The IoT Codex: a Book of Programmable Stickers for Authoring and Composing Embedded Computing Applications

Kristin Williams[1,3]
[1]*Computer Science*
Emory University
Atlanta, USA
kristin.williams@emory.edu

Jessica Hammer[2,3]
[2]*Entertainment Technology Center*
Carnegie Mellon University
Pittsburgh, USA
hammerj@cs.cmu.edu

Scott Hudson[3]
[3]*Human Computer Interaction Institute*
Carnegie Mellon University
Pittsburgh, USA
scott.hudson@cs.cmu.edu

***Abstract*—The Internet-of-Things (IoT) promises to enhance everyday objects with computing, but rarely enables directly authoring or composing that behavior. Lightweight IoT approaches attach identifiers (*e.g.*, RFID tags) to objects to enable networked services. Typically these tags are passive, and so, depend on activity recognition and predefined context. This limits interaction to invoking predetermined behavior. Instead, this work presents The IoT Codex: a lightweight approach to customizing everyday objects with IoT by enabling interactive attachable IDs (aIDs) to compose software-supported behavior *in situ*. This work contributes 1) paper engineering techniques to construct aIDs that embody state, and 2) a tangible, end user programming (EUP) language for customizing IoT within symbolic and idiosyncratic contexts. Here, we provide preliminary validation of our approach with an empirically informed design space, sample applications, and a small co-design workshop. In doing so, we offer preliminary evidence for tangible, end user programming to enable meaningful control over IoT services.**

***Index Terms*—paper interfaces, tangible user interface, end user programming, RFID, customization, interactive book, battery-free user interface, wireless backscatter**

I. Introduction

The Internet of Things (IoT) promises to extend computing to everyday objects like those in the home. Yet, these haphazard environments consist in a variety of artifacts and contexts. To enable an IoT ecosystem in these environments, lightweight approaches use attachable identifiers (aIDs) like quick response (QR) codes, radio frequency identification (RFID), and Bluetooth low energy (BLE) tags. These usually passive, aIDs attach wireless communication to everyday objects to support internet services, and typically depend on machine learning models correctly recognizing intent to support higher level interaction (see [14], [56], [57], [83]).

Yet, many home objects are part of idiosyncratic situations that do not generalize. This makes characterizing significant and repeatable activities suitable for recognizing interaction intentions with an object difficult and calls for a need to customize IoT. Often, home objects are used symbolically and contextually situated within small group norms [101]. Family "clutter" organizes home life amid unpredictable schedules and breakdowns in routines. This context is unlikely to yield appropriate training data for machine learning (ML) [90], [103]. Yet, many approaches need data close enough to new cases for recognition. Even if this data could be collected, the underlying representations of ML models make semantic claims about household data that families need to be able to recognize and manipulate to effectively engage with them. However, activity recognizers' internal representations persistently prove difficult to understand [2], let alone customize.

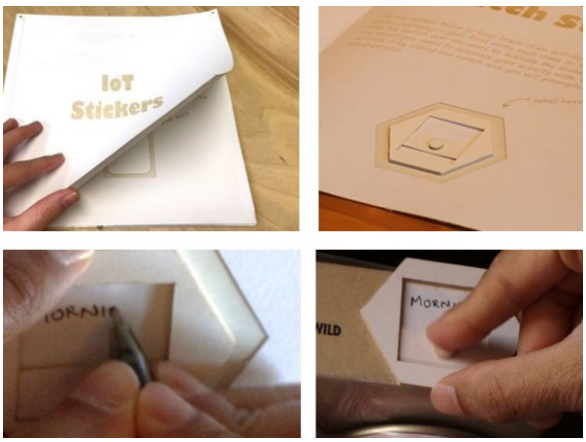

Fig. 1. The IoT Codex is a book of inexpensive, battery-free sensors and interaction patterns to support linking everyday objects to software and web services using stickers. To use, a sticker is selected, customized, peeled from the page, attached to an object, and then invoked using its kinetic mechanism.

We introduce a lightweight approach to customizing everyday objects with IoT by enabling interactive aIDs to compose software-supported behavior. Our system, The IoT Codex, contributes 1) paper engineering techniques to construct aIDs that embody state, and 2) a tangible, end user programming (EUP) language for customizing IoT within symbolic and situated contexts. As an interactive stickerbook, *The IoT Codex* affords customizing, composing, and invoking IoT behavior

This work was funded in part by National Science Foundation grant IIS-1718651.

1943-6106/23/$31.00 ©2023 IEEE
DOI 10.1109/VL-HCC57772.2023.00009

through interactive aIDs—called *IoT Stickers*—that embed battery-free, wireless sensing in kinetic, paper mechanisms. Below, we show how The IoT Codex enables authoring and composing embedded computing applications for meaningful control over IoT services.

II. Related Work

We review smart home EUP, aIDs, and kinetic mechanisms to show how paper engineering techniques could address idiosyncratic smart home needs.

A. End User Programming for Home

Home life strains IoT's EUP systems. Families want to automate household drudgery beyond current toolkit support [85]. Even common tasks, like making coffee, resist distillation into a supportable routine [21], [99], as families improvise schedules and negotiate breakdowns to accommodate one another [21], [55], [85], [100]. Instead of AI-complete activity support, home IoT should aid object-centered care, maintenance, and repair to preserve possessions and creativity [19], [22], [29], [47], [92], [93]. Family member roles and identities are shaped through the selection, use, and placement of their stuff, and could be eroded by centralized AI-complete systems [42], [90], [100], [103]. Family 'clutter' gives the home a sense of place and meaning, of *what it means to program the home* and *who should be responsible* [6], [82], [90], [105].

Ignoring clutter, smart home EUP (SH-EUP) favors abstractions that encapsulate control flow and simplify syntax. For example, trigger-action programming (TAP) uses a form for authoring conditional statements with a graphical user interface (GUI) representing situated context (see [1], [23], [73]). Yet, form filling techniques cannot handle complex procedures [67]. TAP struggles with home processes and sequencing needs such as grouping rules or adding temporal and hierarchical dependencies [16]. While usability studies criticize TAP's syntactical ambiguity [27], [38], social context may be more important. Most people borrow strangers' TAP recipes rather than author their own, and thereby expose their home to security vulnerabilities and exploits [96]. TAP inherits programming stereotypes that concentrate EUP skill and control in one family member, even though shared access is needed [65], [82], [100], [103], [105]. To do so, SH-EUP should support 1) reflection on routines, 2) family emotional connection, 3) attachable/detachable components, 4) use-based debugging, 5) family feedback, 6) interpreting system state, and 7) diagnosing networking problems [105]. Embodied interaction techniques are needed to share SH-EUP control.

B. Concretizing EUP with Identifiers

Concretization shapes usability and sharing. Icons concretize both visual meaning and control of executable programs [51]. When backed by a programming language, concretized user interface (UI) elements coordinate parallelizing team effort through shared components that can be reasoned over and worked on separately [72]. Concrete UI creation *in situ* can foster a climate in which more expert members assist lower skilled members with gaining greater control over software systems [26], [61]. Skill development can be scaffolded by copying, reusing, and tailoring an expert's pre-programmed UI behavior [61], and nurtured with UI metaphors that combine in ways—jigsaw puzzle pieces or magnetic poetry—that balance abstraction with low-level exposure [39], [95]. Block languages use a block metaphor to aid recognition over recall, meaningfully chunk code, and constrain composition to avoid errors [11]. They afford 1) drag and drop composition, 2) interactive editing, 3) nesting with recursive tree structures, and 4) geometrically preventing syntactic errors [80]. Yet, GUI concretization for smart homes contradicts mental models of installing home appliances and splits attention between screens and the physical world [46]. Instead, RFID could enable a tangible and mechanical approach [46].

EUP abstractions are not easily concretized with aIDs. As an unpowered, passive UI without a continuous, system link (*e.g.*, QR codes, RFID), they are difficult to interpret due to their weak preservation of manual interaction or *isomorph effects* [35], [36]. For example, I/O Brush preserves manual painting gestures [84] while aIDs' commands mimic manipulating a token [12], [36]. Machine learning techniques recently expanded aIDs' repertoire enabled gestural commands [56], [57], [106], spatial layout [14], [35], [83], and shape deformation [43]. Physically severing aIDs enables bi-stability [37], [45], [57], [108]. Or, a biased reed switch inserted between RFID's antenna and IC disables reads when 2 tags are close [58]. These techniques support aIDs' direct manipulation, but introduce *indirection* by offsetting the system's response both spatially and temporally (*e.g.*, screen-based output) [12], [35], [36]. Indirection subverts the live feedback critical to block-based programming's success [81]. aIDs' isomorph effects and indirection undermine live feedback.

C. Kinetic Mechanisms for Live Composition

Kinetic paper mechanisms address indirection and isomorph effects challenges. Their 3D moving parts provide live, interactive feedback without a screen [77]. Activating their kinetic mechanisms can resolve uncertainty in aID detection and generate enough kinetic energy to power embedded electronics [44], [56], [87], [108]. A pop-up mechanism's bi-stability strengthens isomorph effects by embodying system state and mapping to UI elements' state machines [68]. Pattern languages for CAD modelling and mechanism design have drawn on this feature of centuries of papercraft [5], [32], [54], [70], [104], [109]. Recent work fabricates, assembles, and activates kinetic mechanisms to test modelled behavior [8], [69], [97]. To date, kinetic mechanism research focuses on sensing and shaping input, but these have not been used to enable EUP.

Interactive books introduce programming concepts with tangible interaction [9], [48], [60], [88], [89]. Pop-up mechanisms enable tangibly manipulating sensors [77]. Stickers support tinkering and peel-and-stick construction of both circuits and remote messaging [25], [33], [78]. Spatially arranging and fitting together stickers scaffolds programming concepts like sequencing and syntax [34], [79]. Their shape and page

slots limit how components fit together [34], [91]. These familiar crafting techniques support user agency by enabling proprioceptive control over dynamically mapping behavior and the programming environment [52]. Interactive books leverage paper engineering's material constraints to expose and exclude available actions.

III. Design Rationale for The IoT Codex

We iteratively designed The IoT Codex to suit American families' idiosyncratic contexts and households. The design rationale describes how family feedback, task context, and home life shaped our redesign decisions for The IoT Codex.

Kinetic Mechanisms for Interaction: Kinetic mechanisms enable aIDs to embody state. This visible state makes the state of the system transparent and concretizes our system's programming abstractions in a tangible form (*c.f.* [24]).

Spatial Management: We enabled *in situ* tracking and room-level support for interactive aIDs to accommodate how families control the home's rooms to nurture their interests and manage family norms [17], [20], [101]:

- ***In Situ State Tracking:*** We created a more compact footprint for aID's paper-based interface to make them easier to attach to and use with objects *in situ*. This gives spatial management an implicit role: when aIDs attach to objects and have a place, they can carry symbolic, contextual meaning associated with their placement.
- ***Room Level Support***: The RFID reader was moved from a mobile platform to a stationary, room-based hub to support interaction at the room level using the fixed, sensing location of interaction with aIDs and a room-level, sensing range. This aligns sensor setup with the room level norms and control that families exert over objects belonging to that room.

Grounded Patterns: Pre-programmed patterns and templates were grounded in common household objects instead of employing interface metaphors. Grounding situates IoT's novel functionality in context and everyday object usage to defamiliarizes domestic possessions and their related routines (*c.f.* [4], [10], [13], [61], [100]).

Tangible Composition: aIDs were designed to be both physically arrangeable and virtually composable with other aIDs so that tangible composition supports thinking through the process of scheduling and assigning household activities (see editability for visual formalisms [67]).

Event-oriented Architecture: By hiding information about triggers' sensed data from related, reactive higher level actions, TAP architectures hide uncertainty about what is sensed. Instead, event-oriented architectures can 1) pass along uncertainty information so that it can be resolved (see [62], [86]), 2) enable developers to work with inputs, events, and behavior separately (see [49]), and 3) enable an end user to directly indicate to the system what to do (see [67]).

Summary: The IoT Codex design rationale translates findings from participatory design with families to resulting principles embodied in the sticker book.

IV. The IoT Codex

The IoT Codex houses a set of stickers—we call them IoT Stickers—that enable embedding internet capabilities in the environment by sticking them to an object. IoT Stickers incorporate existing possessions into the IoT ecosystem (*c.f.* [10]) by associating a unique identifier with both its physical and digital counterparts. IoT Stickers employ RFID tags which can be purchased for <$0.03 in bulk. These passive, radio tags communicate a small amount of data——the tag's unique ID–—when scanned by an RFID reader. In contrast to background infrastructural sensing (see [41], [99]), IoT Stickers enable explicit installation of wireless, network nodes by enlisting user choice of what to attach them to and where. Although minimizing time burdens, IoT Stickers do not eliminate the choice and agency both necessary to family collaboration and important for installation when modifying their home.

A. Sticker Fabrication and Operation

IoT Stickers are the codex's interaction primitives. They are the user interface building blocks for a user's IoT application. Like links developed for paper user interfaces (PUIs) (see [31], [60]), IoT Stickers connect everyday objects in the home to both other stickers and electronic programs. Below, we detail the button sticker to illustrate the underlying architecture.

The button sticker—like all IoT Stickers—uses a layered design so that electronics can be embedded in a fabrication friendly manner (*c.f.* [74]). Each sticker type modifies a basic set of layers: sticker paper, double-sided adhesive with a slot for an RFID tag, and a top layer for aesthetic customization like drawing, writing, or icons. We use RFID tags as a proof-of-concept for cheap, battery-free, wireless electronics capable of embedding digital data in our sticker book without requiring a line of sight. However, many of The IoT Codex's conceptual properties—like scaffolding customization and composition—could be adapted for other implementations of an interactive book with aIDs (*e.g.*, QR codes).

RFID Overview: RFID systems consist of readers and tags. We use passive, ultra-high frequency (UHF) tags capable of wireless communication up to a range of 11 meters with an appropriate antenna setup. Prior work showed how a reader and antenna could be embedded in form factors like a light bulb equipped with a wifi RFID reader for easy home deployment [30]. Building on this, our work focuses on the tags and interpretation of tag data. Passive RFID tags communicate with the reader when it interrogates the environment by emitting an RF signal in the 840-960 MHz range. When the tag receives the signal, it communicates with the reader through modulated backscatter—changing how the signal is absorbed or reflected back to the reader in order to encode a unique tag ID. Conductive material interferes with the tag's communication, and we use this property to represent IoT Stickers and their parts as having a simple binary state: covered or uncovered (for other manipulations, see [56], [87]).

Codex and Sticker Interaction Design: The codex houses each sticker type in layered paper, fold-out sections to embed other materials into the pages to facilitate interaction. Slots,

layered wax paper, and double-sided adhesive enable stickers to be more easily removed and reattached. Imitation gold leaf (ultra-thin copper foil) layers prevent a sticker from being read by the reader when the book is opened. In contrast to aIDs, we call this composite layered, paper device an IoT Sticker (*c.f.*, [40] for the kinetic mechanisms' role in supporting material-like v. machine-like interaction). For example, the codex button page's structure and text guides the user to peel the sticker from the page and attach it to the object of their choice to create a sticker button. Its blank face allows anything to be written or drawn on it. The first button uses a logging pattern to track events with user-specific meaning and function with minimal user intervention (see [3], [59], [94], [98] on the value and functionality of paper-based logging). When a sticker is pealed off the page, the sticker's introduction and setup process launches. After setup, the sticker's behavior can be invoked. For the logging sticker, data is logged and an audio chime plays whenever the user covers the sticker with their hand. This progressive disclosure provides encouragement and incentive to continue with the button setup process (see [102] on the role of curiosity in EUP).

B. IoT Codex Architecture

Events and Event Handling: Implemented in C#, IoT Codex's architecture is event oriented by using an event queue to separate hardware concerns from the rest of the system. Unlike GUI event-oriented architectures that use semi-standardized input hardware, the codex's system has its own hardware abstraction layer to produce events. To enable greater interactivity, the codex supports manipulating parameters to the actions carried out by stickers. Below, we characterize The IoT Codex's underlying architecture in greater detail. This architecture could be extended or adapted to other interactive books, tangible user interfaces, or other embedded computing applications by researchers or programmers.

Hardware abstraction layer: The codex's hardware abstraction layer interprets data from input devices that identify objects (here, RFID tags) to produce events. The layer tracks an object's visibility to a reader and generates events based on the identified object. In The IoT Codex's implementation, this layer manages connecting to a ThingMagic M6e RFID reader, and so, also the book's physical, sticker devices. This layer interprets the reader's RF interrogations to generate events. Unlike typical input handling—like key press events—the hardware abstraction layer also generate events when a previously identified object is no longer visible to the reader. Since RFID reads can be noisy, the layer interprets incoming tag read information before emitting events to handle the reader occasionally failing to read visible tags. Bayesian machine learning models can decide whether a tag is present or not within 300ms [87]. Here, we use a simpler, time-based hysteresis mechanism to stabilize reads (as in [56]). To do this, a state machine tracks a tag's current state as seen by the reader and as seen across the reader's multiple rounds of RF interrogation. When an event is emitted, reader metadata—like RSS and signal phase shift—are also passed along the event queue so that sticker types have the information to make more sophisticated interpretations of the reads (see machine learning use cases in [56], [57]).

Life of a sticker object: The main component of the system 1) keeps track of the life of a sticker object, 2) dispatches events according to a look-up table mapping identified objects to the system's sticker objects, and 3) both updates an object's state and invokes its behavior.

When The IoT Codex first detects an identified object, it uses a look-up table to determine whether the associated sticker object needs to be created. As we will detail later, a single physical sticker can be implemented using multiple parts which are identified separately, and which may become visible or invisible when the sticker is manipulated. The object's associated Sticker object is instantiated at the first appearance of an identified part. All of an IoT Sticker's identified parts are pre-registered and linked to a data structure describing both the type of sticker object they belong to and how it is to be instantiated. When instantiated, a sticker registers the identifying information associated with each of its parts in a second, lookup table. This table is then used to dispatch events to the appropriate sticker object whenever they arrive. The sticker responds to incoming events by updating its internal state based on its state-machine and internal variables, as well as invoking actions.

Actions: Like many procedural languages, the system establishes a set of parameter values, then passes them to the relevant method/actions. These values can be determined by expressions over literal values, variables (taken from local and global variable spaces), constants, and immutable values established at sticker instantiation. The system also supports some unusual ways of establishing parameter values. For example, parameter values can be established, yet *asynchronously evaluated*. For example, some values need to be obtained externally from services or sensors, and so, require asynchronous communication. This necessitates pausing the process of gathering an action's parameters while the system awaits the value. Similarly, for parameters supplied by the user, the system must wait while it initiates interaction with the user to get the value. In addition to asynchronous values, the system enables parameters with values which are established upon their first use, and then reused. This provides for lazy or *just in time* evaluation which only ask the user to supply these when it is definitely needed to be sensitive to the cost to the user that it imposes.

Like asynchronous parameter values, we enable actions to pause execution without blocking the full system. To do so, The IoT Codex system employs actions in a way that is equivalent to independent threads. We implement this internally by breaking actions into "chunks" and providing a simple scheduling mechanism for these chunks. For example, physically manipulating stickers can dynamically compose new capabilities using the codex's composition operations. Yet, the UI abstraction is in one concrete form—the sticker—and does not separate stickers from the (less concretized) actions they perform. So chunking provides more flexible manipulation of

Fig. 2. All five stickers are shown—button, toggle, list, dial, and wrapper—with their kinetic mechanisms in use.

actions. To illustrate, suppose a date-time picker is combined with an action invoking audio playback to schedule playback at the specified time. In this example, *action chunks* support scheduling the initial request to the user for a date-time and deferring the later audio playback until the user chosen time. So, it is sometimes necessary for the underlying system to *pull apart* stickers and their actions.

Summary: The IoT Codex's architecture supports manipulating parameters and actions through proprioceptive control over exposing/blocking tag reads. It does this through a hardware abstraction layer that 1) produces events, 2) tracks the life of identified objects (in this case, RFID tags), and 3) facilitates interactive manipulation of parameters through advanced parameter types and actions that enable thread-like control. The IoT Codex extends the power of working with abstractions by concretizing the architecture's central abstractions of an identifiable object, stickers, and actions to tangible form. Tangible interaction can instantiate sticker objects, update their state, and directly supply parameter values. The codex's architecture enables a range of physical sticker complexity so that tangible designs can transition interaction from using pre-programmed patterns to tinkering with a programming language.

C. Sticker Types and Use Cases

The IoT Codex's form factor progressively introduces 5 sticker types. Page-turning sequences exposure to each sticker's complexity from the Button Sticker to the Wrapper Sticker. By embodying the link between user interaction and action execution in their kinetic mechanisms, IoT stickers' physical complexity scaffolds customization techniques from annotating/placing fixed form and function stickers to tangibly composing small programs using Wrapper stickers. Below, we describe each sticker type and examples we implemented to validate our design space (see [53], [63], [66], [71] on design space validations).

Button Sticker: As the IoT Codex's simplest sticker, the Button Sticker provides push button interaction to fire an action. Physically "pressing" or "touching" it modifies the embedded RFID tag's antenna properties to block backscatter communication. Since skin is conductive, the button sticker's press/touch action attenuates an RFID tag's response. Similarly, the other IoT Stickers' kinetic forms are designed using this binary approach of enabling or disabling transmission of identifying information (here, an RFID tag). IoT Stickers initiate action in response to physical sticker manipulation by leveraging event reporting in the system architecture described above. Changes in ID status drive a state machine for each software-side sticker object, and so, its behavior.

RecordButton We implemented a *RecordButton* Sticker to enable communication between remotely distributed people through audio messaging. When unpeeled from the codex, the RecordButton plays an audio file prompting the user to record a message using a first use parameter. Covering it begins recording and uncovering it stops recording. The recorded message is then saved as a parameter to the button's newly bound play action. Subsequently, this action can be invoked by pressing the button sticker.

Toggle Sticker: The *Toggle Sticker* has a slider mechanism that selectively interferes with one of two tags. This operates the tags in mutual exclusion of one another, and so, enables the sticker to function as a simple user controlled conditional.

CatFeeding We implemented a *CatFeeding* Toggle to coordinate family members feeding a pet cat by enabling a situated message on the food container to track whether morning/evening meals were dispensed. As an IoT backed sticker, toggling can simultaneously log morning/evening feedings to track food supplies and update a shopping list when its threshold is reached. This coordinated logging can enable shared awareness of household chore and shopping needs.

List Sticker: The list sticker has a row of 3 RFID tags covered by pop-up style flaps. Each flap lifts shielding away from the tag. Flaps enable more than one active RFID tag at a time so that this sticker can manage situations like a partially fulfilled To Do list or choosing from a fixed set of values.

BedtimeList We implemented a *BedtimeList* to scaffold children assuming responsibility for important routines. Consider when a parent needs to be away from home during a child's bedtime routine. The BedtimeList can scaffold routine and facilitate expressions of care. BedtimeList invokes an audio action for each flap: 1) play a children's audiobook *The Carpenter and the Walrus*, 2) play a 2 min. teeth brushing song, and 3) play the parent's recorded good night message. BedtimeList sequences the bedtime routine to scaffold behavior when the parent is absent. Even when an alternative caregiver is around, the sticker supports modelling fluent reading, hygiene behavior, and enables saying goodnight.

Dial Sticker: The dial sticker uses 6 circularly arranged RFID and a rotating lever mechanism. This moving arm lined with foil deactivates a tag by covering it. The dial arm selects

among a small set of values or states. This could control the place of play in a video/audio file (*c.f.* audio stream access with pixel access in [75]), manipulate 3D object rotation, or scrolling. Dial stickers can also describe social relationships, like identifying which person in a set is responsible for taking some action like doing a chore.

PlaybackDial We implemented the *PlaybackDial* to control audiobook playback since social reading rarely proceeds as a linear activity from start to finish. We used 4 of the available sticker slots to support 1. play, 2. pause, 3. unpause, and 4. stop. The *PlaybackDial* enables an audiobook to supplement a physical book's text by providing functionality for reading with audio on demand. With a cheap form factor, IoT Stickers could duplicate this functionality for many books to encourage development of visual literacy skills.

Wrapper Sticker: The Wrapper Sticker's larger shape contains a slot for another sticker to fit in it. Its software-backed object modifies that of the slotted sticker's action in a simple and generic way. The Wrapper's form enables concretely composing and manipulating sticker behavior. For example, a wrapper prompting the user for a date-time could support 1. *Delay*, waiting the specified time period before invoking an action or, 2. *OnlyOncePerDay*, blocking subsequent action invocations until the specified time passed. Wrapper stickers modify existing sticker functionality through simple compositions of sticker functions in a concrete and physical form.

SpaTimeWrapper We implemented the *SpaTimeWrapper* to use IoT behavior to support temporary, leisure spaces in their home. The wrapper has a date-time parameter for scheduling the bathroom for a bubble bath so that the sticker's spa settings run on the date and time planned for. We implemented the wrapper as part of a paper door hanger reading "Do Not Disturb" that the user can punch out of The IoT Codex. The slotted spa settings sticker plays a list of relaxing music. The SpaTimeWrapper communicates a situated message on the bathroom door to other family members, but also coordinates IoT appliances at the scheduled time.

Summary: The 5 IoT Sticker types enable a tangible user interface that can customize IoT behavior by modifying their *Parameters* or *Actions*. Our implemented examples show how The IoT Codex's kinetic mechanisms gradually introduce more complex, embedded computing capabilities by leveraging the book's sequential nature and encourage tangible manipulation to tinker with the UI backed programming language. Each IoT Sticker's form factor associates the user's physical manipulation with computational actions to enable low-level control without requiring mastery of low-level programming.

V. IoT Codex Design Space

The IoT Codex's kinetic mechanisms introduce customization by concretizing programming abstractions to generate a design space (see Fig. 3). Each of 5 sticker types selectively embodies customization techniques like remixing pre-programmed behavior, first-use parameters, and composition. A primitive movement vocabulary—annotating, peel-off to initialize first-use parameters, and spatially arranging stickers

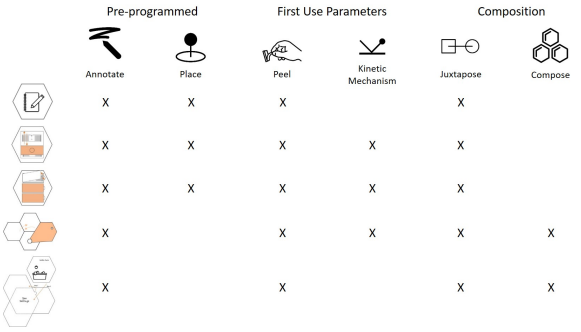

Fig. 3. The customization techniques supported by The IoT Codex's architecture and concretized with IoT Stickers' tangible, kinetic mechanisms enables a design space for tailoring IoT services to idiosyncratic needs.

to juxtapose behavior—supports customizing each sticker type with personal meaning. Each IoT Stickers' increasing, physical complexity reveals the system's support for more complex customization through place making, kinetic manipulation, and software composition. Below, we show how these techniques tangibly introduce customization as the book progresses.

A. Concretizing Customization

All five IoT Sticker types use their material form factor to embody meaning making, initialization of first-use parameters, and juxtaposing sticker behavior. As with paper user interfaces, personal meaning can be assigned to IoT Stickers through drawing and annotation. Peeling an IoT Sticker away from the page can initialize first-use parameters (*c.f.*, lift-off design [76]). This first-use response prompts the user to supply parameters to customize the sticker's behavior. For example, we implemented a text-to-speech customization that shows a dialogue box asking the user to supply a string to be spoken by the speech synthesizer when the sticker is peeled off. This parameter is stored by the system so that the sticker's subsequent invocations will cause the speech synthesizer to speak that text. The *RecordButton* also shows similar first use behavior, but does away with a keyboard and screen by enabling audio-only interaction that readily supplements the book and stickers' tangible interaction.

Physically juxtaposing IoT Stickers begins composing IoT behavior in more sophisticated ways. Placing two independent stickers close together in context supports greater functionality. For example, placing pre-programmed stickers together—like logging and play-an-audio-file—can combine their behavior. A set of stickers that play single notes can be combined to create an instrument to play a song or turn everyday objects into music-making devices. A set of logging stickers could create a self-tracking application by logging different activities. When set up to log time spent on work, chores, and play, a person could track what they spend their time on. In sticker form, this enables activities to be tracked *in situ* using the objects most associated with them such as a laptop, refrigerator, and gaming console. As needs evolve, the sticker form factor enables removal and replacement with different IoT behavior.

B. Tangible Complexity for Introducing Composition

Each IoT Sticker's tangible complexity increasingly embodies more sophisticated programming composition. Kinetic mechanisms, spatial grouping, and slotted construction introduce tangible complexity to each sticker type. In its simplest form, their base hexagonal design enables contextualization through placement. Discretion over where a sticker goes and what it attaches to leverages knowledge of social context and ritual usages of place to embed the sticker appropriately. The button, toggle, and list IoT Sticker types support place-making, contextualization through their smaller footprint.

Kinetic mechanisms on the toggle, list, and dial stickers vary the sticker's behavior depending on which part is being manipulated. These mechanisms selectively invoke different sticker behavior by triggering that sticker part's first use parameter or the action bound to that part. Slot construction guides composing objects together and is concretized in the dial and the wrapper sticker embodiments. The dial leverages turning of a slotted book page to sequence sticker placement. A semi-transparent page with slots punched out in the shape of simple stickers overlays a composition page—page printed with sticker outlines—to limit where the dial's composing stickers should be placed (implemented with 4 instead of 6 composing stickers). Slots and page-turning constrain sticker placement since the simple stickers can pass through the holes of the overlaying sheet, but the dial arm cannot. So they must be placed first. The second technique uses book interaction to reify instantiating the dial sticker: the dial arm must be torn along the designated page's perforations and its related, composing stickers are unpeeled. Similarly, holes, tearing, and shape cue composing the wrapper sticker with another, simpler sticker. The larger, wrapper sticker has a perforated, center hexagon to be torn out to create a hole for the wrappee (the simpler, smaller sticker) to go in. The wrapper is big enough to hold another sticker—the size of three haxagonal stickers together—to cue its "wrapping" behavior. The wrapper also uses a printed outline on a composition page to encourage sticking the wrapper to the page first before sticking the wrappee.

VI. Design Space Validation

For preliminary evaluation of The IoT Codex, we conducted a 2 hr. design workshop to assess the generative power of its design space (see [18]). It took place in a university setting, and participants were compensated $15/hr. We recruited 3 participants through human-computer interaction listservs and channels. We screened participants with a 2 minute survey to verify whether they had taken core courses in human-computer interaction or had comparable experience.

Procedure: Following the dialogue labs method [28], the workshop focused on the feasibility of using the IoT Codex to create home applications. First, participants were consented following the IRB approved protocol. Then, they completed a background survey on their demographics and experience with creating interactive systems. Next, the researcher showed the demo video of the IoT Codex, and then, introduced The IoT Codex, each IoT Sticker type, and that type's demo application (7 min.). Participants were given chances to manipulate the IoT Stickers and ask questions of the researcher (8 min.). Then, for each of 3 design sessions, participants envisioned applications for 1.) pre-programmed IoT Stickers, 2.) first-use parameter IoT Stickers, and 3.) composing IoT Stickers together. Each session was divided between 15 min. silent individual work and 15 min. sharing of ideas with the group (total time 90 min.). Finally, participants rated their experience designing applications for IoT Stickers.

Data and Analysis: We video recorded the study using two cameras angled to record all participants. We hired a professional transcription company to transcribe the audio recordings and then used thematic analysis to code the transcripts [15].

Findings

Participants: We recruited 2 female and 1 male identifying participants ranging from 21 to 26 years of age. Two attained a college level education and 1 a master's. Two reported 1-2 years and 1 3-5 years experience designing interactive systems.

Customizing IoT Stickers to Participants Homes: The 3 envisioned controlling IoT context by assigning indefinite concepts to IoT Stickers. A liminal space could be given context by assigning IoT Stickers its place. P2 explained, "*to be contextual...I'd probably just stick it to the door or somewhere in the entrance.*" IoT Stickers could delegate responsibility by assigning tasks to roommates (2 participants). For example, P3 employed toggle stickers to track whether roommates did their chores. An IoT Sticker was assigned to the fuzzy category, laundry, to track the task's last stopping point (1 participant). Assigning IoT Stickers to places associated with indefinite concepts enabled tracking and contextualizing idiosyncratic norms like arriving home, responsibility, or cleanliness.

All 3 participants thought IoT Stickers could enhance interpersonal communication and coordination. P1 remarked, "*It fosters good communication. It can help maybe avoid...passive-aggressiveness*". Reminding could be offloaded to the system to coordinate shared contribution. P1 elaborated, "*You can have the automated...'bing,' like, 'system has notified you that it's your turn to do the dishes'*". Instead of automating reminders, participants wanted to leverage system anonymity. For example, P3 added names to a few button stickers and linked each to a toggle sticker tracking whether a chore was assigned or done. P3 described using the arrangement to visibly reveal who had not done their chores, then pressing the linked button sticker to send a reminder to the person who hadn't when a discrepancy was noticed. Besides chores, participants listed aspirational ideas like collaboratively planning a movie night or adopting a new skincare routine. In general, the IoT Stickers were envisioned as a way to manage household communication and coordination without creating a personal task to nag or manage roommates.

Concretizing Applications: Tangible manipulation concretized IoT ideas and customized its behavior to context. P3 suggested, "*attach[ing the sticker] to the washer or dryer. It would need to be parameterized with what the object*

is." Kinetic initialization of first-use parameters scaffolded envisioning program sequences and timing: "*there's someone's birthday coming up, and I think when you added this complexity, it helped me concretize with that idea."* Though struggling with first use parameters, P2 thought the codex's page-turning structure gradually introduced more complex programming concepts. P2 compared the codex to "*teaching a programming language or something like that. We start with the simplest feature that this offers you and give a very concrete example. And then after introducing all of those through the booklet, people can get more creative."* The IoT Codex's tangible and kinetic features made assigning IoT variables, behavior, and objects to context a process of deictic reference.

Collaborative Tangible Composition: IoT Stickers' embodiment enabled collaborative composition. P2 praised their visibility: "*the advantage of this digital, kinetic device, is that you see what's done there already."* The 3 participants thought this could coordinate family members. P3 explained how "*the people closest to the control do the thing and make it transparent across everyone else."* Shared visibility facilitated collaboratively learning to translate ideas into applications. P2 exclaimed, "*I think P3's idea was really brilliant, on using this to indicate two people."* Illustrating an idea was supported by tangibly referencing an IoT Sticker. For example, P1 took the stickers P2 had been using and arranged them on the table to illustrate an extension of P2's initial idea and ask a clarifying question. Similarly, to describe composing two stickers together, P2 picked up the dial sticker and rotated the arm to show when to invoke behavior. IoT Stickers' embodiment supported sharing a programming idea. IoT Stickers' embodied context enabled shared visibility and tangible reference to support collaborating on IoT programming ideas.

Summary: The study uncovered preliminary evidence that IoT Codex and Stickers' embodied form facilitated assigning indefinite context through customization and deictic reference to physical user interface elements in a way that facilitated collaboratively creating IoT applications. This provides preliminary validation for The IoT Codex's design space by showing how the codex's primitive movement vocabulary and composition techniques facilitated generating ideas for embedded computing applications for the home.

VII. Discussion & Conclusion

The IoT Codex introduces a lightweight approach to programming embedded computing applications. By embodying programming abstractions, The IoT Codex concretizes indefinite forms of context and composing applications to scaffold programming concepts and collaboration as needed to tailor IoT to idiosyncratic norms. Below, we discuss future work enabled by The IoT Codex as well as some of its limitations.

Identifiable tags: The IoT Codex's system architecture could support other kinds of input devices that can supply a unique ID. Barcodes and computer vision techniques are obvious extensions of this work (see [49]). However, recent work on cheap, battery-free, wireless sensors opens up wider opportunities for looking at other kinds of input devices for the architecture: tags that are radio-based [107], triboelectric-nanogenerator-powered [7], or textile-embedded [50]. If these tags could be integrated with kinetic mechanisms that support bistable states (like the triboelectric-nanogenerator of [44]), this would open up a wider range for kinetic mechanisms to support embedded computing applications.

Abstraction: The IoT Codex uses kinetic mechanisms to balance abstracting low level details with supporting high level composition. These map to a basic set of interactive objects that serve as the architecture's primary abstraction so that it is quick and easy to compose a tangible interface. These abstractions serve as an important contribution of toolkit research. Yet, by hiding low level details, these very same abstractions can create barriers for learners because they hide the information needed to develop competence and expertise [64]. This work's focus on employing kinetic mechanisms for shared visibility and tangible reference suggests opportunities for new tangible kits to scaffold collaborative programming.

Limitations: This work proposed the IoT Codex for smart home and ubiquitous computing applications, but our evaluation provide only preliminary evidence. Further work is needed. For example, we assessed the Codex's architecture with a small set of tags at a time ($<$10) with supported interactions concentrated in and around the book in a laboratory setting. If the stickers were ubiquitously placed on many household items, the system would need to handle different materials, their potential interference (reducing the read range), form factors, and challenges posed by old housing stock. Future work should evaluate the system in an ecologically valid setting to uncover tag behavior in the presence of differing materials, any related interference, and whether multiple readers are needed for a room level read range. Further, each section of the book was tested in isolation from one another. A technical evaluation should be conducted to understand the challenges that may arise when the book is collated. Along these lines, this work has not yet tested the IoT Codex's deployment. Although the stickers' affordances have been well tested by the pop-up book industry, tangible manipulation in an ecologically valid setting or large scale user testing may introduce uncertain read data or false classifications. Further study would uncover these.

Conclusion

Above, we presented The IoT Codex and provided a preliminary demonstration of how it could support authoring and composing embedded computing applications. The Codex employs paper engineering techniques to create EUP abstractions and interaction techniques through a book of programmable, kinetic stickers to convey an EUP language's affordances. The IoT Codex uses both shape and kinematics to constrain authoring expressions. It thus contributes physical computing techniques to aid *in situ* creation of UIs using wireless, battery-free sensors for customizing IoT services. By supporting direct specification of indefinite context, The IoT Codex shows a way to associate IoT services with a dramatically wider set of objects and tasks than previously supported.

ACKNOWLEDGMENTS

We thank John Zimmerman, Rajitha Pulivarthy, Megan Hofmann, and Jasmine Shen for their early stage brainstorming, prototyping, and help on this project.

REFERENCES

[1] If This Then That, 2019.

[2] Ashraf Abdul, Jo Vermeulen, Danding Wang, Brian Y Lim, and Mohan Kankanhalli. Trends and trajectories for explainable, accountable and intelligible systems: An HCI research agenda. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–18, 2018.

[3] Parastoo Abtahi, Victoria Ding, Anna C Yang, Tommy Bruzzese, Alyssa B Romanos, Elizabeth L Murnane, Sean Follmer, and James A Landay. Understanding physical practices and the role of technology in manual self-tracking. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(4):1–24, 2020.

[4] Robert E Adamson. Functional fixedness as related to problem solving: A repetition of three experiments. *Journal of Experimental Psychology*, 44(4):288, 1952.

[5] Michelle Annett, Tovi Grossman, Daniel Wigdor, and George Fitzmaurice. Moveablemaker: Facilitating the design, generation, and assembly of moveable papercraft. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pages 565–574, 2015.

[6] Ian Arawjo. To write code: The cultural fabrication of programming notation and practice. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2020.

[7] Nivedita Arora, Steven L Zhang, Fereshteh Shahmiri, Diego Osorio, Yi-Cheng Wang, Mohit Gupta, Zhengjun Wang, Thad Starner, Zhong Lin Wang, and Gregory D Abowd. SATURN: A thin and flexible self-powered microphone leveraging triboelectric nanogenerator. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(2):60, 2018.

[8] Daniel M Aukes and Robert J Wood. PopupCAD: A tool for automated design, fabrication, and analysis of laminate devices. In *Micro- and Nanotechnology Sensors, Systems, and Applications VII*, volume 9467, pages 169–178. SPIE, 2015.

[9] Maribeth Back, Jonathan Cohen, Rich Gold, Steve Harrison, and Scott Minneman. Listen Reader: An electronically augmented paper-based book. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 23–29. ACM, 2001.

[10] Ralph Barthel, Kerstin Leder Mackley, Andrew Hudson-Smith, Angelina Karpovich, Martin De Jode, and Chris Speed. An internet of old things as an augmented memory system. *Personal and Ubiquitous Computing*, 17(2):321–333, 2013.

[11] David Bau, Jeff Gray, Caitlin Kelleher, Josh Sheldon, and Franklyn Turbak. Learnable programming: Blocks and beyond. *Communications of the ACM*, 60(6):72–80, 2017.

[12] Michel Beaudouin-Lafon. Instrumental interaction: An interaction model for designing post-WIMP user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 446–453, 2000.

[13] Genevieve Bell, Mark Blythe, and Phoebe Sengers. Making by making strange: Defamiliarization and the design of domestic technologies. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 12(2):149–173, 2005.

[14] Jacob T Biehl, Andreas Girgensohn, and Mitesh Patel. Achieving accurate room-level indoor location estimation with emerging IoT networks. In *Proceedings of the 9th International Conference on the Internet of Things*, pages 1–8, 2019.

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

[16] Julia Brich, Marcel Walch, Michael Rietzler, Michael Weber, and Florian Schaub. Exploring end user programming needs in home automation. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 24(2):1–35, 2017.

[17] AJ Bernheim Brush and Kori M Inkpen. Yours, mine and ours? Sharing and use of technology in domestic environments. In *UbiComp 2007: Ubiquitous Computing. 9th International Conference, UbiComp 2007, Innsbruck, Austria, September 2007 Proceedings*, pages 109–126. Springer, 2007.

[18] Stuart K Card, Jock D Mackinlay, and George G Robertson. The design space of input devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 117–124, 1990.

[19] EunJeong Cheon and Norman Makoto Su. 'Staged for Living': Negotiating objects and their values over a porous boundary. *Proceedings of the ACM on Human-Computer Interaction*, 2(36):1–24, 2018.

[20] Marshini Chetty, Ja-Young Sung, and Rebecca Grinter. How smart homes learn: The evolution of the networked home and household. In *Proceedings of the International Conference on Ubiquitous Computing*, pages 127–144, Berlin, Germany, 2007. Springer.

[21] Scott Davidoff, John Zimmerman, and Anind Dey. Principles of smart home control. In *Proceedings of the 8th International Conference on Ubiquitous Computing*, pages 19–34, Berlin, Germany, 2006. Springer.

[22] Audrey Desjardins, Ron Wakkary, and William Odom. Investigating genres and perspectives in hci research on the home. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3073–3082, 2015.

[23] Anind K. Dey, Timothy Sohn, Sara Streng, and Justin Kodama. iCAP: Interactive prototyping of context-aware applications. In Kenneth P. Fishkin, Bernt Schiele, Paddy Nixon, and Aaron Quigley, editors, *Pervasive Computing*, pages 254–271, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[24] Paul Dourish and Graham Button. On "Technomethodology": Foundational relationships between ethnomethodology and system design. *Human-Computer Interaction*, 13(4):395–432, 1998.

[25] Natalie Freed, Jie Qi, Adam Setapen, Cynthia Breazeal, Leah Buechley, and Hayes Raffle. Sticking together: Handcrafting personalized communication interfaces. In *Proceedings of the 10th International Conference on Interaction Design and Children*, pages 238–241, 2011.

[26] Michelle Gantt and Bonnie A Nardi. Gardeners and gurus: Patterns of cooperation among cad users. In *Proceedings of the SIGCHI Conference on Human factors in Computing Systems*, pages 107–117. ACM, 1992.

[27] Giuseppe Ghiani, Marco Manca, Fabio Paternò, and Carmen Santoro. Personalization of context-dependent applications through trigger-action rules. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 24(2):14, 2017.

[28] Alix Goguey, Cameron Steer, Andrés Lucero, Laurence Nigay, Deepak Ranjan Sahoo, Céline Coutrix, Anne Roudaut, Sriram Subramanian, Yutaka Tokuda, Timothy Neate, Jennifer Pearson, Simon Robinson, and Matt Jones. PickCells: A physically reconfigurable cell-composed touchscreen. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–14, New York, NY, USA, 2019. ACM.

[29] Elizabeth Goodman and Daniela Rosner. From garments to gardens: Negotiating material relationships online and 'by hand'. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2257–2266. ACM, 2011.

[30] Jeremy Gummeson, James Mccann, Chouchang (Jack) Yang, Damith Ranasinghe, Scott Hudson, and Alanson Sample. RFID Light Bulb: Enabling ubiquitous deployment of interactive RFID systems. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(2):12, 2017.

[31] Jeremy M Heiner, Scott E Hudson, and Kenichiro Tanaka. Linking and messaging from real paper in the Paper PDA. In *Proceedings of the 12th Annual ACM Symposium on User Interface Software and Technology*, UIST '99, pages 179–186. ACM, 1999.

[32] Susan Lee Hendrix. *Popup Workshop: Computationally Enhanced Paper Engineering for Children*. PhD thesis, USA, 2008.

[33] Steve Hodges, Nicolas Villar, Nicholas Chen, Tushar Chugh, Jie Qi, Diana Nowacka, and Yoshihiro Kawahara. Circuit Stickers: Peel-and-stick construction of interactive electronic prototypes. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 1743–1746. ACM, 2014.

[34] Michael S Horn, Sarah AlSulaiman, and Jaime Koh. Translating Roberto to Omar: Computational literacy, stickerbooks, and cultural forms. In *Proceedings of the 12th International Conference on Interaction Design and Children*, IDC '13, pages 120–127. ACM, 2013.

[35] Michael S Horn, Erin Treacy Solovey, and Robert JK Jacob. Tangible programming and informal science learning: Making TUIs work for museums. In *Proceedings of the 7th International Conference on Interaction Design and Children*, IDC '08, pages 194–201, 2008.

[36] Eva Hornecker and Jacob Buur. Getting a grip on Tangible Interaction: A framework on physical space and social interaction. In *Proceedings*

of the SIGCHI Conference on Human Factors in Computing Systems, CHI '06, pages 437–446. ACM, 2006.

[37] Meng-Ju Hsieh, Jr-Ling Guo, Chin-Yuan Lu, Han-Wei Hsieh, Rong-Hao Liang, and Bing-Yu Chen. RFTouchPads: Batteryless and wireless modular touch sensor pads based on RFID. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, pages 999–1011. ACM, 2019.

[38] Justin Huang and Maya Cakmak. Supporting mental model accuracy in trigger-action programming. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, PERVASIVE '15, pages 215–225. ACM, 2015.

[39] Jan Humble, Andy Crabtree, Terry Hemmings, Karl-Petter Åkesson, Boriana Koleva, Tom Rodden, and Pär Hansson. Playing with the bits: User-configuration of ubiquitous domestic environments. In *International Conference on Ubiquitous Computing*, pages 256–263. Springer, 2003.

[40] Alexandra Ion, Johannes Frohnhofen, Ludwig Wall, Robert Kovacs, Mirela Alistar, Jack Lindsay, Pedro Lopes, Hsiang-Ting Chen, and Patrick Baudisch. Metamaterial mechanisms. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 529–539. ACM, 2016.

[41] Hiroshi Ishii and Brygg Ulmer. Tangible bits: Towards seamless interfaces between people, bits, and atoms. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '97, pages 234–241, New York, New York, 1997. ACM.

[42] Rikke Hagensby Jensen, Yolande Strengers, Jesper Kjeldskov, Larissa Nicholls, and Mikael B Skov. Designing the desirable smart home: A study of household experiences and energy consumption impacts. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 4. ACM, 2018.

[43] Haojian Jin, Jingxian Wang, Zhijian Yang, Swarun Kumar, and Jason Hong. WiSh: Towards a wireless shape-aware world using passive RFIDs. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 428–441, 2018.

[44] Mustafa Emre Karagozler, Ivan Poupyrev, Gary K Fedder, and Yuri Suzuki. Paper Generators: Harvesting energy from touching, rubbing and sliding. In *Proceedings of the Annual ACM symposium on User Interface Software and Technology*, UIST '13, pages 23–30, 2013.

[45] Keiko Katsuragawa, Ju Wang, Ziyang Shan, Ningshan Ouyang, Omid Abari, and Daniel Vogel. Tip-Tap: Battery-free discrete 2D fingertip input. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '19, pages 1045–1057. ACM, 2019.

[46] Fahim Kawsar, Tatsuo Nakajima, and Kaori Fujinami. Deploy spontaneously: Supporting end-users in building and enhancing a smart home. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, Ubicomp '08, pages 282–291. ACM, 2008.

[47] Cayla Key, Fiona Browne, Nick Taylor, and Jon Rogers. Proceed with care: Reimagining home IoT through a care perspective. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, pages 1–15. ACM, 2021.

[48] Scott R Klemmer, Jamey Graham, Gregory J Wolff, and James A Landay. Books with voices: Paper transcripts as a physical interface to oral histories. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, (CHI '03), pages 89–96. ACM, 2003.

[49] Scott R Klemmer, Jack Li, James Lin, and James A Landay. Papier-Mâché: Toolkit support for tangible input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 399–406. ACM, 2004.

[50] Vijay Kumar, Ludovic Koehl, and Xianyi Zeng. A fully yarn integrated tag for tracking the international textile supply chain. *Journal of Manufacturing Systems*, 40:76–86, 2016.

[51] David Kurlander, Allen Cypher, and Daniel Conrad Halbert. *Watch what I do: Programming by demonstration*. MIT press, 1993.

[52] Amanda Lazar, Caroline Edasis, and Anne Marie Piper. Supporting people with dementia in digital social sharing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 2149–2162. ACM, 2017.

[53] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. Evaluation strategies for HCI toolkit research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–17. ACM, 2018.

[54] Johnny C Lee, Scott E Hudson, and Edward Tse. Foldable interactive displays. In *Proceedings of the 21st annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 287–290, 2008.

[55] Matthew L Lee and Anind K Dey. Sensor-based observations of daily living for aging in place. *Personal and Ubiquitous Computing*, 19(1):27–43, 2015.

[56] Hanchuan Li, Eric Brockmeyer, Elizabeth Carter, Josh Fromm, Scott Hudson, Shwetak Patel, and Alanson Sample. PaperID: A technique for drawing functional battery-free wireless interfaces on paper. In *Proceeding of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '16, pages 5885–5896. ACM, 2016.

[57] Hanchuan Li, Can Ye, and Alanson P Sample. IDSense: A human object interaction detection system based on passive UHF RFID. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 2555–2564. ACM, 2015.

[58] Rong-Hao Liang, Meng-Ju Hsieh, Jheng-You Ke, Jr-Ling Guo, and Bing-Yu Chen. RFIMatch: Distributed batteryless near-field identification using RFID-tagged magnet-biased reed switches. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, UIST '18, pages 473–483. ACM, 2018.

[59] Giorgia Lupi and Stefanie Posavec. *Dear data*. Chronicle books, 2016.

[60] Wendy E Mackay, Guillaume Pothier, Catherine Letondal, Kaare Bøegh, and Hans Erik Sørensen. The missing link: Augmenting biology laboratory notebooks. In *Proceedings of the 15th annual ACM symposium on User interface software and technology*, UIST '02, pages 41–50. ACM, 2002.

[61] Allan MacLean, Kathleen Carter, Lennart Lövstrand, and Thomas Moran. User-tailorable systems: Pressing the issues with buttons. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, pages 175–182. ACM, 1990.

[62] Jennifer Mankoff, Scott E Hudson, and Gregory D Abowd. Providing integrated toolkit-level support for ambiguity in recognition-based interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '00, pages 368–375. ACM, 2000.

[63] Nicolai Marquardt, Steven Houben, Michel Beaudouin-Lafon, and Andrew D. Wilson. HCITools: Strategies and best practices for designing, evaluating and sharing technical HCI toolkits. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '17, page 624–627, New York, NY, USA, 2017. Association for Computing Machinery.

[64] David A Mellis, Sam Jacoby, Leah Buechley, Hannah Perner-Wilson, and Jie Qi. Microcontrollers as material: Crafting circuits with paper, conductive ink, electronic components, and an "untoolkit". In *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction*, TEI '13, pages 83–90. ACM, 2013.

[65] Sarah Mennicken and Elaine M. Huang. Hacking the natural habitat: An in-the-wild study of smart homes, their development, and the people who live in them. In *Proceedings of the 10th international conference on Pervasive Computing*, PERVASIVE '12, pages 143–160, Berlin, Germany, 2012. Springer.

[66] Brad A Myers. A new model for handling input. *ACM Transactions on Information Systems (TOIS)*, 8(3):289–320, 1990.

[67] Bonnie A. Nardi. *A small matter of programming: perspectives on end user computing*. MIT press, 1993.

[68] William M Newman. A system for interactive graphical programming. In *Proceedings of the April 30–May 2, 1968, Spring Joint Computer Conference*, pages 47–54, 1968.

[69] Hyunjoo Oh, Jeeeun Kim, Cory Morales, Mark Gross, Michael Eisenberg, and Sherry Hsi. FoldMecha: Exploratory design and engineering of mechanical papercraft. In *Proceedings of the Eleventh International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '17, pages 131–139. ACM, 2017.

[70] Simon Olberding, Sergio Soto Ortega, Klaus Hildebrandt, and Jürgen Steimle. Foldio: Digital fabrication of interactive and shape-changing objects with foldable printed electronics. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, UIST '15, pages 223–232. ACM, 2015.

[71] Dan R. Olsen. Evaluating user interface systems research. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, page 251–258, New York, NY, USA, 2007. Association for Computing Machinery.

[72] Dan R Olsen Jr. MIKE: The menu interaction kontrol environment. *ACM Transactions on Graphics (TOG)*, 5(4):318–344, 1986.

[73] John F Pane and Brad A Myers. Tabular and textual methods for selecting objects from a group. In *Proceeding 2000 IEEE International Symposium on Visual Languages*, pages 157–164. IEEE, 2000.

[74] Huaishu Peng, Jennifer Mankoff, Scott E. Hudson, and James McCann. A layered fabric 3D printer for soft interactive objects. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, page 1789–1798. ACM, 2015.

[75] Richard L. Potter. *Pixel data access: Interprocess communication in the user interface for end-user programming and graphical macros*. PhD thesis, 1999. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2023-02-24.

[76] Richard L Potter, Linda J Weldon, and Ben Shneiderman. Improving the accuracy of touch screens: an experimental evaluation of three strategies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 27–32, 1988.

[77] Jie Qi and Leah Buechley. Electronic popables: Exploring paper-based computing through an interactive pop-up book. In *Proceedings of the Fourth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '10, pages 121–128. ACM, 2010.

[78] Jie Qi and Leah Buechley. Sketching in circuits: Designing and building electronics on paper. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 1713–1722. ACM, 2014.

[79] Jie Qi, Andrew " Bunnie" Huang, and Joseph Paradiso. Crafting technology with circuit stickers. In *Proceedings of the 14th International Conference on Interaction Design and Children*, IDC '15, pages 438–441. ACM, 2015.

[80] Alexander Repenning. Moving beyond syntax: Lessons from 20 years of blocks programing in AgentSheets. *Journal of Visual Languages and Sentient Systems*, 3(1):68–91, 2017.

[81] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, and Yasmin Kafai. Scratch: Programming for all. *Comm. of the ACM*, 52(11):60–67, 2009.

[82] Jennifer A Rode, Eleanor F Toye, and Alan F Blackwell. The fuzzy felt ethnography—Understanding the programming patterns of domestic appliances. *Personal and Ubiquit. Computing*, 8(3-4):161–176, 2004.

[83] Michael Rohs. Visual code widgets for marker-based interaction. In *25th IEEE International Conference on Distributed Computing Systems Workshops*, pages 506–513. IEEE, 2005.

[84] Kimiko Ryokai, Stefan Marti, and Hiroshi Ishii. I/O Brush: Drawing with everyday objects as ink. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 303–310, 2004.

[85] Antti Salovaara, Andrea Bellucci, Andrea Vianello, and Giulio Jacucci. Programmable smart home toolkits should better address households' social needs. In *Proceeding of the SIGCHI Conference on Human Factors in Computing Systems*, New York, NY, 2021. ACM.

[86] Julia Schwarz. *Monte Carlo Methods for Managing Uncertain User Interfaces*. PhD thesis, Carnegie Mellon University, 2014.

[87] Andrew Speilberg, Alanson Sample, Scott Hudson, Jennifer Mankoff, and James McCann. RapID: A framework for fabricating low-latency interactive objects with RFID tags. In *Proceeding of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '16, pages 5897–5908, New York, NY, 2016. ACM.

[88] Lisa Stifelman, Barry Arons, and Chris Schmandt. The Audio Notebook: Paper and pen interaction with structured speech. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '01, pages 182–189, 2001.

[89] Lisa J Stifelman. Augmenting real-world objects: A paper-based audio notebook. In *Conference Companion on Human Factors in Computing Systems*, pages 199–200, 1996.

[90] Laurel Swan, Alex S Taylor, and Richard Harper. Making place for clutter and other ideas of home. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 15(2):1–24, 2008.

[91] Cristina Sylla, Pedro Branco, Sérgio Gonçalves, Clara Coutinho, and Paulo Brito. T-books: Merging traditional storybooks with electronics. In *Proceedings of the 11th International Conference on Interaction Design and Children*, IDC '12, pages 323–326. ACM, 2012.

[92] Alex S Taylor, Richard Harper, Laurel Swan, Shahram Izadi, Abigail Sellen, and Mark Perry. Homes that make us smart. *Personal and Ubiquitous Computing*, 11(5):383–393, 2007.

[93] Alex S Taylor, Susan P Wyche, and Joseph ' Jofish' Kaye. Pottering by design. In *Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges*, pages 363–372, 2008.

[94] Jakob Tholander and Maria Normark. Crafting personal information-resistance, imperfection, and self-creation in bullet journaling. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, pages 1–13. ACM, 2020.

[95] Khai N Truong, Elaine M Huang, and Gregory D Abowd. CAMP: A magnetic poetry interface for end-user programming of capture applications for the home. In *International Conference on Ubiquitous Computing*, pages 143–160. Springer, 2004.

[96] Blase Ur, Melwyn Pak Yong Ho, Stephen Brawner, Jiyun Lee, Sarah Mennicken, Noah Picard, Diane Schulze, and Michael L Littman. Trigger-Action programming in the wild: An analysis of 200,000 IFTTT recipes. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 3227–3231, 2016.

[97] Guanyun Wang, Tingyu Cheng, Youngwook Do, Humphrey Yang, Ye Tao, Jianzhe Gu, Byoungkwon An, and Lining Yao. Printed paper actuator: A low-cost reversible actuation and sensing method for shape changing interfaces. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2018.

[98] Colin Watson, Reuben Kirkham, and Ahmed Kharrufa. PIP Kit: An exploratory investigation into using lifelogging to support disability benefit claimants. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2020.

[99] Mark Weiser. The computer of the 21st century. *Mobile Computing and Communications Review*, 3(3), 1991.

[100] Kristin Williams, Rajitha Pulivarthy, Scott E Hudson, and Jessica Hammer. Understanding family collaboration around lightweight modification of everyday objects in the home. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–24, 2019.

[101] Kristin Williams, Rajitha Pulivarthy, Scott E Hudson, and Jessica Hammer. The upcycled home: Removing barriers to lightweight modification of the home's everyday objects. In *Proceedings of the CHI Conference on Human Factors in Comp. Sys.*, pages 1–13, 2020.

[102] Aaron Wilson, Margaret Burnett, Laura Beckwith, Orion Granatir, Ledah Casburn, Curtis Cook, Mike Durham, and Gregg Rothermel. Harnessing curiosity to increase correctness in end-user programming. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 305–312, 2003.

[103] Charlie Wilson and Tom Hargreaves. Smart homes and their users: a systematic analysis and key challenges. *Personal and Ubiquitous Computing*, 19(2), 2015.

[104] Brian G Winder, Spencer P Magleby, and Larry L Howell. Kinematic representations of pop-up paper mechanisms. *Journal of mechanisms and robotics*, 1(2), 2009.

[105] JongBum Woo and Youn-kyung Lim. User experiences in do-it-yourself-style smart homes. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 779–790, New York, NY, 2015. ACM.

[106] Jianxin Wu, Adebola Osuntogun, Tanzeem Choudhury, Matthai Philipose, and James M Rehg. A scalable approach to activity recognition based on object use. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

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

[108] Clement Zheng, Peter Gyory, and Ellen Yi-Luen Do. Tangible interfaces with printed paper markers. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference*, pages 909–923, 2020.

[109] Kening Zhu and Shengdong Zhao. Autogami: a low-cost rapid prototyping toolkit for automated movable paper craft. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 661–670, 2013.