a simple application interface, its retail price is roughly \$10 [26].

Because our framework is distributed across several physical components, it requires inter-component synchronization. A fine grained time synchronization protocol would interfere with running benchmarks (computational resources, RF interference and energy consumption). For this reason, we only synchronize the starting time of each benchmark and then rely on round-trip measurements on the smartphone and test stand. This has shown sufficient for all our metrics.

Encoding the global benchmarking instructions in JSON ensures portability, simplicity and makes the our approach scalable. The number of connected BLE microcontroller can be varied dependent on the requirements of a benchmark. The coordinator can run on the same machine to which the dongles are connected to. All benchmark results are committed to a Git repository together with the used configuration files. The repository is publicly available (see [15]).

As an example, Listing 1 shows a configuration snippet with instructions for the BLE microcontroller. They instruct the controller to advertise on all three advertisement channels for 30 s with defined short name and under a fixed advertisement interval while being scannable but not connectable for Central devices.

Listing 1: BLE Microcontroller Configuration Snippet

```
{
  "gap_role": "broadcaster",
  "gatt_role": "server",
  "replicas": 1,
  "steps": [
    {
      "adv_channels": "0x07",
      "short_name": "BLEva",
      "adv_interval_max": "0x200",
      "adv_interval_min": "0x200",
      "ble_operation": "advertising",
      "gap_connectable_mode": "gap_scannable_non_connectable",
      "gap_discoverable_mode": "gap_user_data",
      "sr_data": "0x0609424c457661",
      "time": 30000
    }
  ]
}
```

Listing 2 shows a snippet for a possible phone counterpart. The phone scans for 30 s, reporting back all matches for all advertisements without a filter or delay and using a power-balanced scanning mode.

Listing 2: Smartphone Configuration Snippet

```
{
  "gap_role": "observer",
  "gatt_role": "client",
  "steps": [
    {
      "ble_operation": "scanning",
      "callback_type": "all_matches",
      "match_mode": "aggressive",
      "match_num": "max_advertisement",
      "scan_mode": "balanced",
      "time": 30000,
      "report_delay": 0,
      "wifi_state": "off",
      "filters": [],
    }
  ]
}
```

Our framework maps these JSON instructions to native parameters and function calls of Android and the BLED112 programming interface. Each BLED 112 controller is orchestrated through a separate Python process. On Android we streamline the differences between the APIs of Android 4.4, 5 and 6 by implementing an abstraction library that checks the devices API, selects the correct API call and potentially emulates newer features for older devices (e.g., packet batching, filtering). This is especially important considering the high fragmentation of Android (currently, March 2016, Android 6.0 is only installed on 2.3% of Android phones).

3.2 Benchmark Design

3.2.1 Metrics

Multiple potentially relevant metrics exist: Latency, Package Reception Rate (PRR), throughput, Received Signal Strength Indication (RSSI) and power consumption. We define device discovery latency as the time it takes to receive the first advertising package of a Peripheral. This time interval is crucial for the responsiveness of many BLE applications (e.g., it determines the time until a phone can connect to a Peripheral). We define advertising latency as the time interval between the following consecutive packages. This interval determines how fast state changes can be transmitted through advertisements and is thus crucial for all iBeacon applications. In a connected state, we define read latency as the latency for a read operation while write latency expresses it for a write operation. Because a high throughput is not a design goal of BLE and is not a major performance criteria of BLE applications that we observed, we do not measure it directly in our experiments. Instead, we measure it through the latency of single read and write operations. RSSI is an important metric, because iBeacon is based on the RSSI-distance relation of received advertisements. The correct working of iBeacon depends in consequence on the validity of RSSI values.

Smartphones are battery powered, thus power consumptions needs to be considered. Power consumption can be measured in three distinct ways: (i) via an external power monitor (e.g., [12, 8]), (ii) via an internal monitor (e.g., [23]) and (iii) using software power models (e.g., [33]). To meet our design goal of a simple and repeatable benchmarking framework, we do not rely on external monitors or sophisticated software power models. We account for power consumption by extending the time dimension for an operation and measure the battery drop programmatically. The reason is that there is currently no possibility to read power consumption on a per app granularity on Android. Measuring the power consumption physically at the battery terminals does not scale well and can not simply be reproduced by others (e.g., current smartphones are often not designed with a replaceable battery). We have performed experiments with phones of the same model that show a close correlation ($r = 0.93$) between their reported battery levels. This means, that even if the power consumption is not in a linear relation with the reported battery level, it still allows to make a side-by-side comparison of different parameters for the same phone model.

3.2.2 System Parameters

We categorize the parameters that influence BLE performance on smartphones according to their GAP roles defined in the BLE standard (Central, Observer, Peripheral, Broadcaster). In each of these roles, a device can perform BLE operations in a connectionless or connection-orientated fashion. These operations are influenced by the chosen BLE parameters. Additionally, we find three external parameters that might influences any BLE operation: Processing (concurrency model), Environmental State and 2.4GHz State. Table 2 summarizes these parameters and their dimensions.

3.2.3 Experiments

The following experiments are based on our two motivational applications: "iBeacon" and "Personal Smart

Table 2: **BLE Performance Parameters Overview: 4 categories of parameters (BLE Parameters, Processing, Environment, 2.4GHz State) influence the BLE Operations in different GAP roles (Central, Observer, Peripheral, Broadcaster).**

	BLE Operations		**BLE Parameters**	**Processing**	**Environment**	**2.4GHz**
	Connectionless	**Connection-oriented**				
Central	Scanning, Device Discovery	Connecting (1st connect, re-connect), Detecting Services, Transferring, Receiving	Scan Window/Interval, Advertising Interval, Connection Interval, Slave Latency	App in Foreground/Background, Concurrency model, No. of processes active	Distance, Number of BLE devices, 2.4 GHz interference	Wi-Fi active, on, off, 2.4GHz, 5GHz
Observer	Scanning, Device Discovery					
Peripheral	Advertising, Scan Response	Indication, Notification				
Broadcaster	Advertising, Scan Response					

Devices". In these applications and thus in our benchmarks, the phone is in Central and Observer role and the microcontroller is in Peripheral and Broadcaster role. With this role allocation, the resulting main groups of experiments are:

- **Connectionless:** Scanning and advertising on different scan and advertising combinations.
- **Connectionless to Connected:** Device and Service Discovery on different parameter combinations.
- **Connection-oriented:** Data Transfer and Reception (read/write) with different connection interval parameters.

We describe detailed experimental configurations together with our results in the next section.

4. EXPERIMENTAL RESULTS

We performed an experimental evaluation on 9 different Android models from 2013 to 2015 and from different performance and price ranges (see Table 3). For the Nexus 7 and the Moto E2, we use two devices each, for the other models we use one device. All known models implement Wi-Fi and Bluetooth on the same chip. All our experiments use our benchmarking framework described in Section 3.

Most of the following experiments follow the experimental setup depicted in Figure 3. Four BLE dongles are distributed in a semicircle around the antenna area of the smartphone at a distance of 1.5 m. Dongles are active or inactive depending on the current benchmark.

4.1 Connectionless

In connectionless experiments, we evaluate the reception of advertisement packages. Advertisement packages are the base for Apple's iBeacon protocol and are the means in which a Peripheral makes itself visible to

Table 3: **Evaluation Devices: Smartphone models with their release date, API level and BLE chipset.**

Phone	Year	API	BLE Chip	Type
Motorola Moto G	2013	5.1.1	WCN3620	WiFi/BT combo
LG Nexus 4	2013	5.1	WCN3660	WiFi/BT combo
LG Nexus 5	2013	6.0	BCM4339	WiFi/BT combo
Asus Nexus 7	2013	6.0	WCN3620	WiFi/BT combo
LG G3	2014	5.0.2	BCM4339	WiFi/BT combo
Motorola Nexus 6	2014	6.0	BCM4356	WiFi/BT combo
LG Nexus 5X	2015	6.0	unknown	unknown
Motorola Moto E2	2015	5.1	unknown	unknown
Huawei Nexus 6P	2015	6.0	BCM4358	WiFi/BT combo

a Central device. As such they are the first step in the device discovery and connection process.

4.1.1 *Package Reception Rate*

An iBeacon-based BLE application requires the smartphone to continuously scan for advertisements and use the received packages as input to calculate proximity or a coarse location as an application context (e.g., a shopping mall recommondation system pushes product specific advertisements to close-by users). An ideal system should therefore provide a small initial latency until the first advertisement is received, and subsequently a uniform reception of advertisements to propagate state changes timely (e.g., the person or Peripheral moves).

In our experiment, we scan 30 s for advertisements. Scan results are reported back by a callback. We perform this experiment for three different scan modes (`low_power`, `balanced`, `low_latency`) on 9 different smartphone models and with a different number of peripherals (1,2,3,4) on different advertising intervals (20ms,

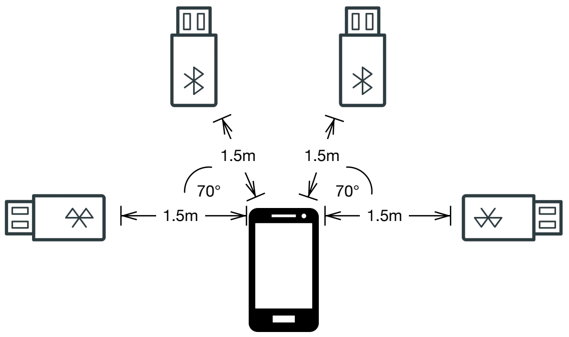


Figure 3: **Experimental Setup in most Experiments.**

160ms, 320ms, 640ms, 1280ms, 5120ms). These parameters result in 72 different benchmarks for each phone model. To mitigate a temporal influence of our results (e.g., due to environmental changes), we run each of the benchmarks in three repetitions at different times.

Figure 4 shows the resulting Package Reception Rate (PRR) of advertisements packages for the three scan modes and broken down to different smartphone models and advertisement intervals for all experiments with a single advertising Peripheral. It reveals several interesting insights:

(i) Most noticeable, PRR does not increase for all models with scan intensity. For some models PRR is not affected at all by the chosen scan mode (Asus Nexus 7, Motorola M, LG G3, LGE Nexus 4). Investigating our results in more detail, we discover that the default behavior of many models seems to be to solely report the first received advertisement of a Peripheral, but not succeeding ones. For other models, PRR increases as expected with scan intensity. The devices affected by this problem run on different Android OS versions up until the most recent Android 6. We suspect that these results might have to do with the specifics of the BLE chips (Qualcomm WCN3620 and WCN3660) and how the Bluetooth driver for these chips is implemented by the manufacturer.

(ii) No phone is capable of receiving all advertisements. The highest PRR we observer is 60%, even when scanning in `low_latency` mode which is implemented as constant scanning on Android (`scan_window=5000`, `scan_interval=5000`). The reason is that the Peripheral advertises on three different channels (2402, 2426, 2480 MHz) sequentially. A receiver listens to all these channels turn by turn, and will therefore not receive all advertisements for all channels.

(iii) PRR on the same parameters varies much between phone models (5% vs 60%). This means that e.g., scanning in `balanced` mode results in the same PRR for one model than in `low_latency` for another.

(iv) Too frequent advertising (20ms) results in a low PRR. A reason for that might be increased interference between packages. However, in BLE, a random delay is added to each advertising interval to avoid interference. Another reason might be that the receiver cannot scope with the amount of packages received. Because 20ms is the lowest supported value according to Bluetooth specification, we suspect that this is a borderline case and not sufficiently implemented by current BLE chips.

Wi-Fi Interference.

Traditional 802.11 is operating on the same 2.4GHz frequency range as Bluetooth and the two RF technologies are thus subject to general interference. Further, as we have shown, all current smartphones share the same chip and antenna for both Wi-Fi and Bluetooth (see Table 1). Wi-Fi and Bluetooth must thus be time-multiplexed on the chip.

To evaluate the implemented RF multiplexing, we perform scanning operations with Wi-Fi *on*, *off* and in an *active* state. In an active state, our framework emulates common Wi-Fi traffic patterns by performing various HTTP GET and POST requests concurrently with the scanning operation. In on state, Wi-Fi is left on, but not actively used. In off state, we turn Wi-Fi functionality programmatically off.

We observe that turning Wi-Fi off while scanning has no significant effect on scanning behavior. However, if Wi-Fi is actively used, it reduces PRR by a mean of 18.3 % for all smartphone models.

4.1.2 Advertising and Device Discovery Latency

Besides a high PRR for advertisements, two latency metrics are important: (i) A small device discovery latency (time until an initial package is received) is crucial for any application that requires a timely interaction between a Central and Peripheral device (e.g., a smart appliance). (ii) The latency of subsequently received advertisements is important for any application that builds on state transmission through advertisements (e.g., any iBeacon-like location tracking, proximity application).

To measure the initial latency when discovering a device, we start to scan and measure the time until we receive the first package. We use a filter on the device name to only receive results from the device we are looking for. For advertising latency, we measure the interval between subsequently received packages. Figure 5 shows the device discovery latency for different scan and advertisement intervals.

As discussed previously, the 20ms parameter, although specified, seems not to be fully supported by current BLE hardware, leading to erratic results across models. At the upper end, a 5120ms interval results in for many application cases unacceptable latencies of up to 26s. In `low_power` or `balanced`, some phones do not

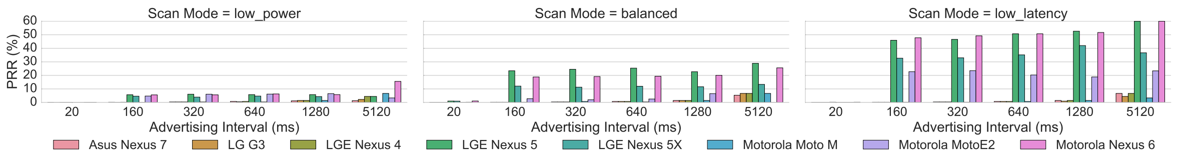
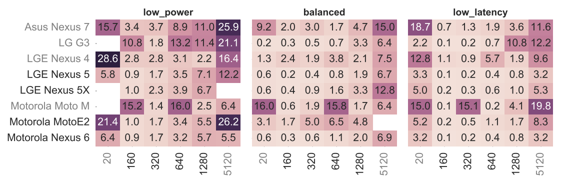


Figure 4: Package Reception Rate of Advertising Packages for different Smartphone Models, Advertising Intervals and Scan Modes.



	low_power						balanced						low_latency					
	20	160	320	640	1280	5120	20	160	320	640	1280	5120	20	160	320	640	1280	5120
Asus Nexus 7	15.7	3.4	3.7	8.9	11.0	25.9	9.2	2.0	3.0	1.7	4.7	15.0	18.7	0.7	1.3	1.9	3.6	11.6
LG G3		10.8	1.8	13.2	11.4	21.1	0.2	0.3	0.5	0.7	3.3	6.4	2.2	0.1	0.2	0.7	10.8	12.2
LGE Nexus 4	28.6	2.8	2.8	3.1	2.2	16.4	1.3	2.4	1.9	3.8	2.1	7.5	12.8	1.1	0.9	5.7	1.9	9.6
LGE Nexus 5	5.8	0.9	1.7	3.5	7.1	12.2	0.6	0.2	0.4	0.8	1.9	6.7	3.3	0.1	0.2	0.4	0.8	3.2
LGE Nexus 5X		1.0	2.3	3.9	6.7		0.6	0.4	0.9	1.6	3.3	12.8	5.0	0.2	0.3	0.6	1.0	5.3
Motorola Moto M		15.2	1.4	16.0	2.5	6.4	16.0	0.6	1.9	15.8	1.7	6.4	15.0	0.1	15.1	0.2	4.1	19.8
Motorola MotoE2	21.4	1.0	1.7	3.4	5.5	26.2	3.1	1.7	5.0	6.5	4.8		5.2	0.2	0.5	1.1	1.7	8.3
Motorola Nexus 6	6.4	0.9	1.7	3.2	5.7	5.5	0.6	0.3	0.6	1.1	2.0	6.9	3.2	0.1	0.2	0.4	0.8	3.2

Figure 5: Device Discovery Latency: Each map depicts a different scan mode (low_power, balanced, low_latency). The rows represent different phone models, the columns different advertisement intervals. The cell numbers are the latency in seconds.

receive any package in a 30s period. This is because Android implements scan modes in terms of different values for scan window and interval. In `low_power` mode for instance, the interval is set to 5000ms and the scan window to 500ms. This means we effectively only scan for 3000ms during a 30s benchmark and if scan windows are not coincidentally aligned with the peripherals advertisements, we miss them.

For mid-range advertising intervals, we can distinguish between two groups of models: (i) a group where latency increases with a growing advertising interval and with a less frequent scan mode (black names) and (ii) a group where results are much less consistent (grey names).

In accordance with our PRR results, it is sufficient for some models to scan in a low frequency mode in order to receive the same results as another model scanning in high frequent mode (e.g., Nexus 6 can scan in `balanced` mode, while the Asus Nexus 7 should scan in `low_latency` mode). To increase the chance of a timely reception on all models, we advice to use a maximum advertising interval of 640ms. User centric applications that require high responsiveness, might even require an interval of 320ms or smaller.

4.1.3 *Multiple Peripherals*

We performed all previously discussed connectionless experiments for 1–4 peripherals to evaluate the reception loss when a scanner needs to detect multiple, simultaneous advertisers. Our results show that PRR worsens slightly with multiple advertisers. While the mean PRR (across all smartphone models, all scan modes and all advertising intervals) is 9.1% with a single advertiser, it drops to 8.3% for two, 7.4% for three and only 5.6% for four. The mean latency until all Peripherals have been discovered worsens by 56% from a single to four Peripherals.

4.1.4 *RSSI and PRR at Different Ranges*

The maximum specified range of BLE is 100 m [4], but in practice BLE is used on a much smaller scale. We design our experiments for up to 30 m, where we perform experiments at various distances. Our testbed setting is a long corridor of 40 m in which we place both receiver and transmitter at a height of 1.5 m. We use a corridor because of its property to channel signals so that signal intensity drops at a regular rate when moving further from the transmitter (see e.g., [19]) and to minimize shadowing effects.

We expect RSSI to correlate with the log-distance radio propagation model which models the path loss of a signal inside a building and is defined as:

$$\mathrm{PL}_{d_0 \to d} = \mathrm{PL}_{d_0} + 10\alpha \log_{10} \frac{d}{d_0} + \chi_\sigma \qquad (1)$$

Where, $\mathrm{PL}_{d_0 \to d}$ is the path loss at distance d, PL_{d_0} is the known path loss at distance d_0, α is the path loss exponent, which is environment dependent (e.g., 1.8 for non obstructed indoor environment), χ_σ is a zero-mean Gaussian distributed random variable (in dB) with standard deviation σ that models the shadowing affect on the signal.

In our experiments, we measure RSSI and PRR. We set the transmitter power to 3dBm. Figure 6 shows the mean RSSI values for different phone models at different distances (1-16 m) for a scanning period of 60 seconds at each location. From these results, we note two issues: (i) As it has been stated in previous work (e.g., [32]), RSSI is in fact not an ideal parameter for distance estimation. The results for one model and location are not stable and vary within 10dBm. (ii) The median values of different models differ further by a factor of >20dBm. These varied results are due to the hardware (radio chip, antenna, case design) and software (driver implementation) differences between different phone models.

Addressing issue (i), requires applying a filtering tech-

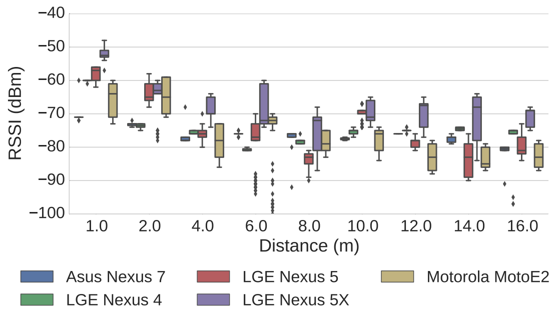


Figure 6: RSSI for different smartphone models at different distances.

nique to the raw values (e.g., median, maximum, Gaussian or Kalman filter), but (ii) requires a smartphone model specific calibration factor. Current iBeacon based proximity and localization applications do not account for this. We discuss our improvements in Section 5.

Looking at the PRR-distance relationship, Figure 7 shows that PRR does not drop when extending the distance. We trace this back to our chosen testbed environment that enforces the propagation of radio signals and the relatively high Tx power of 3 dBm.

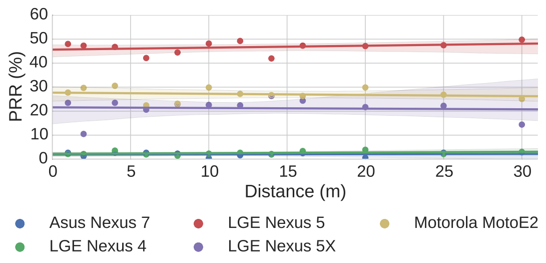


Figure 7: PRR vs. Distance for different models.

4.1.5 *Summary*

In summary, our evaluation of connectionless operations shows great differences for PRR (3% vs. 60%) and Latency (4 s vs. 30 s) between phone models. Some models behave erratically and only report the first advertisement for a Peripheral, while others work as expected. BLE has worked well up to 30 m. The RSSI values are aligned with the Log-Distance Model, but are spread widely between different models (> 20 dBm) and show a high fluctuation at the same location.

4.2 Connection-oriented

After a smartphone has discovered a Peripheral, it will in many scenarios connect to it, discover available services and perform read or write requests. The design goal of BLE is a timely exchange of a small amount of data (e.g., read a new value from a smart scale). We therefore focus on latency aspects of connection, service discovery and read/write requests. Two performance determining connection parameters exist: (i) connection interval and (ii) slave latency. The connection interval determines how often a Central device will ask for data from the Peripheral. The Peripheral can request a connection interval between 7.5ms and 4s, but the Central device decides ultimately on the interval. As such, medium access is coordinated by a Time Division Multiple Access (TDMA) schema. Slave latency is set as multiplication of the connection interval on which the Peripheral can decide not to respond to the Central device. This allows a Peripheral to stay in a low power sleep mode for longer and only wake up if it has e.g., new data from a sensor.

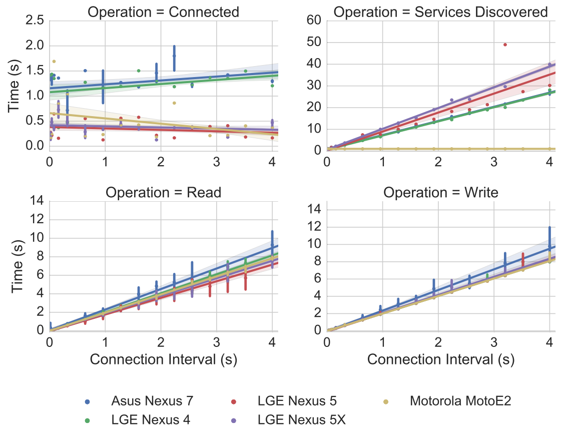


Figure 8: Latency of common GATT operations in Relation to Connection Interval

The Android API does not support to set the connection interval explicitly, but it supports intervals from 7.5ms to 4s. In the following, we therefore let the Peripheral (slave) request different connection intervals to set the parameter implicitly.

4.2.1 *GATT Operations*

We observe a Central device that initiates a connection to a Peripheral. This means that the Peripheral has already been discovered (see Section 4.1.2 for benchmarks of the discovery operation). In our experiment, we sequentially connect, discover the services and read and write to a characteristic. We perform this sequence for different connection intervals (7.5–4000ms) and in 100 repetitions on each phone. Figure 8 shows our results for these GATT operations. The Peripheral is advertising in 320ms intervals.

As expected, the connection establishment is bound to the advertising interval and not to the connection interval. We measure a mean of 721ms. This is due to the non-time-synchronized nature of the advertising channels. All other operations correlate with the connection interval. The results for the Motorola MotoE2 are an exception. Our first explanation was that the device performs some caching of discovered services that is persistent even when the Bluetooth Adapter has been reset and the application restarted. However, we noticed this behavior as well for different peripherals and a full Android restore. When comparing the timestamps on the Peripheral and on the Moto E, we discovered that the Moto E performs the service discovery always before the connection interval can be changed.

In general, the experiments show that it is important to consider the impact of the connection interval on the service discovery process. A mean latency of 4s for a read request might still be acceptable for some applications, while the corresponding discovery latency of 30s might not be. The Android BLE stack should therefore on a OS level rely on caching services of known devices to speed up subsequent read/write requests.

4.2.2 Summary

Two parameters have a crucial impact on BLE connection latency: initial connection between smartphone and peripheral and service discovery. In order to improve latency, it is thus advantageous to keep a connection to a Peripheral open or to cache service information of a Peripheral across connections. Our experiments show that such measures can halve the time it takes to read or write.

4.3 Power Consumption

4.3.1 Connectionless Power Consumption

We measure scanning power consumption for five different phone models and for the three scan modes of Android. In these experiments, we first fully charge each phone and then perform a scan operation for the period of 5 hours. We also measure the idle battery drop of the battery which in our experiments has been in the range of 1–3%. Figure 9 depicts the battery drop for different scan modes. The mean battery drops are 8.5% (`low_power`), 17% (`balanced`) and 19.8% (`low_latency`). The small difference between `balanced` and `low_latency` scanning suggest that `low_latency` can be used to improve PRR and latency without sacrificing battery lifetime much.

4.3.2 Connection Power Consumption

We have experimented with different connection intervals and kept a connection open for a period of several hours, but we could not notice any differences between small and big intervals. We conclude that the impact on power of an open connection might in many cases be negligible when considering the overall battery budget of a phone. In future work, we want to investigate this aspect of power consumption by measuring power directly at the battery terminals.

5. BLE LIBRARY EXTENSION

In the previous section, we showed big discrepancies in the BLE stack behavior on different smartphone models through an extended experimental evaluation of 9 different models. In this section, we discuss the initial design of an library that sits on top of the native Android BLE stack and show some results of its implementation.

5.1 Design Goals

We have the following design goals:

- **Uniform BLE behavior.** Ideally, every smartphone should exhibit the same BLE performance characteristics. With our library, we want to achieve some soft performance boundaries that Peripheral designers can count on when building their systems.
- **Off-the-shelf devices.** We want out library to be compatible with any unmodified (e.g., un-rooted) mobile devices. As such, our implementation needs to be designed on top of existing abstractions of the Android OS.
- **Different Android OS versions.** As the Android ecosystem is fragmented, our goal is to support different OS versions, independent from their API. We want our implementation to be backwards compatible to Android 5.0 (38.4% of install base, March 2016).
- **Dynamic and extendable.** We aim for a system that is dynamic and extendable towards future API changes, new phone models and novel uses of BLE that require different performance characteristics. We want our approach also extendable to different mobile operating systems (e.g., iOS, Windows Mobile).

5.2 Design and Implementation

To meet these goals, we design our system around the central repository of benchmark results (see Section 3). The aggregated results for each smartphone model serve as input for an Android library. This library can be added to any Android project. It follows the BLE APIs of Android 5.0, but maps the API calls to the ones that correspond to the smartphone's OS version. Besides this API translation, the library introduces a smartphone model specific behavior in terms of (i) function parameters, (ii) processing and (iii) output.

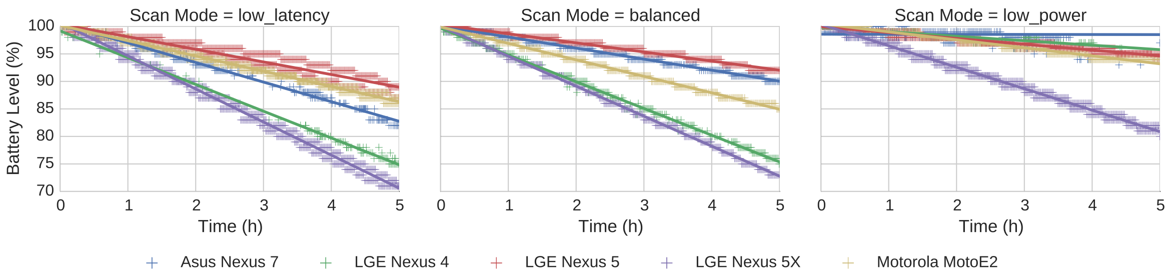


Figure 9: Scanning Power Consumption: Battery drop for different phone models and scanning modes.

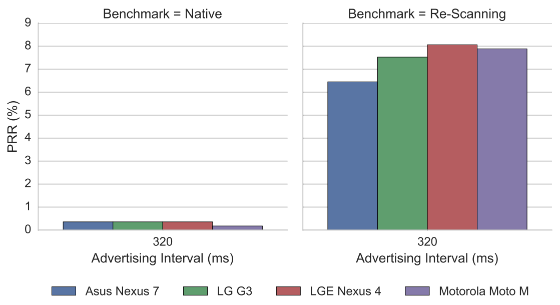


Figure 10: Boosting performance by re-scanning after a received advertisement.

5.3 Initial Results

We now show some initial results of our implementation in the context of scanning performance and iBeacon accuracy improvements.

5.3.1 Improving Scanning Performance

As Section 4.1 has shown, some devices show a scanning performance that is insufficient for many application scenarios. To improve PRR on affected models, we implement a simple re-scan strategy in where we restart scanning immediately after we successfully receive a package. This improves PRR by more than 20-fold (see Figure 10).

Comparing the energy consumption with the native solution during a scanning operation of 5h has shown no measurable difference between the two approaches. This is because also in the native implementation, the smartphone keeps scanning, but does not report new advertisements back for known Peripherals.

5.3.2 Improving iBeacon Accuracy

We have shown a hardware caused variety in phones' RSSI values at the same location. These differences make a generic distance/proximity estimation impossible (e.g., RSSI varies by 20dBm at 1m distance).

We approach this problem by creating a regression-model for each phone model. There are two problems when doing so: (i) the model is specific to our testbed equipment and (ii) the model is skewed by the physical environment (signal reflections, shadowing) To remove the influence by our testbed, we build our model not based on the absolute RSSI value, but on the ratio of RSSI and reference RSSI at 1m:

$$\text{Ratio} = \frac{\text{RSSI}_d}{\text{RSSI}_{d_{1m}}} \tag{2}$$

As such, our model will give correct results as long as the assumption that the reference RSSI at 1m is correctly determined by the manufacturer.

To mitigate that the model is skewed by the physical environment, we assign increasing uncertainties to our measurements as distance increases. These uncertainties are then used as weights in the least-square regression. We further remove data outliers by disregarding all values outside 1.5 IQR below the first quartile and above the third quartile. Finally, we base regression on the mean.

Figure 11 shows the result for the different smartphone models. An initial evaluation of our model has shown improved results over a generic distance propagation model. It reduces the overall mean error for distances of 1–16m from 3.35 to 1.36m when we apply it in a second indoor test environment.

6. RELATED WORK

In [38, 37], the authors emphasize the non-deterministic behavior of Android. Their RTDroid system replaces the default Android Java VM with a real-time VM, which provides predictability for Android applications. Our approach is different. We do not aim to provide a fully deterministic system, but dynamically change the working of the Android BLE stack to provide some predictability across models and OS versions for BLE operations. Our system is built as an abstraction on top of existing Android libraries opposed to redesigning and replacing Android OS components. This makes it applicable for all off-the-shelf smartphones. In spite of

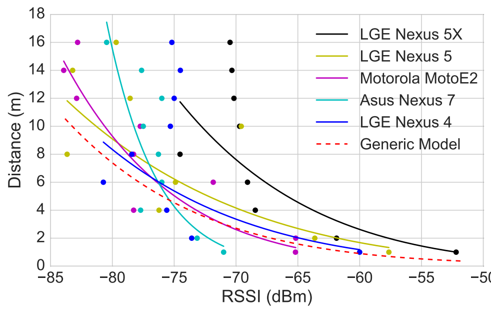


Figure 11: Optimized Distance Model

that, our approach is complementary and could run on top of a system like RTDroid.

[30] describes a smartphone-based platform for data-intense, embedded sensing applications. It consists of three tiers (smartphone, peripheral board and cloud service) for which it minimizes energy consumption while adhering to upper bounded processing delays. This is achieved by a task partitioning framework that assigns tasks to tiers based on their time-criticality, compute intensity and latency/power consumption profiles.

The authors in [20] characterize the performance of smartphone network applications by performing measurements on a variety of different phones an mobile networks. Similar to our work, their benchmarks are applications based and performed on real phones. Their findings show that performance varies widely due to hardware and software differences. Our work gives similar resukts for smartphones' BLE stack.

In the sensor network community, there have been efforts to develop testbed platforms to achieve a baseline for comparing different sensor network protocols. The Mirage testbed by Intel Research at Berkeley provided a wireless sensor testbed in which users bid for testbed resources in an auctioning scheme [9]. Likewise [11] provides a remotely accessible mobile wireless and sensor testbed. Recently, in [7] the authors develop JamLab, a low cost infrastructure for generating RF interference in sensor networks. [31] provides a large testbed of mobile phones which can be used to deploy experiments. With our system we provide a low-cost testbed infrastructure for BLE performance on off-the-shelf smartphones that enables a common baseline for arbitrary BLE benchmarks in similar fashion. Because smartphones and our testbed components are available off-the-shelf, we do not see the need to provide a centralized testbed platform like [9, 11]. Opposed to that, we provide a simple framework that builds on these components and a central repository for benchmarks and their results.

There has been various work that discusses performance aspects of the Bluetooth Low Energy protocol. In [25, 35, 16] the authors present an early description and performance evaluation of BLE. They compare BLE with other protocols like ZigBee/802.15.4, 6LoWPAN, Z-Wave and classic Bluetooth. In [36] the authors evaluate interference between BLE, 802.15.4 and 802.11, while [28] analyse the device discovery of BLE. All this work aims at evaluating BLE as protocol and as its implementation on various microcontroller. Our work looks at the main application of BLE in the wild, which is in smartphone-peripheral systems. We first evaluate the current state and then implement our design to achieve a general better performance while achieving a greater uniformity across different smarphone models.

7. CONCLUSION

With Bluetooth Low Energy (BLE), smartphones can act as situated gateways between smart infrastructure and cloud-based data processing. Understanding, and if needed improving, BLE performance on smartphones is crucial to meet the requirements of such gateways. This paper introduced a framework for benchmarking BLE performance on smartphones. Our framework builds on off-the-shelf components. It is available online as a resource for the community (see [14]). We presented a detailed experimental evaluation of nine different Android smartphones. Our evaluation shows that native Android BLE stacks fail to provide a homogeneous abstraction for different BLE implementations. We addressed this shortcoming and proposed an Android library that adapts to the idiosynchracies of a specific BLE implementation by relying on aggregated, model-specific data, obtained with our benchmark. Our library is designed as a wrapper around the native Android BLE stack. We showed that our library provides significant improvement for scanning performance and distance estimation. We argue that the principles we introduced for the design of our library should become an integral part of Android.

8. REFERENCES

[1] Apple. Getting started with ibeacon. https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf, 06 2014.

[2] Apple. iBeacon for Developers. https://developer.apple.com/ibeacon/, 2016.

[3] J. T. Biehl, A. J. Lee, G. Filby, and M. Cooper. You're where? prove it!: towards trusted indoor location estimation of mobile devices. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 909–919. ACM, 2015.

[4] Bluetooth SIG. Adopted specifications. https://www.bluetooth.com/specifications/adopted-specifications.

[5] Bluetooth SIG. Our history. https://www.bluetooth.com/media/our-history, 01 2016.

[6] Bluetooth SIG. Services. https://developer.bluetooth.org/gatt/services/Pages/ServicesHome.aspx, 01 2016.

[7] C. A. Boano, T. Voigt, C. Noda, K. Römer, and M. Zúñiga. Jamlab: Augmenting sensornet testbeds with realistic and controlled interference generation. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 175–186. IEEE, 2011.

[8] N. Brouwers, M. Zuniga, and K. Langendoen. Neat: a novel energy analysis toolkit for free-roaming smartphones. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, pages 16–30. ACM, 2014.

[9] B. N. Chun, P. Buonadonna, A. AuYoung, C. Ng, D. C. Parkes, J. Shneidman, A. C. Snoeren, and A. Vahdat. Mirage: A microeconomic resource allocation system for sensornet testbeds. Institute of Electrical and Electronics Engineers, 2005.

[10] G. Conte, M. De Marchi, A. A. Nacci, V. Rana, and D. Sciuto. Bluesentinel: a first approach using ibeacon for an energy efficient occupancy detection system. In *BuildSys@ SenSys*, pages 11–19, 2014.

[11] R. Fish, M. Flickinger, and J. Lepreau. Mobile emulab: A robotic wireless and sensor network testbed. In *IEEE INFOCOM*, 2006.

[12] R. Fonseca, P. Dutta, P. Levis, and I. Stoica. Quanto: Tracking energy in networked embedded systems. In *OSDI*, volume 8, pages 323–338, 2008.

[13] J. Fürst, K. Chen, M. Aljarrah, and P. Bonnet. Leveraging physical locality to integrate smart appliances in non-residential buildings with ultrasound and bluetooth low energy. In *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 199–210. IEEE, 2016.

[14] J. Fürst, K. Chen, and P. Bonnet. Bleva repository. http://github.com/EnergyFutures/bleva, 2016.

[15] J. Fürst, K. Chen, and P. Bonnet. Bleva results repository. http://github.com/EnergyFutures/bleva-results, 2016.

[16] C. Gomez, J. Oller, and J. Paradells. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *Sensors*, 12(9):11734–11753, 2012.

[17] Google. Android interfaces and architecture. https://source.android.com/devices/.

[18] Google. Android source repository. https://android.googlesource.com, 2016.

[19] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki. Practical robust localization over large-scale 802.11 wireless networks. In *Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 70–84. ACM, 2004.

[20] J. Huang, Q. Xu, B. Tiwana, Z. M. Mao, M. Zhang, and P. Bahl. Anatomizing application performance differences on smartphones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 165–178. ACM, 2010.

[21] Insidermonkey. 10 best selling smartphones in the world 201x. http://www.insidermonkey.com/, 2015.

[22] T. Instruments. Cc2540. http://www.ti.com/product/CC2540.

[23] Intel. Intel power gadget. https://software.intel.com/en-us/articles/intel-power-gadget-20, 2014.

[24] X. Jiang, C.-J. M. Liang, K. Chen, B. Zhang, J. Hsu, J. Liu, B. Cao, and F. Zhao. Design and evaluation of a wireless magnetic-based proximity detection platform for indoor applications. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, pages 221–232. ACM, 2012.

[25] S. Kamath and J. Lindh. Measuring bluetooth low energy power consumption. *Texas instruments application note AN092, Dallas*, 2010.

[26] B. S. Labs). Bled112 bluetooth smart dongle. https://www.bluegiga.com/en-US/products/bled112-bluetooth-smart-dongle/, 2015.

[27] R. V. Levine and A. Norenzayan. The pace of life in 31 countries. *Journal of cross-cultural psychology*, 30(2):178–205, 1999.

[28] J. Liu, C. Chen, Y. Ma, and Y. Xu. Energy analysis of device discovery for bluetooth low energy. In *Vehicular Technology Conference (VTC Fall), 2013 IEEE 78th*, pages 1–5. IEEE, 2013.

[29] P. Martin, B.-J. Ho, N. Grupen, S. Munoz, and M. Srivastava. An ibeacon primer for indoor localization: demo abstract. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, pages 190–191. ACM, 2014.

[30] M.-M. Moazzami, D. E. Phillips, R. Tan, and G. Xing. Orbit: a smartphone-based platform for data-intensive embedded sensing applications. In *Proceedings of the 14th International Conference on Information Processing in Sensor Networks*, pages 83–94. ACM, 2015.

[31] A. Nandugudi, A. Maiti, T. Ki, F. Bulut, M. Demirbas, T. Kosar, C. Qiao, S. Y. Ko, and G. Challen. Phonelab: A large programmable smartphone testbed. In *Proceedings of First International Workshop on Sensing and Big Data Mining*, pages 1–6. ACM, 2013.

[32] A. T. Parameswaran, M. I. Husain, S. Upadhyaya, et al. Is rssi a reliable parameter in sensor localization algorithms: An experimental study. In *Field Failure Data Analysis Workshop (F2DA09)*, page 5, 2009.

[33] A. Pathak, Y. C. Hu, and M. Zhang. Where is the energy spent inside my app?: fine grained energy accounting on smartphones with eprof. In *Proceedings of the 7th ACM european conference on Computer Systems*, pages 29–42. ACM, 2012.

[34] B. Shneiderman. *Designing the user interface-strategies for effective human-computer interaction*. Pearson Education India, 1986.

[35] M. Siekkinen, M. Hiienkari, J. K. Nurminen, and J. Nieminen. How low energy is bluetooth low energy? comparative measurements with zigbee/802.15. 4. In *Wireless Communications and Networking Conference Workshops (WCNCW), 2012 IEEE*, pages 232–237. IEEE, 2012.

[36] S. Silva, S. Soares, T. Fernandes, A. Valente, and A. Moreira. Coexistence and interference tests on a bluetooth low energy front-end. In *Science and Information Conference (SAI), 2014*, pages 1014–1018. IEEE, 2014.

[37] Y. Yan, S. Cosgrove, V. Anand, A. Kulkarni, S. H. Konduri, S. Y. Ko, and L. Ziarek. Real-time android with rtdroid. In *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*, pages 273–286. ACM, 2014.

[38] Y. Yan, S. Cosgrove, E. Blantont, S. Y. Ko, and L. Ziarek. Real-time sensing on android. In *Proceedings of the 12th International Workshop on Java Technologies for Real-time and Embedded Systems*, page 67. ACM, 2014.

A Practical Model for Human-Smart Appliances Interaction

Jonathan Fürst, Andreas Fruergaard, Marco Høvinghof Johannesen, Philippe Bonnet
IT University of Copenhagen
{jonf, afru, mhoj, phbo}@itu.dk

ABSTRACT

Built environments are increasingly equipped with smart appliances that allow a fine grained adaption to the personal comfort requirements of single individuals. Such comfort adaption should be based on a human-feedback loop and not on a centralized comfort model like Predicted Mean Vote (PMV). We argue that such a feedback-loop should be achieved through local interaction with smart appliances. Two issues stand out: (i) How to impose logical locality as a restriction when interacting with a smart appliance? (ii) How to mediate conflicts between several persons in a room, or conflicts between building-wide policies and local user preferences? In this work, we approach both problems by defining a general model for human-smart appliance interaction. We present a prototype implementation with an off-the-shelf smart lighting and heating system in a shared office space. Our approach minimizes the need for location metadata. It relies on a human-feedback loop (both sensor based and manual) to identify the optimal setpoints for lights and heating. These setpoints are determined by considering individual comfort preferences, current user location and a global goal of minimizing energy consumption.

1. INTRODUCTION

Built environments are increasingly equipped with smart appliances that allow a fine grained adaption to the personal comfort requirements of single individuals. Buildings are a prime deployment area for such appliances: they consume $\sim 40\%$ of our energy while the average person spends $> 90\%$ indoors [2, 6]. Two main categories of smart appliances in buildings are lighting and HVAC.

In larger, non-residential buildings, Building Management Systems (BMS) aim to improve energy consumption and comfort by applying a central comfort and energy model (e.g., PMV model). Even past work has made BMS accessible to user input (e.g., Thermovote [4]), smart appliances enable unarguably for much more direct and fine grained forms of user interaction. However, the current interface abstractions of smart appliances are not well designed:

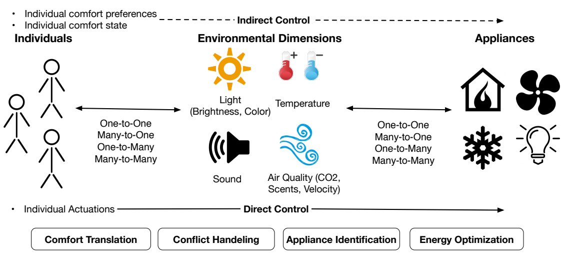


Figure 1: A Model for User-Smart Appliance Interaction

If we look at analog lighting and HVAC control, the link between a user's preference and the appliance is a physical object, e.g., a wall switch or thermostat. In BMS and in many off-the-shelf smart appliances, this physical link has been replaced by a textual namespace and by applying metadata to different appliances and their location. E.g., a smart light might be tagged with a building, floor no., room no. and a inter-room location: `ITU/4/4D21/Ceiling1`.

If such naming is consistent, it allows for building wide control (e.g., turn off all lights on the 4th floor). However, relying on such naming to create a human interaction interface, causes problems. As a scenario, imagine a non-residential building is equipped with smart lighting. The intention is to allow occupants to adapt lights to their current needs (e.g., by changing the brightness to their mood and work task). A model that relies on naming to specify location information suffers from the problem that users need to know where they are located (e.g., I am in meeting room `5A24`), but also where appliances are located inside a room (e.g., which ceiling light is above me). These dynamic relations are difficult to represent. Further, most appliances influence more than a single user. Individual preferences are different. This makes it necessary to resolve potential conflicts. Such conflict resolution needs be based on the current comfort preferences and inter-room location of the individuals. Our insights for improved smart appliance interaction are thus:

- **Logical Locality.** Appliance identification should be based on the impact (logical relation) on the single user.
- **Conflict Mediation.** Conflicts should be mediated between individuals locally (taking their preferences

into account), or between local individual preferences and global infrastructure-use policies.

In this work, we approach these problems by (i) defining a general model for human-smart appliance interaction and (ii) by implementing a prototype with an off-the-shelf smart lighting and heating system in a shared office space. Our approach minimizes the need for location metadata. It relies on personal sensors (e.g., smartphones, wearables) and human-in-the-loop feedback to identify the optimal setpoints for lighting and heating. These setpoints are determined by taking comfort preferences, current physical and logical location and a minimal energy consumption into account. Our prototype achieved 94% energy efficiency for lighting using a probabilistic method of identification, while adhering to the occupants' comfort ranges. For heating, we achieved an improvement of 80% in comfort while keeping a 3.3 °C lower overall heating setpoint.

2. RELATED WORK

Several approaches facilitate an identification of appliances: (i) Physical appliance tags provide a local context to the user (e.g., using BLE [9], RFID [8], QR-codes [7] or infrared [10]). Besides of precision limitations (especially RF based methods) and high infrastructure overhead, these techniques define human-appliance relations based on physical distance, but not based on a logical relation (e.g., a smart light that is physically distant from the user, but has a strong radiation effect on her). (ii) Metadata based approaches allow users to identify appliances based on textual descriptors. This is not human-friendly. E.g., if the metadata scope is building wide, selecting the right appliance is cumbersome. Indoor localization reduces this scope, but does not model the logical human-appliance relation.

In contrast to these methods, our model captures the human-appliance relation logically in terms of the appliance impact on human comfort and considers global goals for energy efficiency. SurroundSense provided a first implementation of such a logical localization on the granularity of different stores using environmental smartphone measurements [3]. We push these ideas by (i) adding a human feedback loop, (ii) knowledge of the current appliance state in a room (e.g., which lights are on/off) and (iii) by introducing small modulations of the environment to speed up the reduction of the search space. This enables a logical localization per appliance opposed to per room.

3. INTERACTION MODEL

Our main design goal is to enable an adaption of the smart environment to the comfort needs of the current occupants. On an abstract level, this means that different environmental dimensions (Light, Temperature, Sound, Air Quality) must be matched to the individual comfort preferences (see Figure 1). This matching is achieved by altering the output of available appliances accordingly (e.g., the user's light state is too dark → increase brightness of affecting smart light). Hence, it requires that the logical relations between humans and appliances are identified beforehand.

We argue that in most scenarios, appliance authorization can be based on physical locality (if I am physically inside a room, I should have access to its appliances) as we proposed in [5]. The system must allow for both, an automated

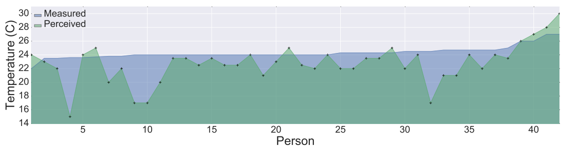


Figure 2: Measured and Perceived Temperature

adaption and for a manual, user initiated change of the environment. The bottom of Figure 1 shows the four main building blocks of our model. We will now discuss each of them.

3.1 Comfort Translation

In order to adapt the environment to the individual, it is necessary to translate comfort preferences between human and machine. We performed a survey among 50 students and employees at our campus to understand their sensitivity to temperature. Figure 2 shows that most people are not able to estimate the current temperature accurately. The mean error is 2.1 °C, while single estimates are off by 8.6 °C. An initial survey among colleagues showed that the error in human perception is even higher for less common units like Lux. In our system, we thus use relative user intents (e.g., "I am too cold/hot") to construct individual comfort ranges. This approach is been successfully followed in industry (see e.g., [1]).

3.2 Appliance Identification

Appliance identification builds on two levels: (i) appliances need to be mapped to their respective environmental dimensions (macro level) and (ii) appliances need to be mapped to individual occupants by their logical relation (micro level). Both mappings are only in the most simple cases one-to-one. Some appliances influence more than one dimension (e.g., heating influences both temperature and air quality; window shades influence both, light and temperature). Multiple occupants are in many cases influenced by a single appliance (e.g., a ceiling light or the room heating).

In our model, we base the dimensional mapping of appliances on metadata (e.g., a light has its primary output in the light dimension and its secondary in the temperature dimension). The logical mapping of appliances to individuals is dynamic and not solely expressible with metadata. It depends on the present occupants and their location. This mapping must thus be created by a feedback loop. The feedback loop combines both human-input ("I am too hot.") and sensor readings (personal temperature, brightness sensor).

We abstract these relations as a weighted property graph $G = (V, E)$, with $|E| = m$ edges and $|V| = n$ vertices. V and E have a set of key value pairs that represent properties P (e.g., vertex type, primary/secondary output, current state, edge cost…). This graph is constructed by using the feedback-loop as input for a graph exploration algorithm. The algorithm incrementally creates a property graph of appliances, their output (e.g., light) and affected occupants. Figure 3 depicts an simplified example of a constructed graph for our shared office with two occupants. It is important to note that this graph is not static, but in constant change. It is usually only partly and not fully explored. We discuss its construction process by means of our

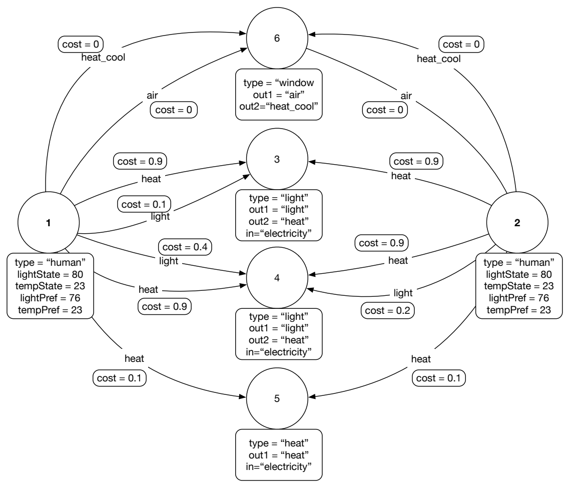


Figure 3: Property Graph for two occupants, two lights, an automatic window and a heating system.

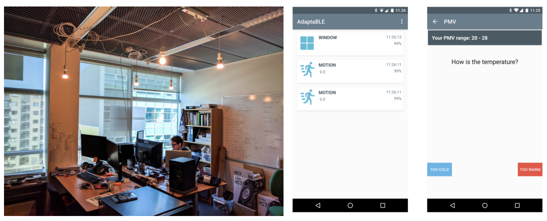


Figure 4: Office Testbed and Android Application

implementation in Section 4.

3.3 Comfort and Energy Optimization

We model comfort and energy optimization as constrained optimization, where the individual comfort preferences are hard constraints of a building-wide energy optimization. We argue that occupant comfort is more significant than a low energy consumption. When comfort requirements of occupants are divergent, our system provides strategies to mediate them. It first tries to resolve them by exploiting existing one-to-one mappings (e.g., two persons with different lighting preferences, but individual desk lights). In case the conflict cannot be solved in this way (e.g., only a single heating for a room), our system provides aggregation based strategies (e.g., mean/median of all user preferences).

We minimize energy use when possible (e.g., if occupants are comfortable with setting the brightness to 70%, the system uses this lower bound and not a maximum brightness).

4. IMPLEMENTATION AND EVALUATION

Our testbed consists of a shared office ($\sim 20\,\mathrm{m}^2$) with four smart lights (Philips Hue), three smart thermostats (eQ-3 Homematic) and a window that we modified so it can be closed and opened remotely (see Figure 4).

Our prototype uses acoustic signals to transmit an alternating key to smartphones in the same room. This key is needed to prove that one is inside the room.[1] The smart infrastructure in the room can be accessed through Bluetooth Low Energy (BLE) from an Android application on the user's smartphone. This application further uses the phone's light sensor to measure brightness levels at the user's location and stores the user's comfort preferences for different rooms. We use a room coordinator (a Raspberry Pi) to coordinate requests between different users and mediate conflicts. Users check-in to the room via our application and can either manually switch identified appliances or let the system decide on the best values based on comfort preferences. Preferences are build up incrementally based on relative user input ("I am too hot/cold.").

4.1 Graph Exploration

Algorithm 1 shows our graph exploration algorithm in a simplified form. As a first step, vertices are put into different groups according to their environmental dimension. In an unexplored graph (room), each vertex is assigned a random probability and added to a priority queue. The algorithm then iterates through that queue, modifying each appliance state slightly and taking the sensor reading and human input as feedback. If a threshold is reached, the appliance setpoint is iteratively incremented/decremented further. Both steps are repeated until the individual preference value has been reached. (for lighting this process takes $<2\,\mathrm{s}$; heating requires a time delay between actuations.) If the individual comfort level cannot be reached by these steps, we increase the modification level. This allows to identify appliances that are not as strongly connected (e.g., a light with indirect radiation). The resulting incomplete graph is stored on the smartphone of each user. If a user visits the same room a second time, we use this information to speed up the identification process (we pick the most probable appliance first).

4.2 Results

4.2.1 *Identification Time and Energy Efficiency*

Users require a timely identification time and are annoyed by too frequent modifications of the environment to identify logical appliance relations. We evaluate different strategies: (i) *Random* (we start identification with a random choice), (ii) *Explore All* (we explore the full lighting graph), (iii) *Probabilistic* (we learn by past results). We perform experiments at each of the three work desks of our office. Figure 5 shows these strategies in their time dimension. The time for exploring the full graph grows linear with the number of appliances. We implemented state transitions smoothly, over a longer time period (1.5 s) to keep human annoyance to a minimum, which is why identifying four lights takes 7 s. The results of the random strategy are widespread. In best case we initially pick a light with a strong logical relation to the user, but in worst case, we iterate until the last light. In the probabilistic approach the light can mostly be identified with the first, and latest with the second try.

Exploring the whole graph can improve energy efficiency. Figure 6 shows the same strategies in their efficiency dimension. We define 100% energy efficiency when $\sum E.cost$ is minimal for the set of lights and setpoints. Exploring all

[1]Because of the natural attenuation of sound waves by walls, this assumption is not unreasonable (see [5] for a detailed description of this setup).

Algorithm 1 Graph Exploration

Precondition: User H enters room R with $|V| = n$ vertices; $H = \{(s_1, p_1, E_1), (s_2, p_2, E_2), \dots\}$, where (s_n, p_n) are current comfort state and preferences for different dimensions, $E_n = \{e_1, e_2, \dots\}$ is a set of known edges. If R is unknown to H, this set is empty.

```
function IDENTIFY($(s_1, p_1, E_1), V$)
    if $E_1 = \emptyset$ then
        $V_1 \leftarrow V$
    else
        $V_1 \leftarrow E_1.V_1$
    end if
    for $i \leftarrow 1$ to $|V_1|$ do
        $e_2 \leftarrow$ TOGGLE($(s_1, p_1, E_1), V_1[i]$)
        if $e_2 \neq \emptyset$ then
            $E_n \leftarrow E_n + e_2$
        end if
        $s_2 \leftarrow$ GETSTATE()
        if $s_2 = s_1$ then
            return $E_n$
        end if
    end for
    return $E_n$
end function

function TOGGLE($h_1, v$)
    $e \leftarrow$ NEWEDGE()
    do
        $s_1 \leftarrow$ GETSTATE()
        ACTUATE($v, 0.1$)
        $s_2 \leftarrow$ GETSTATE()
        $e.cost \leftarrow (s_2 - s_1)/0.1$
    while $abs(s_2 - s_1) > 0$ & $s_2 < h_1.s_1$
    return $e$
end function
```

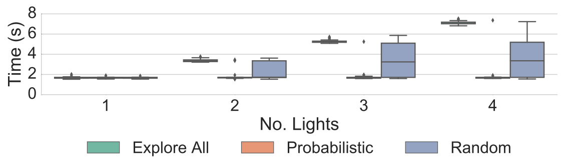


Figure 5: Identification time for different strategies.

the graph ensures that the light (s) with the lowest cost are always chosen (100%). Random strategy results become more spread with a growing number of lights (mean value 78%). Our probabilistic strategy achieves 94% mean. The relative high result for the random strategy are due to the small number of lights in our testbed and because we use a small threshold before we assign a logical relation between an appliance and a user.

4.2.2 Conflict Resolution

In our testbed, we are able to mediate lighting conflicts by exploiting one-to-one relations. However, room heating affects all occupants. To resolve conflicts we set the heating setpoint to the median of the current users. This maximizes comfort for most people, while outliers are not weighted as heavily as when using the mean. This is especially important to mitigate the impact of users that game the system by providing extreme preferences to either end. Applying this strategy to our survey dataset, results in a median of 21 °C, which is inside the comfort ranges of 36 out of 50

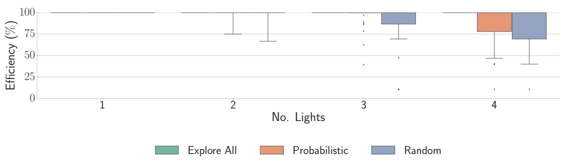


Figure 6: Energy efficiency for different strategies.

persons. The temperature is 3.3 °C lower than the mean measured temperature. We thus improve comfort from 20 to 36 persons (80%) while saving heating energy (results were obtained during heating season in Denmark).

5. CONCLUSION

We presented a model for smart-appliance interaction that reduces the need for metadata and builds on a logical human-appliance relation for identification. We have shown an initial implementation of such a model using smart lighting and heating. Looking forward, we believe that central building management can be partly substituted by such a decentralized model that combines local user preferences with global energy goals. We expect that humans will be increasingly equipped with wearable sensors that enable an identification of logical relations in all relevant comfort dimensions. This also opens up a move from traditional RF based localization methods (RSSI fingerprinting) to ambient based ones (Ambient fingerprinting).

References

[1] Comfy. https://comfyapp.com/.

[2] U. E. P. Agency. Building energy data book, 2011.

[3] M. Azizyan, I. Constandache, and R. Roy Choudhury. Surroundsense: mobile phone localization via ambience fingerprinting. In *MobiCom'09*. ACM, 2009.

[4] V. L. Erickson and A. E. Cerpa. Thermovote: participatory sensing for efficient building hvac conditioning. In *BuildSys'12*. ACM, 2012.

[5] J. Fürst, K. Chen, M. Aljarrah, P. Bonnet, et al. Leveraging physical locality to integrate smart appliances in non-residential buildings with ultrasound and bluetooth low energy. In *IoTDI'16*. IEEE, 2016.

[6] U.S. EPA. The inside story: A guide to indoor air quality, 1988.

[7] J.-t. Wang, C.-N. Shyi, T.-W. Hou, and C. Fong. Design and implementation of augmented reality system collaborating with qr code. In *ICS'10*. IEEE, 2010.

[8] R. Want, K. P. Fishkin, A. Gujar, and B. L. Harrison. Bridging physical and virtual worlds with electronic tags. In *SIGCHI'99*. ACM, 1999.

[9] H. Wirtz, J. Rüth, M. Serror, J. Á. Bitsch Link, and K. Wehrle. Opportunistic interaction in the challenged internet of things. In *MobiCom'14 workshops*. ACM, 2014.

[10] B. Zhang, Y.-H. Chen, C. Tuna, A. Dave, Y. Li, E. Lee, and B. Hartmann. Hobs: head orientation-based selection in physical spaces. In *ACM symposium on Spatial user interaction*. ACM, 2014.

Chapter 5

Conclusion

Based on our original thesis, we addressed two classes of problems: (i) Using off-the-shelf IoT systems for human-building interaction and (ii) aligning this interaction with Mark Weisers vision of ubiquitous computing [8]. We approached these problems from a system perspective, by exploring the design space for IoT based human-building interaction on different dimensions: (i) Human-building interfaces, (ii) Means of identification and (iii) System architecture. We explored each dimension by designing and implementing systems according to their requirements by adapting Weiser's core ideas and issues to IoT based human-building interaction in 2016. For each system, we then evaluated specific, non-functional requirements experimentally. These experiments and evaluation confirm our initial thesis that a seamless augmentation of physical human-building interaction with digital capabilities through IoT off-the-shelf systems can be derived from Mark Weiser's vision of ubiquitous computing.

5.1 Summary of Results

We started out by creating *BUSICO 3D*, a building virtualization that provides a natural visualisation of past and current sensor data, building wide control, configuration and scheduling. Changes made in the physical world are reflected in the virtual world and likewise. Our toolchain derives its virtual replication from an existing building information model (BIM). This facilitates a fast deployment across different buildings and the introduction of a simplified energy and thermal simulation mode, based on the structural model of the BIM. BUSICO is implemented in the modern game engine Unity 3D.

With *Babel*, we then built a system that incrementally, and by crowdsourcing the building occupants, creates a link between a digital point in a building management system (or likewise a system of IoT devices) and the physical device and its location. This process eventually achieves a consistent metadata state. Such consistent metadata enables portable cyber-physical applications and eases the continuous building commissioning task.

In the last part of this thesis, we approached the architectural integration problem of off-the-shelf smart appliances in non-residential buildings. We took this problem as a starting point to develop *BLEoT*, a Bluetooth Low Energy (BLE) based, decentralized system of smart appliances and sensors that can replace a central BMS, while setting the focus on direct user control to achieve an adaptive building management.

Our system implementation and evaluation of BLEoT showed some inconsistencies in the native Bluetooth abstractions on current smartphones. Hence, with *BLEva*, we then provided a detailed, in-the-wild evaluation of BLE in the context of smartphone-peripheral systems. We also designed a distributed benchmarking framework that enables researchers to collect reproducible results. In a second step, we used these results to implement a prototype of improved, dynamic BLE abstractions that result in a more predictable BLE behavior.

We finished this thesis by presenting *"A Practical Model for Human-Smart Appliances Interaction"*. Its appliance identification is based on a logical human-appliance relation and relies on the decentralized architecture developed in BLEoT. We also present an initial implementation using a smart lighting and heating system in a shared office that shows improvements in occupant comfort and energy consumption.

5.2 Lessons Learned and Open Issues

We chose to design our building-wide interface BUSICO 3D as a virtual reality system following Weiser's argumentation of an acceptable use case for virtual reality [8]. In hindsight, the implementation in Unity 3D was the right design decision. It enabled a fast prototyping and allowed us to make use of Unity's graphics, physics and lighting engine. We deployed our application in form of a native binary. Since our implementation of BUSICO in 2014, GPU-accelerated web-content has caught up dramatically in performance and ease of implementation (e.g., using the popular Three.js library that provides simple and browser-independent abstractions on top of WebGL [7]). Unity allows the export to WebGL since version 5 (released in 2015). Moving BUSICO into a web browser would increase accessibility by making it directly available on a multitude of devices (traditional computers, tablets, smartphones), without requiring the installation of a native binary or browser extension. An open issue is to better automate the mapping of the physical building to its digitalized representation. For now, we assume the existence of a BIM, whose namespace is consistent with the BMS. This assumption does not apply in many cases.

We believe that crowd-sourcing was the proper approach to solve inconsistent metadata and to facilitate identification between users and digitally enhanced appliances. Physical environments are never static, but always in change. Devices get moved or replaced. Hence, in all smart environments, the digital state needs to be continuously kept in synchronization with the physical world. Current off-the-shelf smart appliances are not location aware themselves, which makes crowd-sourcing a viable option. Considering the rapid growth of connected IoT devices, it will become critical that devices get location aware and self-descriptive. Humans will not be able to scope manually with the high number of devices.

The choice of BLE as a communication protocol in our decentralized IoT architecture had advantages and disadvantages: BLE showed a low energy consumption and allowed us to deploy our system with unmodified smartphones. However, our design decisions were constraint by the design space predetermined through the Bluetooth specification. Especially direct peripheral-to-peripheral connections, multi-hop and extended broadcasting messages would have been advantageous for parts of our design. Some of these issues are going to be solved with newer BLE specifications (e.g., Bluetooth 5 extends the broadcasting capability eight-fold [1]). After our implementation of BLEoT, the Bluetooth SIG proposed similar ideas by introducing RESTful API's for enabling BLE devices to communicate with web-services, and likewise, web-services with BLE devices [2]. Their white-paper proposes a stationary gateway device, whereas we directly use the user smartphones. We believe that our opportunistic architecture has several advantages, especially in a non-residential building: (i) it increases security by physically disconnecting devices from the web when users are not present and (ii), it enables individual data sharing policies on personal phones. Individual data sharing policies can define a data-flow control point between local infrastructure and the cloud. We currently have not implemented such policies, but our architecture allows for their inclusion. Another issue that we have not touched with BLEoT is the integration of an existing BMS with smart appliances. Such incremental addition of appliances is likely to happen in the near future to replace or extend existing building infrastructure.

Our human-smart appliances interaction model is a start in the right direction, but might need to be extended towards multi-objective optimization and local, peer-to-peer networking. In our architecture, smartphones act as personal gateways between smart infrastructure and the cloud, and as user interface. Therefore, our system should include local, conflict mediation algorithms that rely on direct phone-to-phone communication. BLE is an obvious communication protocol for such algorithms, due to its omnipresence and short range properties. The capability to switch between an advertising and scanning role was introduced with Android 5 (API level 21) and can as such be used to enable this local and dynamic BLE network of smartphones.

5.3 Research Perspectives

We see several directions for future research. One is rooted in the placement of computation in distributed IoT systems. Should computation be placed locally or in the cloud? As we have shown in this thesis, the placement has several implications on usability, fault-tolerance, scalability and privacy. Usability is affected by the latency of IoT systems (humans are able to perceive latencies greater than 100ms [6]). Systems that rely on the cloud may further not be as resilient as purely local systems and eventually reach a scalability problem. However, maybe the most important implications of this trade-off are the impacts on privacy of user data and the security of IoT devices. Promising approaches introduce a local computational tier, so called cloudlets [5] or fog computing [3].

Virtual Reality (VR) and Computer Vision (CV) as means of human-building interaction are other directions for further research. One problem is how to deal with an ever changing environment. These environment changes require that the digital representation for both VR and CV based approaches is continuously kept in synchronization with the physical state. Because of their high computational requirements, and the strong privacy implications of VR and CV data, the trade-off between local and cloud computation is complex. Further, current academic systems are based on a relatively small amount of data. How can we scale systems from a single building to an entire city?

Another research direction lies in authentication and authorization issues. If we assume that all of our environment transforms into a smart environment, then how should we authenticate and authorize users? Many devices need to be accessible to all occupants (e.g., lighting, heating, window control), while others should in many cases require authentication and authorization (e.g., a projector during a lecture). This thesis has only focused on the former ones. Should such an authentication and authorization system be centralized or decentralized? Should it rely on asymmetric encryption or build on a decentralized blockchain for trust (e.g., [4])? We believe that it is crucial to (i) find the right trade-off between security and simplicity/usability and (ii), to choose the adequate design and technology for implementation.

References

[1] Bluetooth SIG. Bluetooth 5 Coming Soon. `https://www.bluetooth.com/news/pressreleases/2016/06/16/-bluetooth5-quadruples-rangedoubles-speedincreases-data-broadcasting-capacity-by-800`. 2016.

[2] Bluetooth SIG. Internet Gateways. Bluetooth White Paper. 2016.

[3] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the mcc workshop on mobile cloud computing*. ACM, 2012, pp. 13–16.

[4] Ethereum Project. `https://www.ethereum.org`.

[5] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. The case for vm-based cloudlets in mobile computing. *Ieee pervasive computing*, 8(4):14–23, 2009.

[6] B. Shneiderman. *Designing the user interface: strategies for effective human-computer interaction*. Pearson Education India, 2010.

[7] Three.js. Javascript 3D library. `http://threejs.org`.

[8] M. Weiser. The computer for the 21st century. *Scientific American*, 265(3):94–104, 1991.