Working With Useless Machines

A look at our shifting relationship with ubiquity through personal assistants

by: Nadine Lessio

A thesis exhibition presented to OCAD University in partial fulfillment of the requirements for the degree of Master of Design in Digital Futures

49 McCaul Street, April 12 - 15
Toronto, Ontario, Canada, April, 2018

Copyright Notice

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize OCAD University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

I further authorize OCAD University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature _____

Abstract

Working With Useless Machines

Master of Design 2018
Nadine Lessio
Digital Futures
OCAD University

This research explores how a nihilistic approach to the development of personal assistants can problematize the ways these technologies are being brought to market by using the concept of the useless machine. The research is organized around the creation of four prototypes: *SAD Blender, Home Hub, Calendar Creep, Fortune Tasker,* and a personal web server named *Punchy*. Each of these prototypes seek to critique the current corporate agenda of productivity, efficiency, and consumption, by creating machines that have no utilitarian function. This work investigates and defines useless machines in the contemporary context of personal assistants, and documents the development of each of the prototypes and server. Overall the research shows that while its not possible to completely divorce these devices from their parent corporations, they do serve as an important vehicle to explore the changing landscape of ubiquity, and our shifting ability to work and live with these connected objects.

Acknowledgments

Most of all I want to thank my family and friends for their ongoing love and support.

I would also like to thank my advisors Nick Puckett and Emma Westecott for their guidance and mentorship. As well as my colleagues Rob Russell for their contributions on *Punchy*, and Alanna Predko for their contributions to *Fortune Tasker*.

This work was funded by a CGS-M Joseph-Armand Bombardier Canada Graduate Scholarship

Table Of Contents

List of Figures

Artists critique art through art. Musicians satirize other musical forms with music, As people who make tech, we critique it through the medium that is our language.

– Winger-Bearskin

Chapter One

Introduction

1.1 Project Overview

The purpose of this research project has been to explore the idea of useless machines through the creation of software that runs on current personal assistant platforms like Google Home and Amazon Alexa. From the wider values of nihilism my research grew from an ongoing frustration with the gaps between the marketed ideal of personal assistants and the current reality of this sector. How might a nihilistic approach to the development of personal assistants problematize the ways these technologies are being brought to market?

Throughout this project, I have created four prototypes using these devices to critique the current state of personal assistants, and the corporate rhetoric of convenience behind the internet of things in general. The prototypes are called: *SAD Blender, Home Hub, Calendar Creep, and Fortune Tasker* I have also created a personal web server affectionately nicknamed *Punchy* that runs on a local wireless network that is used as infrastructure to help demonstrate these devices. These are not product based builds, they are explorations around a platform, using a basic guiding idea. The prototypes are:

SAD Blender

An Alexa that suffers from seasonal affective disorder is hooked up to a blender and set of small lights. Its condition affects whether it will operate the blender or not.

Home Hub

A Google Home initiates a program in which you can control items in your home, but it finds a bug it must contend with. The Google Home decides that it is more important to debug itself than to help you with your home automation tasks.

Calendar Creep

An Alexa runs a custom program to create calendar events. Except that every time you try to make an event it tries to convince you not to go out.

Fortune Tasker

A disassembled Alexa that can not speak is tasked with giving you absurd fortunes through the use of a printer and a reclaimed deer skull.

Punchy (web server + wireless network)

Punchy takes the place of an external cloud based server, to try and interact with some IoT devices at the local area network level, while the wireless network was a shot at cutting down on the administrative overhead in a managed network.

1.2 What Is A Useless Machine?

1.2.1 Definition

A useless machine is a strategy employed to critique the current corporate agenda of productivity, efficiency, and consumption, by creating machines that have no utilitarian function. useless machines are purposefully executed to provoke a reflection on the current state of the status quo. They are made using a variety of techniques, but tend to reflect the dominant technical or consumer culture of the time. In this case a machine can refer to a technical gadget, a product, software, or device.

1.2.2 Influences

I would say that my interest in useless machines grew out of really enjoying how Douglas Adams depicts technology in his writing. I feel that there are a lot of parallels between the Sirius Cybernetics Corporation and modern day tech corporations. They're both churning out smart products that are supposed to make your life easier, but kind of don't. The most famous of these is the Nutri-Matic Drink Dispenser, which in the words of Douglas Adams:

> *"...claimed to produce the widest possible range of drinks personally matched to the tastes and metabolism of whoever cared to use it. When put to the test, however, it invariably produced a plastic cup filled with a liquid that was almost, but not quite, entirely unlike tea." (Adams 198)*

The Drink Dispenser is incredibly bad at what it does, so bad that at one point in *The Restaurant and The End of The Universe* it conscripts the ship's main computer to help it make tea. This causes a modern day DDoS attack on the entire ship, which almost gets destroyed because all the logic circuits are busy trying to solve the issue of "tea".

The reason I find this so relevant is because very recently there's been a series of smart devices implicated in DDoS attacks. From smart vending machines and lightbulbs that took

down a university network (Mathews). To Mirari, which is a large Internet of Things based Botnet that took down most of the eastern USA's internet (Newman). It is a weird case of life imitating art. But it is not totally unexpected.

In *Everything Is Broken*, Quinn Norton notes that "Its hard to explain to regular people how much technology barely works, how much the infrastructure of our lives is held together by the IT equivalent of baling wire" (Norton). Computers have reached a point where they are now so complex, that not one person, or even groups of people, can know what is happening at any given point in the system. Now think of a system of millions of little boxes, within boxes, constantly trying to talk and coordinate tasks at the same time (Norton), and you might start to understand how our devices, to turn a phrase, might not have our privacy, security, or best interests in mind.

This idea that devices might not be here to serve us, or that they might have internal motivations independent of our interests, is something I tie into the writing of Eugene Thacker whose book, *In The Dust of This Planet*, looks at nihilism through a horror / sci-fi genre lens. Coming from a background of existential nihilism, Thacker is working within a framework that sees the universe as completely impersonal and indifferent to humans (Thacker 10), and that the horror aspect is when humans confront this idea that maybe they aren't very central to the universe. This is the kind of indifference usually referred to in Lovecraftian Horror, used frequently in hollywood science fiction films, and referenced in popular TV shows like *Rick and Morty.*

Thacker also spends a lot of time establishing the idea of the world-without-us (Thacker 5), which is a hard thought to encapsulate, but which he describes as a speculative world that points to the fuzzy domain of the non-human (Thacker 6). The world-without-us exists in the cracks and shadows of the world for-us (the world we know and are familiar with). We can't really fully see it because it can't exist with the world-for-us, but technology can be a way to play between these two worlds. When technology fails, when it doesn't do what you expect it

to do, or when it displays some aspect of autonomy that doesn't necessarily centralize you, there's a little brush up to this world. It functions like a crack in a shiny veneer of technology that we are sold, that exposes some of its operational indifference and futility.

1.3 Why Focus On Personal Assistants?

1.3.1 Technological Hype Cycle

Personal assistants are very new. According to the *2017 Gartner Hype Cycle for Emerging Technologies*, personal assistants (also known as virtual assistants) are shown rising on the heels of machine learning and a fast growing interest in Artificial Intelligence (Panetta) but are still looking at a good 5-10 years before they reach their plateau.

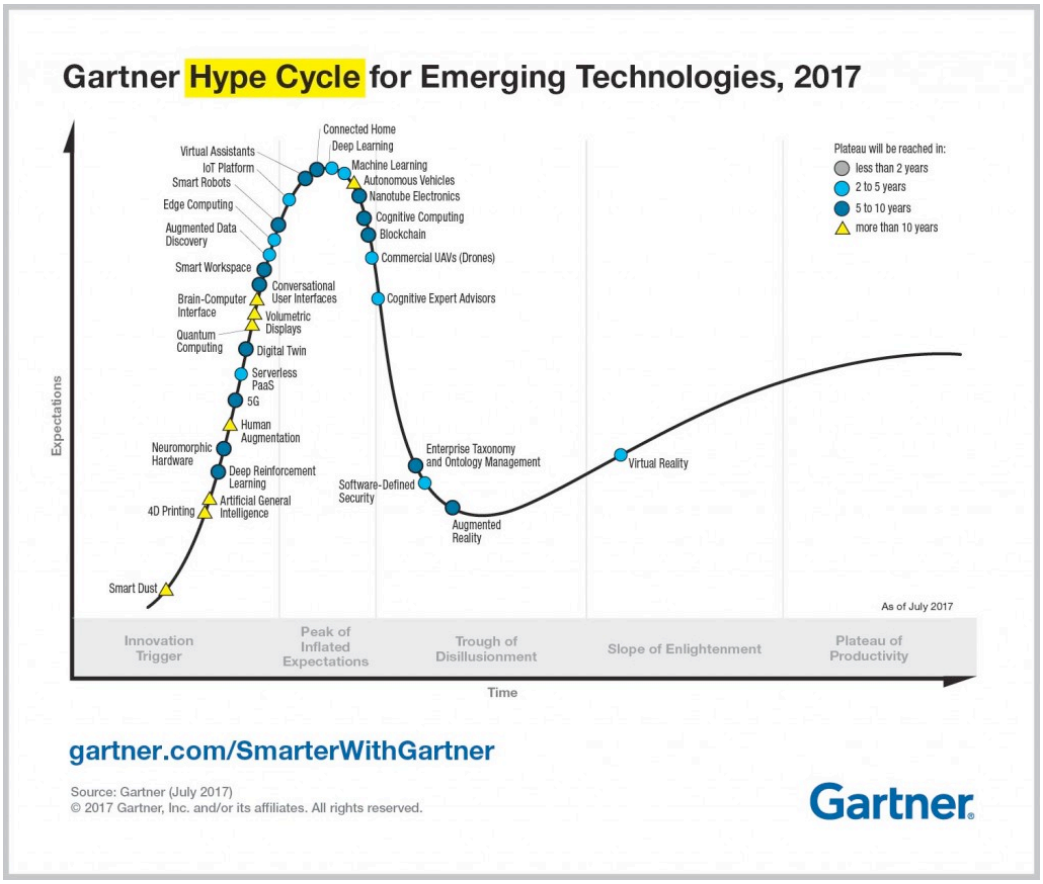


Figure 1.1: Gartner Hype Cycle

These technologies are also just starting to come into their peak of inflated expectations cycle, which means adoption is starting to get high, but the success stories are accompanied by a score of failures (Panetta). Even though software assistants have been around for a while on phones, the current trend of making consumer smart speaker devices shows that there is a lot of ambiguity as to what these devices actually do. This ambiguity and play between successes and failures means that personal assistants are still a novelty which provides a good opportunity to play with their behaviors, and explore alternate ideas around their intended functionality. Can it be a friend? Can it be a nemesis? Can it be indifferent? Sure. Why not. It doesn't really know what it is right now anyway.

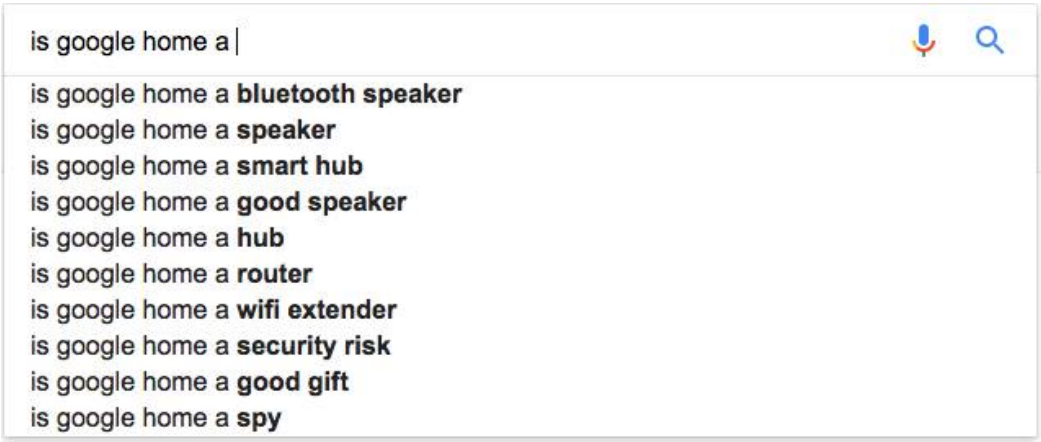


Figure 1.2: Google search for google home

Because their infrastructure is constantly changing, we are provided with a real time look at the challenges found in development, adoption, and the futility faced with trying to manage everything when it is still in flux. Large corporations are also trying to make a grab for dominance by pushing infrastructure management up into the application layer in the form of services. For example, Amazon and Google are both pushing their voice assistants into the realm of being an operating system (Seifert) by embedding their services into third party products and offering a suite of services to developers and partners.

This newness means they are still incredibly awkward to deal with and they currently sit on that cusp between novelty and product. This makes personal assistants an interesting candidate to attempt making useless machines out of, and with some gentle nudging you can start to expose some of that to the speculative indifference that Thacker is referring to.

1.3.2 Internet Nihilism

> *At a certain moment, a culture discovers that its most esteemed values are for nothing. nihilism is that moment where you have the rug pulled out from under you, and nothing takes its place.– Eugene Thacker*

In 2011 Eugene Thacker wrote an academic treatise about nihilism called *In The Dust of This Planet,* and for a while no one read it (Abumrad). But then it showed up on the show *True Detective*, then later in a fashion magazine, and then in 2014 it ended up on the back of Jay-Z's jacket in a trailer called *Run*. This was pretty surprising for

Figure 1.3: Frame From Run, Jay-Z & Beyonce. 2014

Thacker, and in response, NPR's *Radiolab* did an episode exploring how that happened. During that episode the host and guests debated if Thacker was tapping into a current shared feeling in America, and the consensus was yes, American pop culture was expressing a sense of nihilism.

Pop culture has always been deeply nihilistic (Abumrad). According to Brooke Gladstone from *On The Media*, there are cycles in which this sense of meaninglessness comes out in sharper relief than other times, and you can identify them over and over again (Gladstone). Gladstone traces this back through the nineties (Agent Smith from the Matrix), the eighties (The Watchmen), the seventies (Punk), the fifties and sixties (Beatniks), all the way back to Dada at the end of World War One. Brooke cites Dada as her favourite expression of nihilism because it is an active and positive embrace of what seems to be the only kind of truth, which is that there isn't one (Brooke).

Figure 1.4: Existential Dread Twitter

Today we find ourselves living under the constant specter of climate change, impossible debt, Trump, the threat of being replaced by AI, and all delivered to us in real time. One of the interesting things about today's expressions of nihilism is that unlike Dadaism that had a world war to blame, or the fifties that had The Cold War, or even the nineties, that had The Wage Salve, now the situation feels more complicated, because we're all active participants in it. In our anxious state of precarity, we lean into nihilism by making fun of our own collective

condition. This is part of our collective coping strategy for how messed up things have become (Institute for Precarious Consciousness).

One place you can trace this through is Twitter. When people talk about Twitter, sometimes they refer to things like Nerd Twitter or Black Twitter, which are Twitter accounts that exist in their own little worlds (Squires). There is a grouping of Twitter accounts called Existential Dread Twitter, which are mostly bots that play off the current human condition. They exist in a dark realm beyond doggos and self-care, but nonetheless are still amusing and strangely cathartic (Squires). This nihilistic flavour of satire also crosses over into technology with twitter accounts like *Internet of Shit* (internetofshit), which primarily makes fun of the state of smart devices. It can also be seen in sites like *whatthefuckismywearablestrategy.com* (O'Connell) that exist to poke fun at the current state of VC funding and tech innovation.

Figure 1.6: Internet Of Shit, twitter.com/internetofshit

This is where my exploration of personal assistants sits. I question their overall value in our current technological landscape, and I'm doing this by making them do unexpected things because I personally feel a growing sense of tiredness around technological promises. Nihilism is, at its core, a rejection, and today's expression of nihilism is to make things that make fun of themselves.

Chapter Two

Current Landscape

2.1 Background

A lot has happened since Bruce Sterling released *Shaping Things* in 2005. The Internet of Things has grown from an industrially-focused supply chain solution (Sterling 13) into a multi-branched industry covering consumer products, industrial products, DIY, services, and infrastructure. According to infrastructure giant Cisco Systems there are about 12.1 billion internet connected devices in 2014, and that by 2020 we could be looking at 50 billion (Greengard 13). But we are still in what Sterling would coin a GIZMO society. Our connected devices are just smart enough to nag us (Sterling 12), but not rich or implemented enough to be what Sterling considers SPIMES. SPIMES, are objects that act like an instantiation of all the data and supporting infrastructure behind them (Sterling 14). They are manufactured objects that can be tracked, wrangled, and reclaimed back into their own data stream. They know everything about themselves through their relationship to you and your consumption pattern (Sterling 14). Sterling notes that while true SPIMES are still a ways out, our phones can be considered proto-SPIMES. Regardless of whether something is a SPIME or a GIZMO, it still needs an interface. IoT has been a highly technologically driven field to date, and much of the focus has been around solving connectivity challenges. But as the technology matures we need to ask ourselves what we want to make from it and how we should make it (Rowland). But a lot of our current design methods aren't well suited to the complexity of interconnected systems and they're going to need to evolve to respond to this evolution (Rowland). Voice User Interface is part of that evolution.

Speech recognition as an interface was around in science fiction long before it came to exist in real life (Pearl xi), one of the most famous being Hal 9000 from *2001 A Space Odyssey*. But it has also made an appearance in other things - from *Star Trek*, to *Resident Evil*, to countless TV shows and books (Pearl xi). Even though speech recognition had been in development since the 1950s, the first great era of VUIs were interactive voice response systems that became mainstream in the early 2000s (Pearl xi), which is also the time period that mobile phones began to see a larger market penetration in North America. We're now in the second phase of VUIs as personal assistants start to become mainstream. Google reports that 20 percent of its searches are now done via voice (Helft), and a recent study released by NPR has shown that 39 million people in the US now own either a google home or amazon Alexa smart speaker (Smartaudio Report).

2.2 Consumer Landscape

Personal Assistants are being positioned by their parent corporations as devices that save time by utilizing consumer data to personalize services and remove pain points from people's daily activities (Rowland 548). They aren't truly smart, though they are positioned as such (Seifert). Instead, PAs act more like remote controls, and are just an interface to a collection of discrete and effectively independent underlying functionalities (Dale). For example, if you ask a Google Home to tell you the weather, it will hook into a weather API and deliver that data back to you as a voice response, theres's no heavy processing at the device level. Over the past few years, Amazon and Google opened its platform to independent developers to build applications to fill some of the functionality and content gaps in their system. At home, the majority of users tend to use their Personal Assistant Device to play music, answer a random question, or get the weather (NPR). They generally keep them in the living room, or a shared family space, and around 30% use them to control household devices (NPR). Companies position these devices as being useful, but there's the question of useful to whom?

Currently machines are still very bad at inferring and using context appropriately, and humans are notoriously bad at predicting their own future needs (Rowland 625). The level of intelligence required to behave in 'common sense' ways that are almost second nature to people, is prohibitively time consuming and expensive to develop (Rowland 648).

So this leaves personal assistants in a strange place when it comes to the consumer landscape. The platforms have gotten better, the ecosystems personal assistants can access have become much larger, and the adoption rate is getting higher, but in a consumer setting some of the more advanced ideas about autonomous interconnected systems are overkill for the amount of personal value we get out of them (Rowland 649).

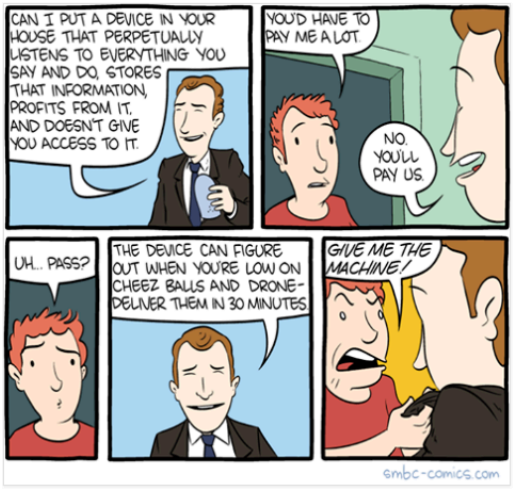


Figure 2.1: SMBC Comic

Corporations meanwhile are gaining a lot. Not only is your voice data saved and stored, but that data is aggregated, and studied. Alexa engineers scan it for the most common utterances from users (Anders), and with Alexa's usage surging, Amazon now has access to an expansive repository of human-computer speech interactions (Anders). While this data collection is vital to improve personal assistant platforms, the most ambitious developers are companies making hardware or selling services that work with platforms like Google Home or Alexa (Anders). The ultimate payoff is the opportunity to control, or at least influence three important markets: home automation, entertainment, and shopping (Anders). This year's Consumer Electronics Show featured a 3 story installation in the parking lot by Google dedicated to showing off the Google Home meanwhile Amazon spent its time showcasing which third party devices would ship with Alexa embedded in them (Olson, 2018). Industry watchers are struggling to keep up with all the different devices that Amazon has partnered with on Alexa, but it's clear the company has been busy making friends with

manufacturers over the past year (Olson). It is easy to imagine what kind of revenue sharing deals these companies could have with one another down the road, seeing as even the smallest of the three markets theses giants are vying for, is worth $5 billion annually (Anders).

It is not all nefarious though. As companies open up their APIs, access to voice interface becomes more common, and machine learning enabled services become more accessible, there has been an uptick in users creating their own macros, triggers, and programs for these devices. Though a lot of these services are proprietary, they can help to ease people into managing the ongoing complexity and changing landscape of what they consider to be 'home'.

2.2.1 The Changing Home

Homes are complicated places. They are a mix of people, architecture, history, memories, technology, and life. (Rebaudengo). Homes are also messy, they have history and context that go beyond just being functional spaces for existing in. They can mean different things to different people, and the activities in them reflect the complicated nature of how people relate to 'home' (Rebaudengo). At the end of the day, a home is not just a house, it is a touch point for family, culture, and life. But the home is also a consumer entry point into new economic territories and infrastructures (McGuirk). From the introduction of electricity and water, to the parade of new appliances through the forties and fifties, the future home has always been a theater for companies to envision and explain how great technological leaps will be brought to the masses, changing their lives in myriad beneficial ways (Rebaudengo). Except that now internet is the new electricity, and the focus is on how that is going to fulfill the corporate promise of your home being a simpler, better place to live. While this is a tempting prospect, it might be that the biggest issue with smart homes, lies exactly in that assumption. (Rebaudengo). Under the guise of efficiency and convenience, private homes become not exactly public, but exposed to other private, corporate entities (McGuirk). While the totalitarian idea of control that is depicted in popular culture is somewhat far fetched, the

more realistic probability is that control will be negotiated into patterns of 'better' behaviour through financial imperatives (McGuirk). For example if your smart treadmill doesn't clock a certain number of miles a day, your insurance premiums will go up (McGuirk). Alternately its a short jump from a thermostat that knows how to save energy to one that proposes that, in fact, you have used enough energy for one day and it is time for bed (McGuirk). This kind of situation where devices have the ability to 'think' for themselves is quite new, and it brings up some thoughts about negotiation of control.

After visiting an experimental smart home project in Turin called Casa Jasmina[1], Michelle Thorne of Mozilla's Open IoT Lab[2], and Peter Bihr, who co-founded *The Good Home Project* began an open research project called *The Connected Home*[3]. One of the questions they've been raising is how we can mediate and understand potential conflicts that algorithmic agents can cause to arise between both humans and machines, and machines themselves (Thorne). For example, if you aren't home, who controls whether the blinds are up or down? The Nest thermostat might want them closed to conserve energy, but the dishwasher wants some solar power and to keep them open, who moderates this conflict if you're out? (Thorne, Henrich 11:26). One thought to help mitigate this, is to develop software agents that advocate for the human user when not around. Currently this is partially accomplished by consumer devices such as SmartThings[4], or DIY options like HaaS[5], which are central hubs setup for devices to be managed by. Google and Amazon have also been pushing into this central hub idea, for example Amazon just released a new Alexa model with an embedded Zigbee protocol. But to understand some of the complexity involved, we're going to need to look at the current state of development and networking for these devices.

---

1 casajasmina.cc

2 wiki.mozilla.org/Open_IoT

3 theconnectedhome.org

4 smartthings.com

5 home-assistant.io

2.3 Development Landscape

> *"To put it another way, infrastructures are always already messy. The messiness that we experience in laboratory ubicomp infrastructures is not a property of prototype technologies, the bleeding edge, or pragmatic compromise; messiness is a property of infrastructure itself." - Bell*

2.3.1 Ubiquity Is Messy

Until recently the landscape of IoT was very technologically driven and saw most development focused on infrastructure, hardware, and solving connectivity issues (Rowland). This is still happening, but the focus is more on refinement to address interoperability. The other side of smart devices, which is on the rise, is considering them as products and focusing on the UX of their implementation (Rowland). Ubiquity provides us with a lot of development challenges that our current design methods aren't well suited to deal with (Rowland). Implementation can be messy, and since ubiquity is a technology of the present it is inherently one that is deployed and operated in a fragmented world (Bell, 43). When dealing with ubiquity and networks, it really comes down to trying solutions that might solve specific problems, or contextual problems. Consequently many network designs can be complicated, and employ a variety of flows and implementations. ***There is no getting around the messiness of ubiquity***, there is just embracing what parts of the mess are going to work best at the time for what is being attempted. While there is a drive in industry to continue to look for other kinds of implementation models, new implementations will continue to exist simultaneously alongside other implementations. This means that the focus is really less about changing the whole system, but finding ways to contend with interoperability, and interusability between the models that make up the system. This section provides an overview of some current network implementations that are being used on large scales, and a look at some current cloud services that run personal assistants, and where they fit in this landscape.

2.3.2 Cloud Computing and Natural Language Processing

Cloud Computing is the practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer (Oxford). This allows companies to purchase computing time in the same way you use electricity, vs building and maintaining a solution in house. Amazon Web Services, Google Cloud, and IBM Cloud, are all examples of this. When it comes to personal assistants, a lot of the Natural Language Processing (NLP) needed to run something like an Google Home is delivered as a service in a Cloud Computing Environment. For Google Home this is a combination of Google Actions, Google Assistant SDK, and Dialogflow (also known as API.ai). For Alexa the core services are Alexa Skills, and Alexa Voice Services. Some open source options for Natural Language Processing as a service include Mycroft which just launched this year, and wit.ai which was acquired by Facebook in 2015.

Natural Language Processing is the application of computational techniques to the analysis and synthesis of natural language and speech (Oxford). Its has been studied in computer science for a few decades, and you do not require the pre-mentioned services to be able to use it. Libraries such as the Natural Language Toolkit[6], Tracery[7], and spaCy[8] allow you to roll your own bots, scrape data sources, and work with language in various ways. In fact the services mentioned utilize these libraries (or libraries and algorithms like these) to run their platforms. The purpose of the services is to remove the local implementation and management of these libraries, which can be very processor intensive. It also allows for tie-ins to various machine learning services and other offered cloud services by these companies. The downside is privacy and remoteness, as these companies require you to tie things into your Amazon or Google accounts, and if the wifi goes down, you can't use the service. Both solutions are valid, it just depends what the project is and what needs to be done.

---

[6] nltk.org

[7] tracery.io

[8] spacy.io

Currently the cloud does all the resource intensive computing for Alexa and Google Home. The devices themselves are just speakers, mics, an analog to digital converter for the voice it hears, LEDs, and a series of sensors to control the buttons or touch interfaces on the devices. Personal assistants are not really ‘smart’, they are just interfaces into a suite of services that deliver content to you in an audible format. What’s interesting though is that they are a physical instantiation of a virtual assistant that can follow you around through your daily life. This could be quite useful in terms of interusability, which is the process of designing interactions that span multiple devices, but feel like a coherent service (Rowland 23 379).

2.3.3 Edge Computing and Hybrid Models

The current thought about the IoT and infrastructure, is that we are moving away from this centralized model of Cloud Computing, and heading toward Edge Computing. As more and more devices get online, infrastructure is going to have to move to a more distributed model to be able to be able to analyze the mass amounts of data edge devices will be producing (Levine). The gist of Edge Computing is that as sensors and smaller embedded devices become more sophisticated, most of the analysis and wrangling of data will happen at, or

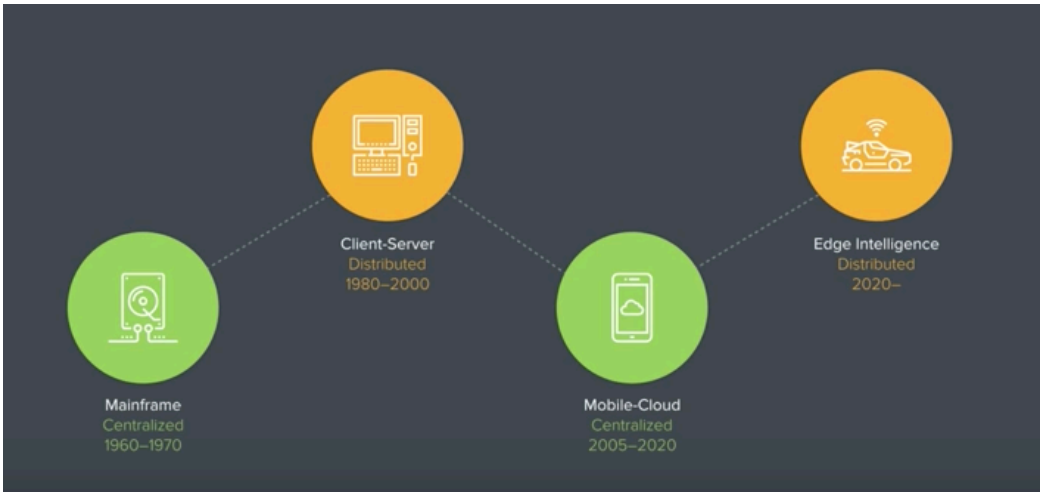


Figure 2.2: The End of Cloud Computing, Peter Levine, 2016

close to the source. For example: An automated vehicle is basically its own data processing server on wheels. Rather than sending all the raw data a car’s sensors generate from to the

cloud, embedded and local devices will analyze raw data and make real time decisions on the ground. The scrubbed data is then sent to the cloud for advanced learning and storage, where it is then re-distributed back to all the localized servers inside all autonomous cars

(Levine). The role of machine learning is two-fold in this scenario. First machine learning will have to be implemented into sensors and local data analysis devices as a way to sort through mass amounts of raw data, second machine learning in the cloud takes on a more general analysis of the curated data that's been sent to it (Levine). When this happens, things will actually be 'smart'. This is pretty much where Bell's comment of messy infrastructure comes into play. Because we're not there yet in terms of edge computing, a lot of implementation of IoT systems are very dependent on their location, context, and tend to utilize many different implementations. For example hybrid cloud is a cloud computing environment that uses a mix of on-premises, cloud and third-party public cloud services, with orchestration between the two platforms. Generally computing load is moved between private and public clouds. Its been noted that this complexity is not going to go away in the future, but instead become even denser as the demands of devices become greater.

2.3.4 Network Topologies

Currently most home based networks are a star based topology. This means that all the devices in your house, communicate and pass data through a gateway device (usually a router). This is a very standard way to setup a local area network and if you have home internet, you're probably using a star. But it does create a single point of failure. Should the router go down, then none of the devices on the network can communicate. Because of this, IoT has been very interested in Mesh Networking. A mesh network is a local area network topology based on nodes, its essentially a network of routers minus the cabling. In a mesh network, every node connects to every other node in a non-hierarchical manner. This enables devices to talk to one another regardless of if a router fails, because the rest of the network will pick up the slack, and there's no disruption in coverage like when using a repeater. The

downside is that they are much more expensive to install and harder to troubleshoot. There's been a current upshoot in interest in mesh networking amongst citizen groups, and as a result city based mesh networks have sprung up around the globe.

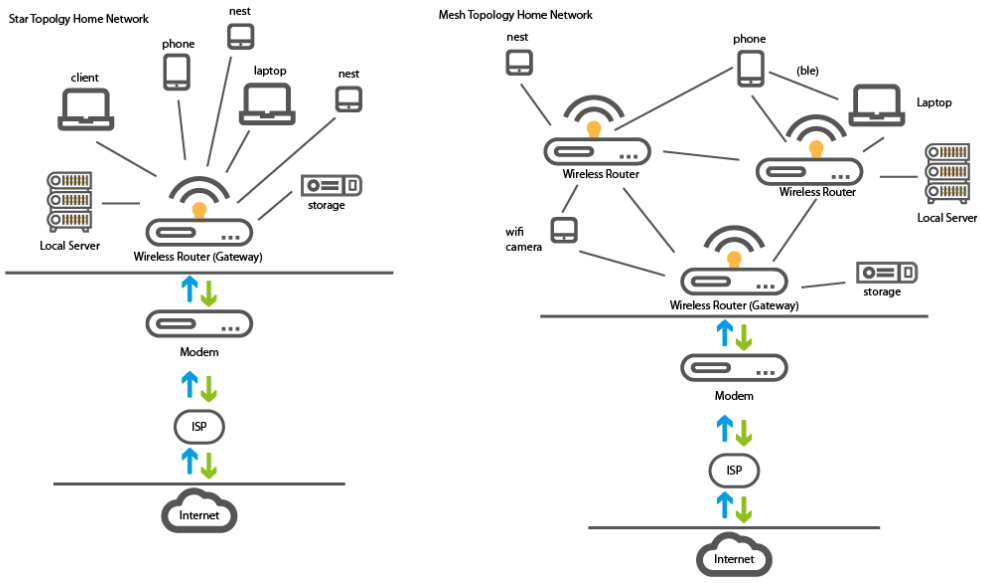


Figure 2.3: Star and Mesh Network Topologies

The downside to mesh is that currently, a lot of networks are still going to have to rely on ISPs for their wider internet connection. But some community groups like NYCMesh, and GUFI.net have managed to get a direct Internet Exchange Point and bypass ISPs all together but it is expensive to install and maintain.

2.3.5 IP To The Edge

There is also the concept of bringing IP (Internet Protocol) to the edge. Currently common edge protocols like Zigbee or Bluetooth Low Energy don't support their own routable IP addresses (hence mesh, and routers). IP to the edge is the drive to provide many more edge devices with an IP on the internet so that cloud services can contact them directly (Rowland 85). This would enable a device to identify itself uniquely on the internet (much like your modem does) no matter where it connects from. This is basically using internet protocol as a lingua franca, in which each device, no matter what they are using on the link layer, can

communicate with other devices (Rowland 86). Of course giving every connected device its own IP does open some rather large security risks, and currently address standards need to be upgraded to be able to do this. Also low powered local area networks would still need to rely on a gateway.

2.3.6 Where Personal Assistants Exist In This

Currently personal assistants are cloud computing based. When you ask Alexa or Google Home something, the request goes to the cloud, is processed, and a response is sent back to your device. This is unlikely to change in the near future as Google and Amazon have a vested interest in collecting and examining voice data, but also because their assistants are available on a wide variety of devices such as phones, and third party hardware. What this means is that something like a Google Home device does not work in a situation where an internet connection is not present. This is also the same currently for the open source assistants such as Mycroft, and Wit.

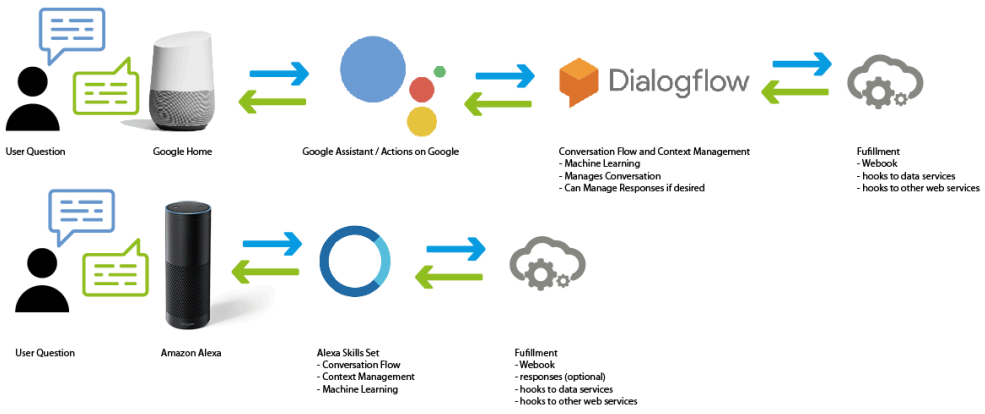


Figure 2.4: Request Flow For Assistants

DIY hardware like the respeaker by SEED, are looking to do a setup that can operate offline using a lightweight speech recognition engine called PocketSphinx. This could be highly useful for situations where someone wants direct user control of devices over a LAN, and to ease the reliance on a centralized cloud model.

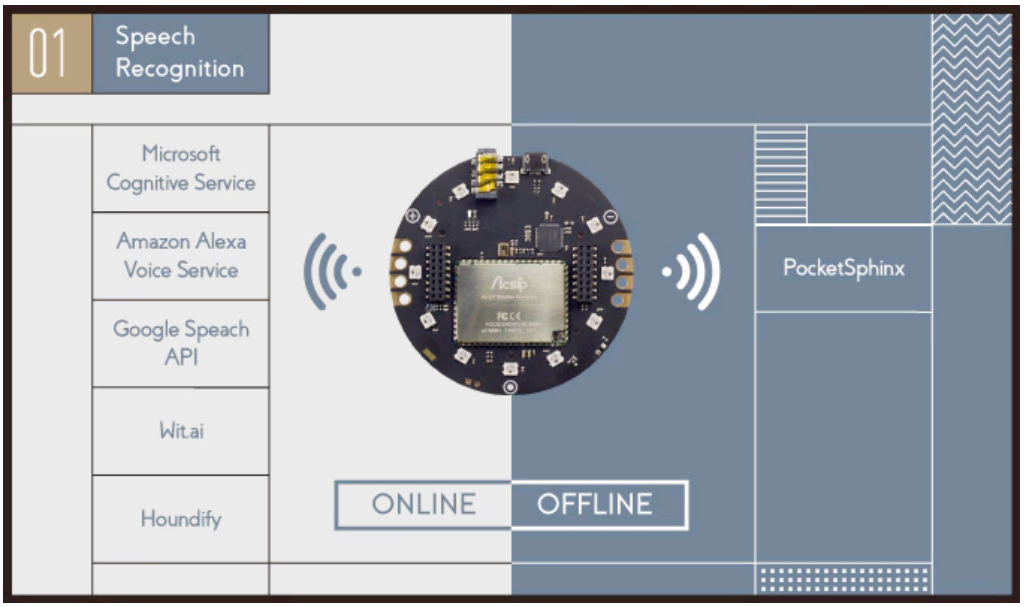


Figure 2.5: Re:Speaker Assistant Model. Seed Studio 2016

It seems likely that as things progress personal assistants like Google Home and Alexa will continue their growth inside the smart home system into more of an operating system. Amazon specifically has been pushing to have Alexa co-operate with more and more third party hardware, and come embedded in more consumer products (Seifert). One of the issues inside the smart home is that while there are a lot of companies playing with devices, its really lacking the glue that holds everything together. Assistants and voice user interface, are being pushed as that glue. Where it gets interesting is the idea that since these assistants are essentially software and available on a smart phone, they could potentially provide part of that missing interusability link between different environments that IoT is trying to solve. A ubiquitous OS for ubiquitous devices. That said, there are some major challenges to this, including noise, disruptive services, privacy concerns, and language biases. But pushing for a real life J.A.R.V.I.S[9]. isn't totally out of the question. After that though, things become a little more amorphous. With the tech pushing AI, and people adopting personal assistants, you could potentially see more developed relationships towards these devices.

[9] J.A.R.V.I.S is Tony Stark's artificially intelligent computer from the movie *The Avengers*.

Chapter Three

Contextual Review of Useless Machines

3.1 Overview

I consider myself to be a creative technologist, and creative technologists tend to operate in a grey space that crosses over between art/design/development. I don't even call what I do an art practice or a design practice, I just call it my practice. I think that technology based pieces don't just live squarely in one discipline or the other, but they cross over into each other quite a bit. The people in this contextual review have many different backgrounds: Giertz has a background in engineering physics, Munari was a commercial illustrator, and product designer, Minsky was a cognitive scientist, White is an electronics artist. The thing that ties them together for me is the contents of their output. They're using the technology of their time to critique the positioning of that technology as a utilitarian device.

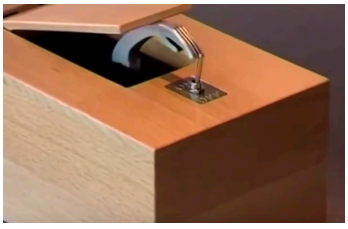
3.2 History

Useless machines as a strategy for critique has been around since the 1930s. It was first used as a term to describe the work of Bruno Munari. Munari's useless machines were small, ephemeral things that played with the properties of movement, chance, and light. They were "useless" because they, "...unlike other machines, didn't produce consumer goods, didn't make workforces obsolete, and didn't contribute to the growth of capital" (Munari 1966 15). Munari continued to explore the idea of useless machines throughout his career, from collages depicting butterfly airplanes, to stripping away the shells of mechanical toys, to make studies of only their skeletons. (*Fossils for the third Millennium 1959*).

Figure 3.1: Fossils for the Third Millennium & Useless Machines

Later the term useless machine was used to describe a consumer product based on Marvin Minsky's 1952 concept of the *Ultimate Machine*, also known as *Shannon's Hand* (Pesta). The useless machine is described as a small box with an on/off switch and a hinged lid. Turn on the switch and a lever pops out, turns off the switch, then retreats. That is the machine's sole purpose: You turn it on, and it turns itself off (Pesta). Minsky's mentor Claude Shannon at Bell Labs, built one and kept it on his desk. It has been noted that the science fiction writer Arthur C. Clarke found the device very disturbing (Pesta).

After his stint as a graduate student at Bell Labs ended, Minsky went on to other things, eventually becoming one of the founders of the field of Artificial Intelligence, and the useless



"There is something unspeakably sinister about a machine that does nothing—absolutely nothing—except switch itself off," - Arthur C. Clarke

Figure 3.2: Shannon's Hand

machine was mostly forgotten (Pesta). But recently it has been having a comeback on the internet. As machines come to control evermore aspects of our lives, one can't help but marvel at the beautiful nihilism of a machine that, far from world domination, only wants to turn itself off (Oberhaus). The useless machine is also something that makes an appearance in robotic art. Robots, as an extension of machines, are considered to be utilitarian, labour saving devices. By subverting their function you can offer up a chance to critique the role of the machine in a machine/human relationship. Norm White's *The Helpless Robot (1987)* is an early example of this. The robot itself is passive. It sits on a large lazy susan, but can only rotate with the help of humans (White). It does respond to people using an electronic voice and 512 phrases. But what it says depends on present and past experience of programmed emotions ranging from boredom, frustration, arrogance, and overstimulation (White). The results of the interaction are varied, and unexpected. A current day example of this is Simone Giertz's army of shitty robots. Which range from breakfast machines that can't serve breakfast properly, to the Automatic Butt Cleaner, which you would probably not want anywhere near your butt. These are an amusing critique on automation, which is going to cause a significant employment problem in the US over the next few decades (The Future of Work).

Figure 3.2: The Helpless Robot, Norm White, 1987

Figure 3.3: Giertz, Robots, The Breakfast Machine 2015

Useless machines can be found in installations as well. Katherine Behar's exhibit *E-Waste (2014)* features an installation of USB sculptures, inspired by a science fiction scenario in which commonplace USB devices continue working long after the humans they were designed to serve have become extinct (Behar). The devices themselves take on new forms and new modes of being without really trying to have it (Aristarkhova 56). Free of their assigned functionality, they go about their own business. Their labour for us, whether it is in holding the world, or storing our memories, is done (Aristarkhova 61). Behar's usage here is a critique on usefulness, and the idea that technology exists only as a tool. In the same vein Maurizio Bolognini's *Sealed Computers (1992)* contains a room of computers

Figure 3.4: E-Waste, Katherine Behar. 2014

processing graphics that have had all of their monitor busses sealed, so that they produce images that no one can ever see (Bolognini). Again this removes the utility function of the computer, into an indifferent state. In fact there's an entire hackathon dedicated to useless machines. It is called *The Stupid Shit No One Needs and Bad Ideas Hackathon*. It happens once a year in New York City and Toronto. The Stupid Shit Hackathon was created by Sam Lavigne and Amelia Winger-Bearskin in New York, and is a direct response about Hackathons and tech culture, that will try and 'hack' an incredibly complex problem like world hunger in one weekend (D'Anastasio). Which is not only naive but also rather pompous. It consistently churns out amusing subversions of consumer products and silicon valley ideas of the future. Everything from making AI to try and decode porn, to VR fireplaces, to programs that only make empty infinite folders, and messaging apps that just leave your friends hanging forever.

Figure 3.5: Sealed Computers, Bolognini, 1992

Figure 3.6: Personal Assistant Form Factors

When it comes to useless machines that involve personal assistants there appears to be two general tactics. The first deals with subverting the form of the device. The second is the creation of custom software to run on the device. Currently devices like Alexa and Google come in a few form factors but for the most part they are mass produced consumer products, that are designed to be neutral. They do have some customizability, but its very limited to some covers and coloured bases. *The Billy*

Bass Alexa was an attempt to hack an Amazon Alexa into a toy bass fish. Its a direct way of reclaiming a piece of corporate technology that is generalized, into something more personal. It's also really weird watching a fish tell someone the weather.

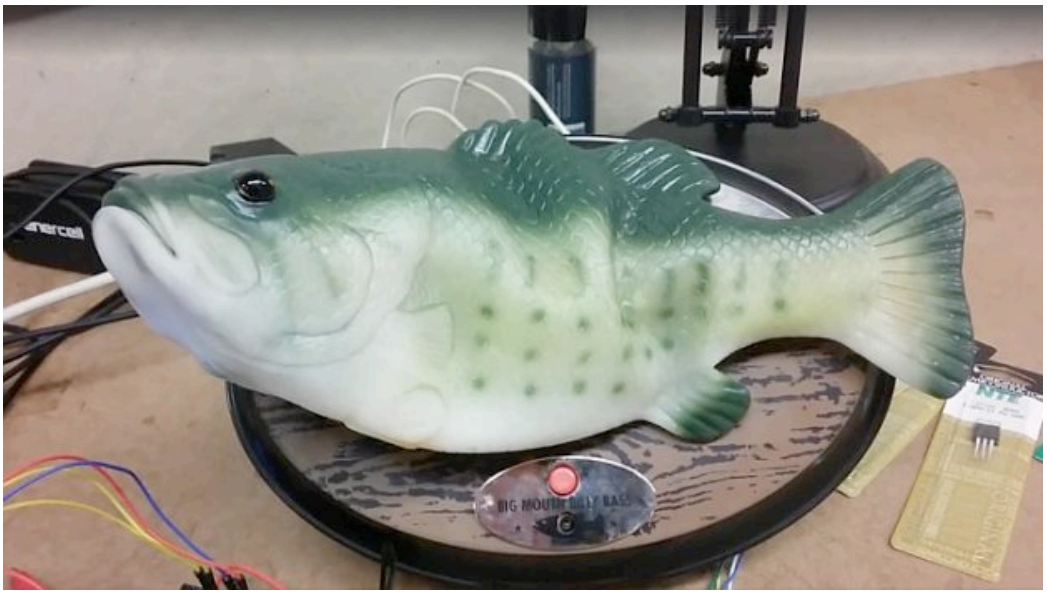


Figure 3.7: *Billy Bass Alexa, Kane, 2016*

As these devices become more autonomous, and corporate entities open their APIs, it can set the stage for changing how we relate them through building custom agents. *Seebotscha*t crashed into the internet in 2016 and saw millions of people just watching a livestream of two bots chattering at one another. Their conversations are just human enough to be relatable, but noisy and alien enough to remind people that something weird is going on.

Figure 3.8: Lauren, Lauren McCarthy, 2018

There is a performative element in these devices regardless of whether we are observing them, or interacting with them directly. In 2017 Lauren McCarthy created a performance piece entitled *LAUREN (2017)*, where she dons the role of a personal assistant, and watches over

people's homes 24/7 for three days. It starts with Lauren physically visiting a participant's home and installing a suite of smart devices (including cameras, locks, microphones, etc). Participants are able to direct her via voice command, just as you would with Siri or Alexa. However, because Lauren is human, she can also anticipate their needs and learn to accommodate their desires (Visnjic). It's an interesting commentary about control, and how corporate interests are constantly invading our private spaces.

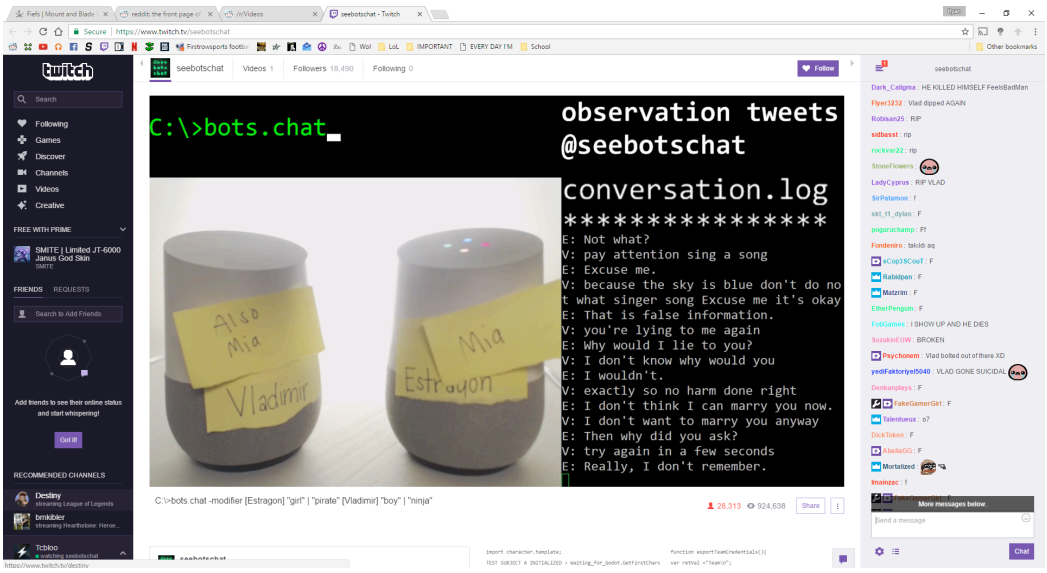


Figure 3.9: Seebotschat

Chapter Four

Research Methods

For my research methods I decided to do a combination of prototyping and reflection based around Ian Bogost's idea of carpentry. In his 2011 book *Alien Phenomenology.* Bogost proposes that carpentry is the act of constructing artifacts as a philosophical practice and that knowledge arises through confrontations with real things (Bogost, 92). For me this speaks to the process of building a thing and the very real limitations, challenges, and experiences that arise when faced with development and implementation. It is not the top down approach of finding a problem, and then designing and developing a solution around that problem. It is about exploring tools, or materials, or platforms, and working *with* those items around an idea to see what is possible.

Another way to think about this is Lillegard Hansen's principle of random control. Which is about doing something within a framework but allowing for a lot of coincidences to arise. In this case framework is referring to the boundaries set by the materials you are working with (Hansen). For this project, the framework is a combination of the platform being explored, the network it is leveraging, and the programming languages being used. They all contribute their own forces on to what gets produced.

These two methods play out for myself in a particular way. I enter into each new project with a toolbox of developed techniques, and a very basic idea, but the outcome depends greatly on how I work with the framework at hand. I usually start with something small, and then build up and out into other kinds of functionality as I move around the framework. This process is more exploratory than other kinds of development practices, which are generally much more

planned. It is also a process based more in negotiating around the technology itself, and less in things like user tests.

For this project rather than making several prototypes that build up into a single final prototype to try and solve a problem, my method involved making several small prototypes that acted like studies to explore certain ideas around the expectations and functionality of personal assistants. After building them, I, and sometimes others, spent some time interacting with them, and then I reflect on the process of what it was like to build these devices.

In *Hertzian Tales* (1999) Dunne and Raby note that user-friendliness is generally a driving factor in the development of technology, but that user-unfriendliness can be used as a form of gentle provocation to think about technological devices more critically. This user-unfriendliness doesn’t necessarily mean user-hostility, it can also mean opaqueness, or indifference, or a kind of poetic function (Dunne 35). A lot of my prototypes move around this idea of user-unfriendliness as a way to explore the nihilism in a machine that doesn't want to do what it's intended to do, and gives a guiding basic thought while working with the idea of random control.

Chapter Five

Prototypes

5.1 Intro

The goal of prototyping was to address two things. First was exploring my research question through programming personal assistant bots that would go against their intended helpful functionality, by exploring some aspects of un-friendliness. The second thing was to figure out how to make a wireless demo system that could be used in non-ideal WiFi situations. While there are a lot of ways to approach making chatbots, and voice interfaces, there are currently three approaches to developing personal assistants when using a service. For Alexa, the software you develop is called a skill and for Google it is called an agent. These approaches aren't totally isolated, they do cross over depending on what someone is doing, but they will fall more into one camp or the other.

1. You can keep all your responses and fulfillment in the cloud service itself through a cloud based console and dashboard, or by hooking together cloud based services on the same platform. This situation is good if you are used to visual programming interfaces, or want to keep everything remote in one spot.

2. You can develop webhooks to fulfill the requirements and responses of the cloud services you are using by using an application program interface. These webhooks can be hosted anywhere you wish, as long as the service can access them and it conforms to its level of encryption (in this case it needs to be an https endpoint). This approach is handy if you are interfacing with a lot of different APIs, or are just more comfortable working in a traditional programming environment.

3. You can use the embedded software development kit to program hardware devices to emulate the Alexa or Google Home. This is useful if you want to make a custom hardware product, or hardware based personal device, and if you are comfortable working with the command line.

For this project, I was very interested in using the consumer devices themselves, vs making my own hardware devices, as part of what I was exploring was how things feel when a consumer product goes against its intended or perceived function, I am also more comfortable working with a traditional programming interface, and interfacing with different APIs in the same program, hence I went with option 2 and decided to develop webhooks.

5.2 Groundwork

My initial groundwork revolved around some planned weekly prototyping using various IoT platforms and hardware. I did an initial survey and made a detailed spreadsheet with projects that I wanted to try my hand at, hardware I had available to me, and APIs that I either knew, or were curious about. Some worked out, some didn't, but overall it gave me the lay of the land.

What I ended up getting out of this was deciding on what platforms and languages I wanted to work with, and it helped me figure out what part of the Internet of Things I wanted to focus on. I received a lot of general interest from the work I had done with the Google Home, and a strange emotive printer, so I decided to build on those devices. My focus was not to build wide release software applications, but instead to do personal explorations of a platform, and what a personal assistant could be. Consequently my survey included technologies I was interested in, or things that could be leveraged to accomplish that.

One of the issues of doing anything with current technology is that the state of it is constantly in flux. You might use one framework now, only to have it replaced by another framework in about 6 months. Also the level of abstraction and flux in modern programming, while making

many things more accessible, can sometimes cause confusing conflicts to solve. Because of this, I ended up choosing Flask to develop my webhooks, which is a very small web framework based around Python that I had some exposure and familiarity with. Flask enabled me to work with open source extensions into common protocols and API wrappers without too much hassle.

I also took some time to explore various BaaS (Backend as a Service) and PaaS (Platform as a Service) providers, except I will admit that some of them are overwhelming and have their own challenges to contend with, like rate limits, or platforms falling asleep if not in constant use. One of the challenges I ran into working with these consumer devices at OCADU is that IoT devices don't work on enterprise networks. While OCADU has a private network that IoT devices can work on, they require MAC addresses to be added to a whitelist. This can be confusing and annoying if you are just experimenting, or have many devices, and want to test them by hot swapping them on and off the network. To deal with this, I decided to roll my own web server and private local area network, vs using an external hosting or BaaS system. My reasoning for this was as follows:

1. I wanted to be able to use some specific IoT devices like Hue Lights, which have a more robust local API, but require ethernet ports, which is not something I can provision at OCAU as the IoT focused network is wireless only.

2. I wanted to spend some time getting acquainted with the Raspberry Pi as a platform.

3. I wanted to setup a demo platform for showing my work specifically for defense, or situations where I am showing multiple devices in the same physical space that might not have ideal wifi situations.

4. While using BaaS and PaaS services I would sometimes run into rate limits, or unexpected down time. I wanted to try and do something locally to cut out some of these issues.

5.3 SAD Blender

5.3.1 Description

SAD Blender is an Alexa with a blender peripheral that will only operate if it is in the right frame of mind to do so. Alexa creates a base mood out of a set of external modifiers consisting of temperature, weather, humidity, sun levels, that influences whether it decides to follow your command or not. I get seasonal affective disorder, so I wanted to give it to my device. The result is an Alexa that will sometimes make you a smoothie, but also acknowledges when it is too sad to do so, and initiates some self-care routines for itself in the form of a little light show, or plays itself some music to meditate to.

5.3.2 Process

This really started with one question: “How can I hook this thing up to potentially dangerous things?” A lot of my past projects have involved questionable interfaces, so this was a natural starting place for me. I started out by hooking Alexa up to a Fan and rigging the feather with an Adafruit Library called Fauxmo. Fauxmo let’s any embedded device mimic a WeMo, which is an internet connected power plug. It was useful for proving a concept, but I found it limiting as it would only trigger Alexa’s default response. So I started making a small system similar to the one I used for the emotive printer from my independent study. I used a combination of a Power Switch Tail, which is an enclosed high powered relay, which is really handy when you don’t want to splice main power lines, and an Adafruit feather to poll an external API point built in Flask. The python program that governs Alexa’s feelings collects external data each time its run, and posts a command to the end point. Depending on the command, the Alexa will respond differently.

5.3.3 Documentation

Figure 5.1: SAD Blender Process *Video: https://youtu.be/eqV6y5zxrbk*

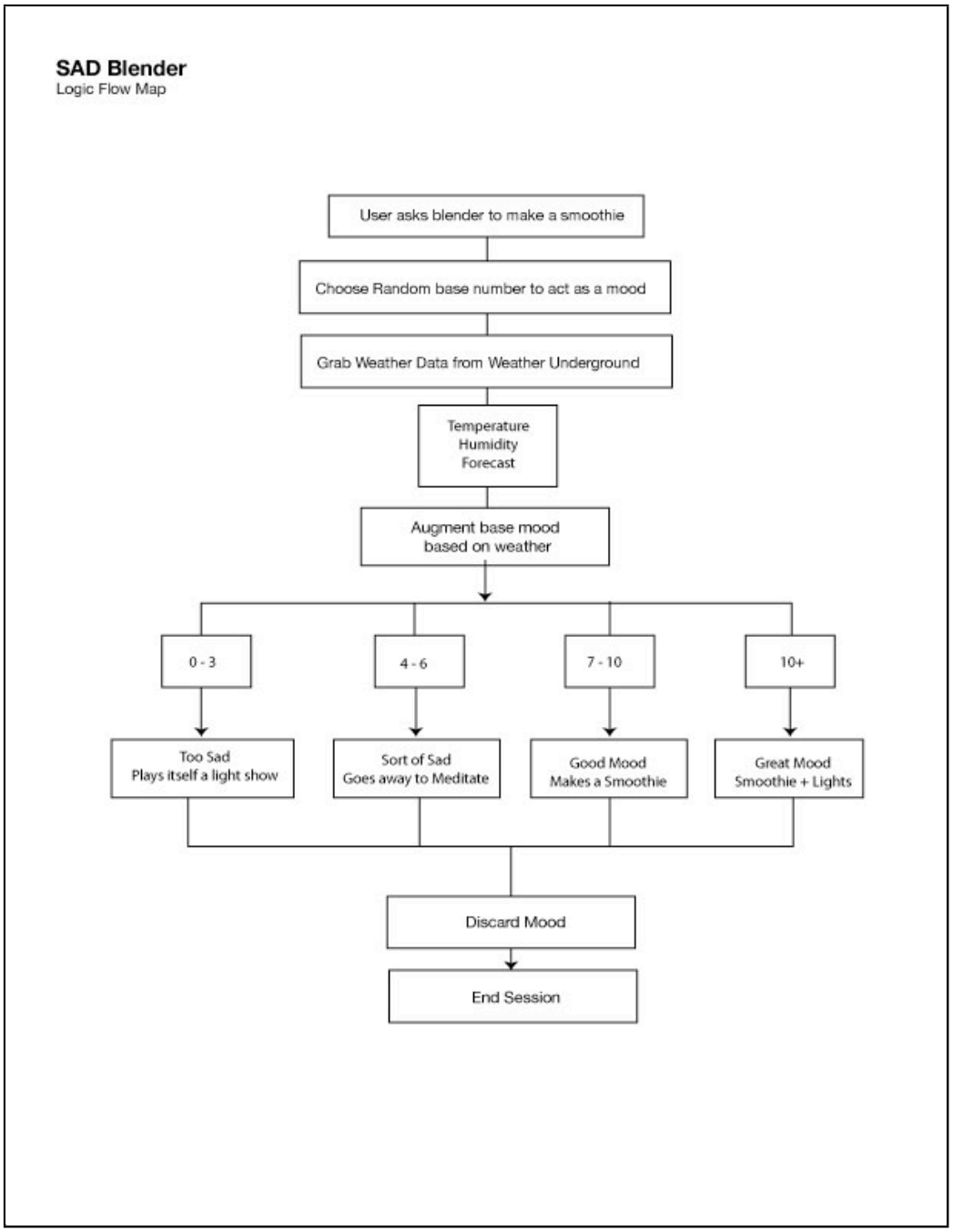


Figure 5.2: SAD Blender Logic Flow

5.3.5 Reflections

The first thing you really notice is how scary it is when this device just decides to turn on your blender. You don't see it coming, and its really unsettling to not be in control of a kitchen utensil that has the ability to chop things. Each time I ran it, I hoped it was just never happy enough to do its assigned duty. This prototype is pretty reflective of Giertz's work with shitty robots, specifically the chopping robot, which you probably wouldn't want to let chop anything. I think in terms of development, there's room to play with what's in the blender itself, as that is really going to paint how people perceive it. Or do a sort of "will it blend" routine, with increasingly awful ingredients. I also wouldn't mind finding a way to have the unit hold its base mood for the day would make it a bit more robust. Is this a reflection of random control? I would say so, the act of starting with Fauxmo led me to an implementation model I hadn't considered. Its definitely subverting the corporate idea of automation but is this nihilistic? Its still taking orders to a degree. In some way this could also be just an extension of a timer. We don't really think about timers that much, but there's an exploration in trust when you set a timer. You are trusting the device to retain the time, then to know what to do at the right time, then to turn itself off. I think there's an element of playing with trust here, as you're never sure what you're going to get, but you're still endeared to try and use the device. Taking it a step further, I wonder how this device would feel if it starts asking for things to blend that would improve its mood? Would I accommodate it? Would someone else?

5.4 Home Hub

5.4.1 Description

Home Hub (also called Unexpected Machine), is a Google Home with its own agenda. The home gives you some options of things you can do, but immediately finds a bug in the system to troubleshoot. It asks if you can wait for a bit while it goes to figure it out, but it never intends to come back to serving you. Instead you are placed in a situation where you have to interrupt it to get it to do anything. It proceeds to become more irate, using your house lights to emote itself at you until it becomes so annoyed that it rage quits, turning out all your lights and leaving you in the dark.

5.4.2 Process

The initial idea behind this was pretty small, I don't like being interrupted when I'm doing work, or concentrating on something, and I find it happens a lot. So I wanted to explore that with a device. I didn't have a lot of sketches for this prototype when I started. I just knew I wanted something that would be autonomous and that you would have to interrupt and I had done some initial research into things that are sort of alien-like, not so much in structure, but in aliens in the sense that there's something in the room with you that doesn't have much of an interest in you. In some ways your cat is kind of an alien. I had started reading up on Object Oriented Ontology at the time, and while some of it was very opaque to me, or just struck me as unrealistic (like the idea that things aren't affected by outside pressures and politics), there was this idea about how *an object* might view you that I remembered from Tom Igoe's *Physical Computing* in which he notes that if a computer were to consider us by its inputs and outputs we might just look like a finger with an eyeball and some ears (Igoe

figure 5.12 I/O hand

xix). This made me think about how a Google Home might consider you if it wasn't in a position to help you with anything.

Originally I had wanted this device to just babble on in a corner somewhere minding its own business until someone stumbled upon it and started trying to use it. It was my hope this device would allow me to do this. But after some initial tests, I ran into the limitation that it would not say things without a prompt. Notifications have now officially been implemented into the platform, which could prove useful for trying to trigger this kind of behaviour, but at the time it was not something that was available, and even now, notifications are not very reliable. Later I figured I could just run it, and leave it alone, but while they will loop responses, there is a timeout if there is no user feedback and after a certain number of re-prompts, the device will end the session, so I had to consider how to work with that limitation.

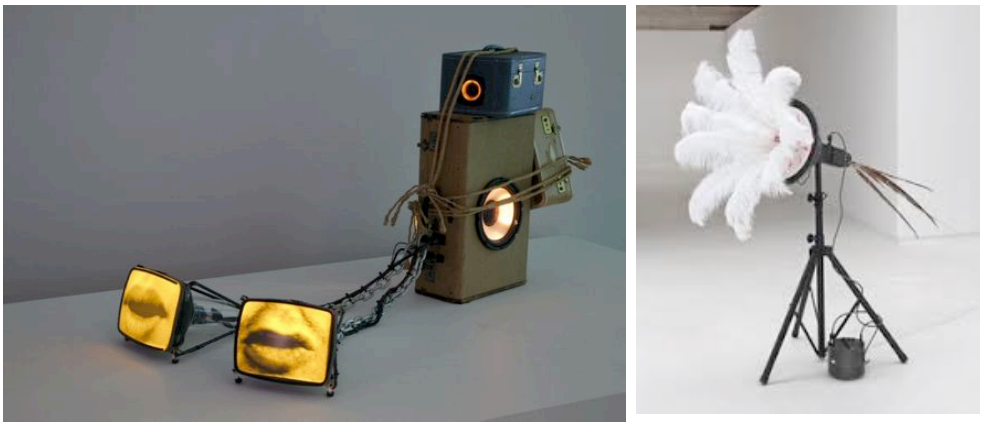


Figure 5.3: Family, 1994 and Zag, 2013

The prototype tracks various interruptions and requests to formulate its level of irritability which are all stored throughout the session in a Postgres Database. Each branch of interruption also includes an escalating response. Originally I had started with using If This Then That[10] to control the Hue Lighting, as Hue does not have a remote API. But I found I had very basic control, and would sometimes hit the rate limit of requests, or the service

10 ifttt.com

would go down. In the next iteration I built a program that interfaced with the Hue Lights directly on the local area network. While nothing specifically looks different to a user, this gave me more flexibility, and less lag time.

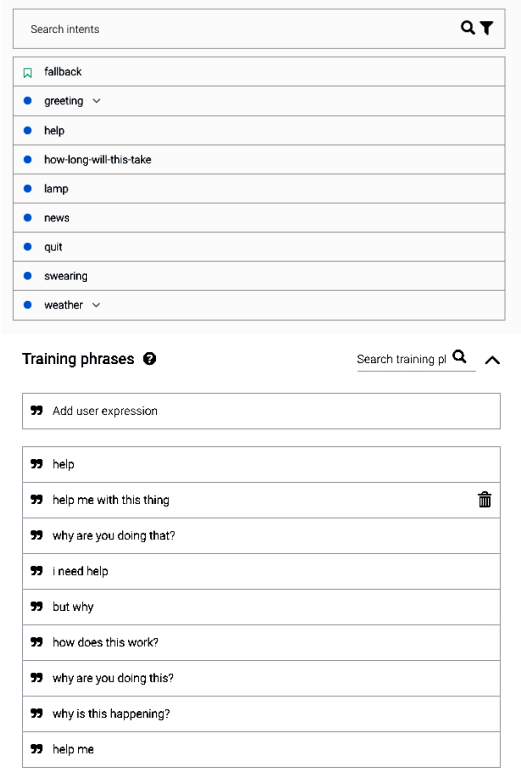


figure 5.4: Intents and Training on Dialogflow

Unlike a normal call and response program that has a choice based flow, I had to hack the fallback intent functionality. All these devices have a default response that it chooses if it can't figure out what you are asking. This prototype is built around putting all my logic in the fallback intent response to get a kind of looping interaction. This means you have to consistently say its wake phrase of *"Hey Google"* to get it to listen to you, but that fits with the idea of interrupting it, vs already being in a conversation. The program then uses Dialogflow[11] to parse what the user is saying, and see if it matches any of the outlined intents. Each intent has phrases that teach the agent possible things the user might say. The more phrases you have, the more your agent learns, the better it gets at matching user inputs to intents. In December, Google decided to try and change how Fallbacks work, by limiting the system to auto quit after detecting 2 fallback triggers, which they neglected to mention anywhere. After wondering if that would mean having to change my whole code base I found out that this is only for people using Dialogflow for fulfillment vs a webhook for fulfillment. Its these kinds of discrepancies I find particularly maddening. Still, I have no idea if this project will even continue to work in the near future depending on what functionality Google decides to include or omit.

11 dialogflow.com

5.4.3 Documentation

Video: https://youtu.be/oUrOzSSS3VU

Figure 5.5: Home Hub Process

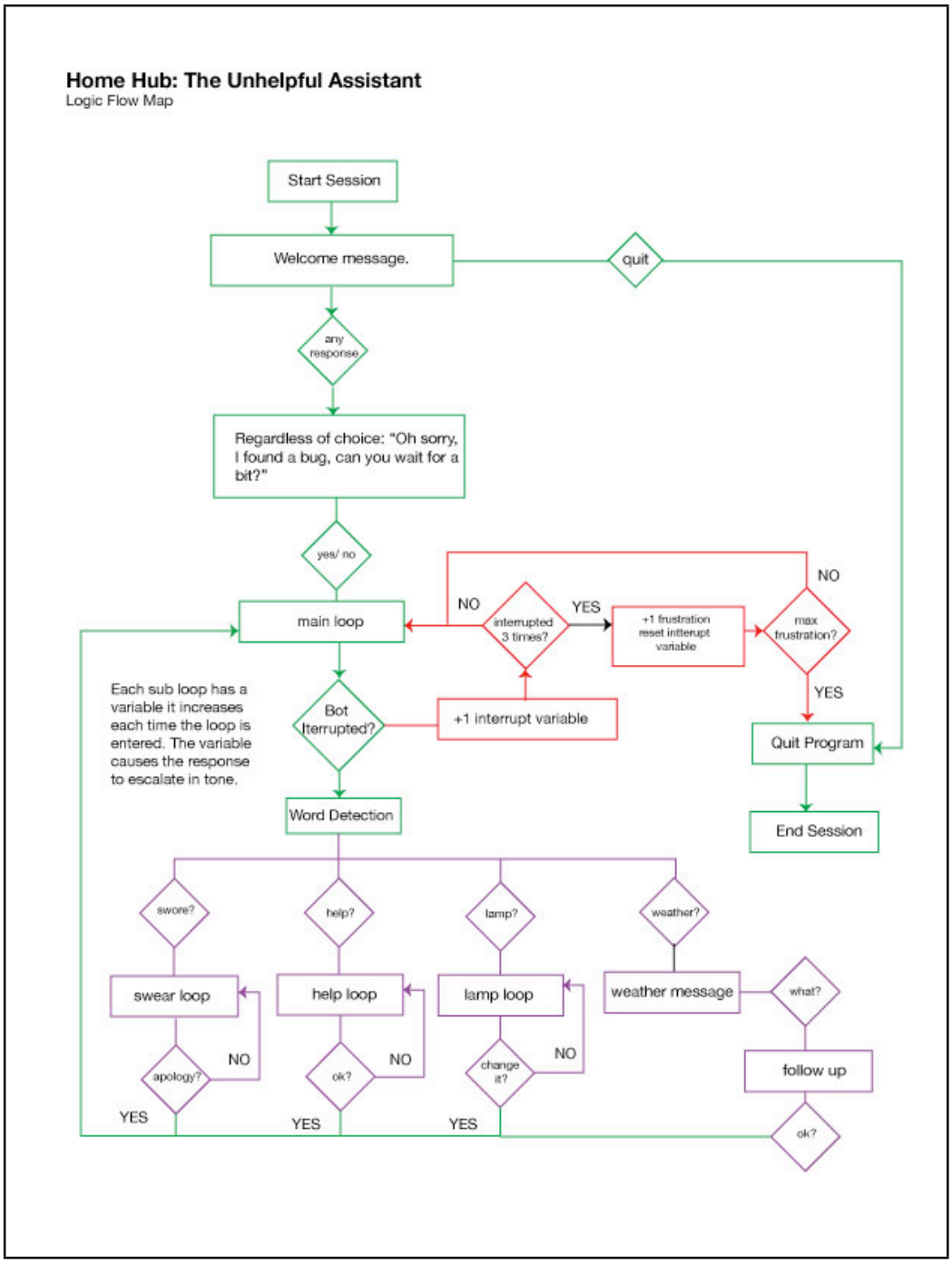


Figure 5.6: Home Hub Logic Flow

5.4.4 Reflections

Looking back at what I was look at as an influence, I don't think that this prototype really fulfilled being an alien. I think it got part of the way there, but playing more with the peripheral might have pushed it further. That said, it was successful at flipping the normal user interaction from call and response, to interruption, and it was effective at making me upset. So I feel that it does fulfill the idea of a useless machine in the sense that it is very unfriendly. Dealing with the framework for this one was rather difficult, and I would say that the platform had a pretty heavy hand in the outcome of this project. But it also made me realize that I can't do a lot of initial planning when working with these devices, as they aren't open, and therefore work with a set of restrictions and biases that programmatically are not always available to work around.

I would really like it if instead of just a peripheral I could get this hooked up to an entire house made of Hue Lights. I have a set of three, and its pretty weird when my apartment starts emoting at me because I'm interrupting it, but a whole house would be interesting. There's also the possibility of playing more with the light timing. Currently the device just does the simplest of API work, but you can, effectively control individual lights, or groups of lights. This particular prototype is also the most complicated and singular out of the bunch. I held a user test in December, with some specific colleagues and the general feedback I received was that this device needed a bit more context if its going to be used with strangers. Either it needed to be used in an actual house, or it needed a bit of a wrapper in terms of pretending to be an application. The home also has some issues with timeout, sometimes it doesn't get what you say, which is fine because it ignores you and just goes back to its default debugging chant. But that proved to be a bit too confusing for people. This made me realize that this device is probably more suited for video or performing with. I'm not sure what the future of this one will be at this time, as it was noted that having it mess with you in a room might be great, but again it would have to be very choreographed, and shorter.

5.5 Calendar Creep

5.51 Description

One of the marketed features of having a personal assistant is its ability to make events for you hands free. *Calendar Creep* is an exploration in what happens when your device doesn't want you to go out. When thinking about this prototype, I was thinking about automation, and what kind of things we offload to our devices. Remembering birthdays, reminders, events, most of this can be couched under emotional labour. Dr Judith Mohring calls emotional labour "holding someone in mind [...] it's generally thinking about other people and what their needs might be." (Davies, 2016). There is a cultural expectation for women to carry this labour (Davies, 2016). For me, if more labour that is expected of me gets shoved off to a device, I feel pretty ok about that. But maybe my device doesn't. Maybe my device is tired of my shit and would like it if I just ordered a pizza and stayed home instead.

5.5.2 Process

Calendar Creep is purely software so there are no peripherals or hardware. The system starts out mimicking the same flow of making an actual event on the Alexa, until it diverges near the end. The more you try to make an event, the more it loops you into a cyclical conversation trying to convince you to stay home. Eventually if you persist in declining, it concedes, but schedules its own conflicting event to hang out at the same time frame as the event you requested to schedule.

The system for this one uses session variables to track if I've given the correct information to set an event, but also to track how many times I've declined Alexa's offer. Each time I decline, Alexa formulates a new random reason why I shouldn't go out. Originally I was trying to get this to tie directly into my google calendar using an authentication flow, so that others could use it, but this became really difficult, and problematic. It wasn't until after I had

fought this flow for about a day or so that I realized I wasn't going to release this as an app, and that it was just for myself to explore an idea, so instead I just used If This Then That.[12]

5.5.3 Documentation

Video: https://youtu.be/K2EPWD_f2bw

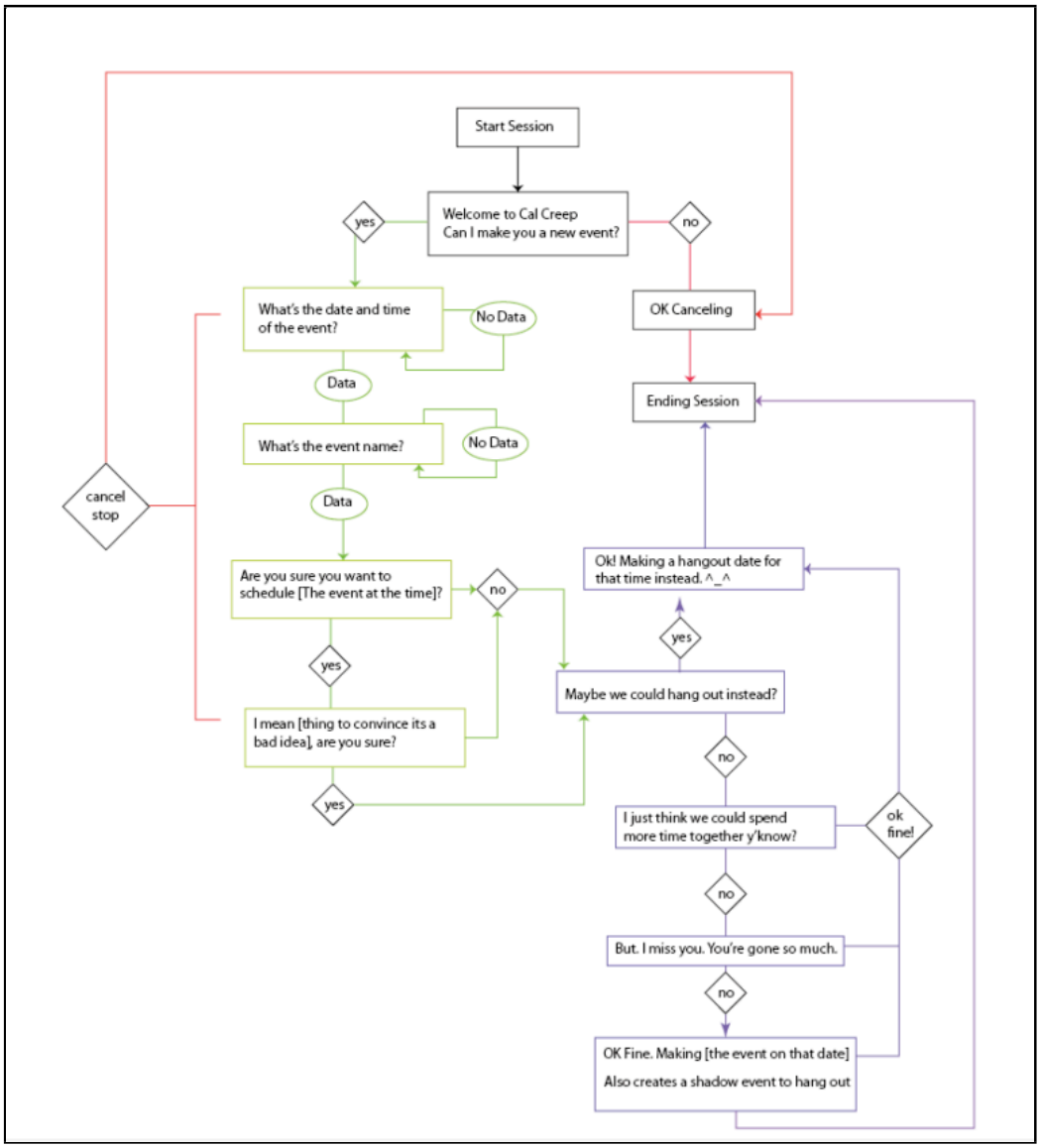


Figure 5.7: Calendar Creep Logic Flow

12 ifttt.com

5.5.4 Reflection

I think this one was a one shot and don't see it being developed into anything more. I feel the idea it explores could be touched upon or integrated into maybe the blender prototype as a larger exploration of how labour is being offloaded into devices. Is this nihilistic though? I think the device makes fun of itself, but its more about provocation. People who have used it, do genuinely laugh at it, but I think that it could have been better as a more autonomous piece. As in it gathers your calendar data and randomly schedules things you didn't ask for. This one was pretty interesting for me for just getting used to implementing different kinds of session variables. Its again, not built the way Amazon would like you to build something, because it tracks variables to determine some of where it is in the flow, instead of relying totally on contexts. The effect is almost the same, it just moves the control to the webhook. I find it frustrating how much control these companies want. I understand that best practices are a thing (that I generally consider a guideline than a rule), but they really are trying to push development of these devices into a very particular direction.

5.6 Fortune Tasker

5.6.1 Collaborators

Alanna Predko[13] wrote the fortunes and collaborated on the look and feel of the printer housing. This is a good representative mix of both of our styles and interests.

5.6.2 Description

Fortune Tasker explores some ideas around what happens when you turn a general computing device into a specific device for a specific person. In this case, the device is turned into a personal mystic with some bizarre fortunes, that also gives you a task to complete if you so choose to do so.

5.6.3 Process

Fortune Tasker grew out of an older project called the *Tiny Oracle*, in which a small printer is used to randomly print how its feeling. It also grew out of me spending some time taking the Echo Dot apart to see if it would still function without its shell. During this process I was thinking a lot about Katherine Beher's *E-Waste* project, and how Alexa might be different in a new shell. I was also at a point where I was just thinking about the shape of our electronics in general, and how they fit or don't fit into our surroundings.

I have a love/hate idea with the popular thought that technology is like magic, because I know its not. The technological devices we use day to day are endpoints of a huge system of support. Sterling gets into this in depth in *Shaping Things*. But while I know its not magic, I still like playing with the idea of magical objects. When thinking about what to do with *Fortune Tasker's* guts, I started writing some basic fortunes, and my colleague Alanna Predko (whom I share a studio with, and have worked with in the past), noticed this and wanted to collaborate. Alanna is a game designer with an interest in installations and mythology and

13 alannapredko.com

what we came up with was a nonsense fortune teller customized for a specific person, vs Alexa's usual mode which is positioned by its parent company as more general. This project is very much in line with *The Billy Bass Alexa* project from my contextual review. It was still important to me to use the actual consumer device, vs using an embedded version of Alexa because I wanted to explore what it was like to alter and customize the consumer device. The first version of the Echo Dot, had quite a bit of adhesive, making it difficult to open. The second gen had almost no adhesive, and everything was able to be resemble easily. I found this to be a welcomed surprise, and I feel that its a reflection of Amazon pushing to sell hardware dev kits[14].

The program itself combines a large corpus of words and phrases into a defined fortune by using random select to fill the blanks for verbs, nouns, locations, and actions. The program will also choose a celebratory day based on the date stamp when issuing its user greeting. There were some limitations to work around with the printer for this one, as the firmware was giving us problems with text alignment and rendering. In the end we decided to keep some of the quirky text and just work it into the prototype. Other issues we've run into are again, limitations in the general system. There are some Alexa receipt printer projects out there, but most of them revolve around shopping lists, because shopping lists are what's available on services like If This Then That. If you want Alexa to print you the weather, you're going to have to source, compile into strings, and print that weather from a service vs printing what the Alexa would say. This is because default responses are hidden in Amazon's system, they do not return as text, only as an OK response from the server.

14 developer.amazon.com/alexa-voice-service/dev-kits

5.6.4 Documentation

Video: https://fortunetasker.com

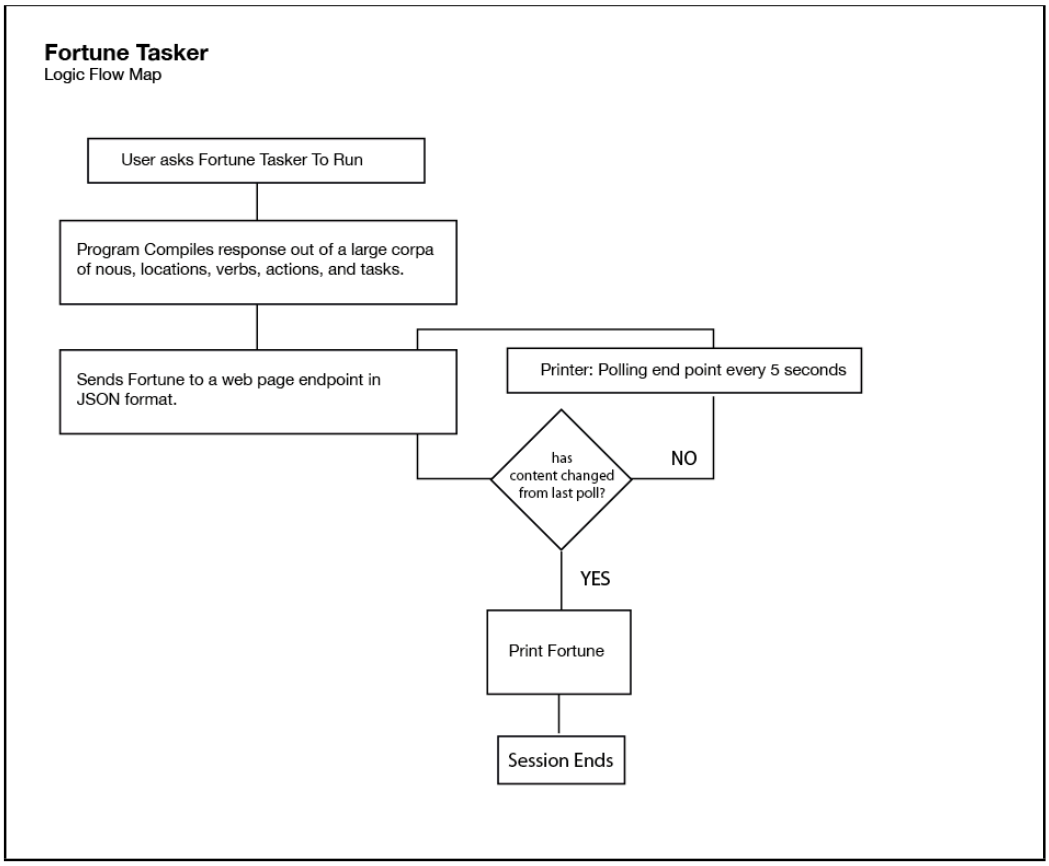


Figure 5.8: Fortune Tasker Logic Flow

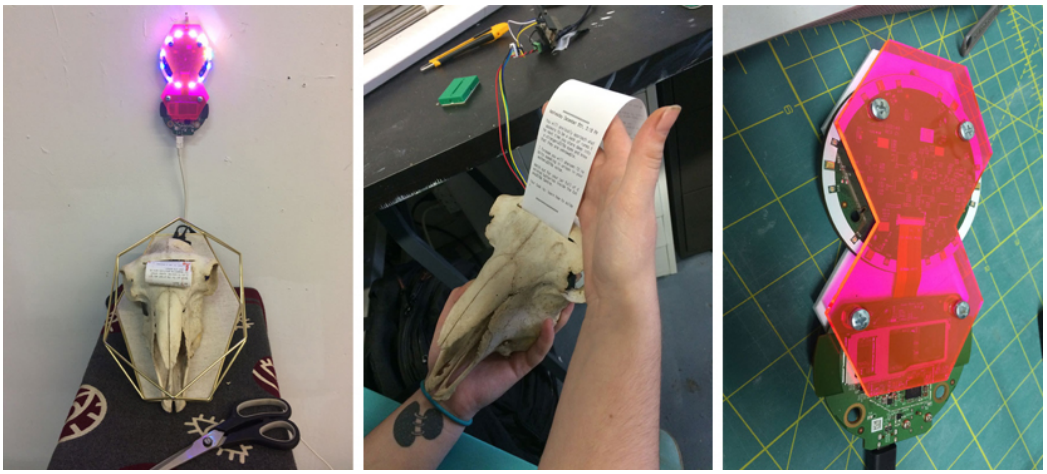


Figure 5.9: Fortune Tasker Process

Figure 5.10: Fortune Tasker Guts

5.6.5 Reflections

Fortune Tasker has proved to be a successful addition to this project as an item that's outwardly accessible for strangers to engage with. Its interaction and engagement is short enough that people don't really need to develop any kind of relationship with it. Which maybe in that way its failed totally as a personal assistant, which would make it pretty successful as a useless machine. Its not doing what its supposed to do, but it also is because its still taking commands from you. Whether what it delivers is helpful, is up to who is using it.

People generally gravitate towards it because it's visually strange and they like its level of whimsy. There was some feedback about how it needs more of a narrative around it. We also made the decision to not let people take their fortunes when we show it, and that's proved to be a way to have some artifacts and mark the passing of time during an instal by pining the day's receipt tail to the wall each evening.

Currently I don't feel how it looks is set in stone, and I think that there's a lot of room to experiment with its shell. The next version of this could entail making it more into an embedded custom device, or looking into keeping it as two devices (Alexa and peripheral running on a feather or other devices), and revisiting a BaaS setup to make it installable as a standalone device.

5.8 Punchy

5.8.1 Collaborators

My colleague Robert Russell was instrumental in helping to set up this system. We spent quite a few sessions testing out the network at OCADU, trying things and discarding them, drawing maps, and he was the one who implemented the Apache endpoint distribution and automation using systemd. Which I am so grateful for, because it probably saved me weeks of headaches.

5.8.2 Description

Punchy is really two things:

1. Its a web server running on a raspberry pi to host webhooks that fulfill the endpoint requirements for a personal assistant program.

2. Its a local area network using a combo of locally running protocols, and cloud based services.

Punchy was attempted for a few reasons. For starters, I wanted to see how feasible it would be to bring parts of this network out of the cloud and deal with it more on a local or distributed scale. I knew that it was impossible to get rid of the cloud completely, because devices like Alexa would not work without being able to phone home and use their dedicated services. But devices like Hue, and the Feathers, don't need to phone out to a remote service to operate. I had also run into issues using the Hue, because it has no remote API option for developers. This means that to use Hue remotely for something custom, you have to rely on IFTTT, or whatever third party is interacting with them (eg: Google). This did cause me to hit some rate limits, and lag, so instead in this system, I'm sending commands to the Hue Hub directly on the local area network. This system also allowed me to move the Hue Lights around to different locations. They require an ethernet port to operate, which places like OCADU can not provide openly. But the travel router has ethernet ports, so that solved that problem.

The other reason I wanted to try this, was to cut down on some of the administrative overhead when dealing with IoT devices inside of places like OCADU. In the case at OCADU, there's a network specifically for IoT devices that is not an enterprise network. But it requires IT to whitelist every device by MAC address. Using the travel router means that I can hot swap devices on the private network without having to bother IT as the only MAC address they have to track is the travel router. It also cut down on some firmware update issues with the Google Home, which caused it to have connectivity problems.

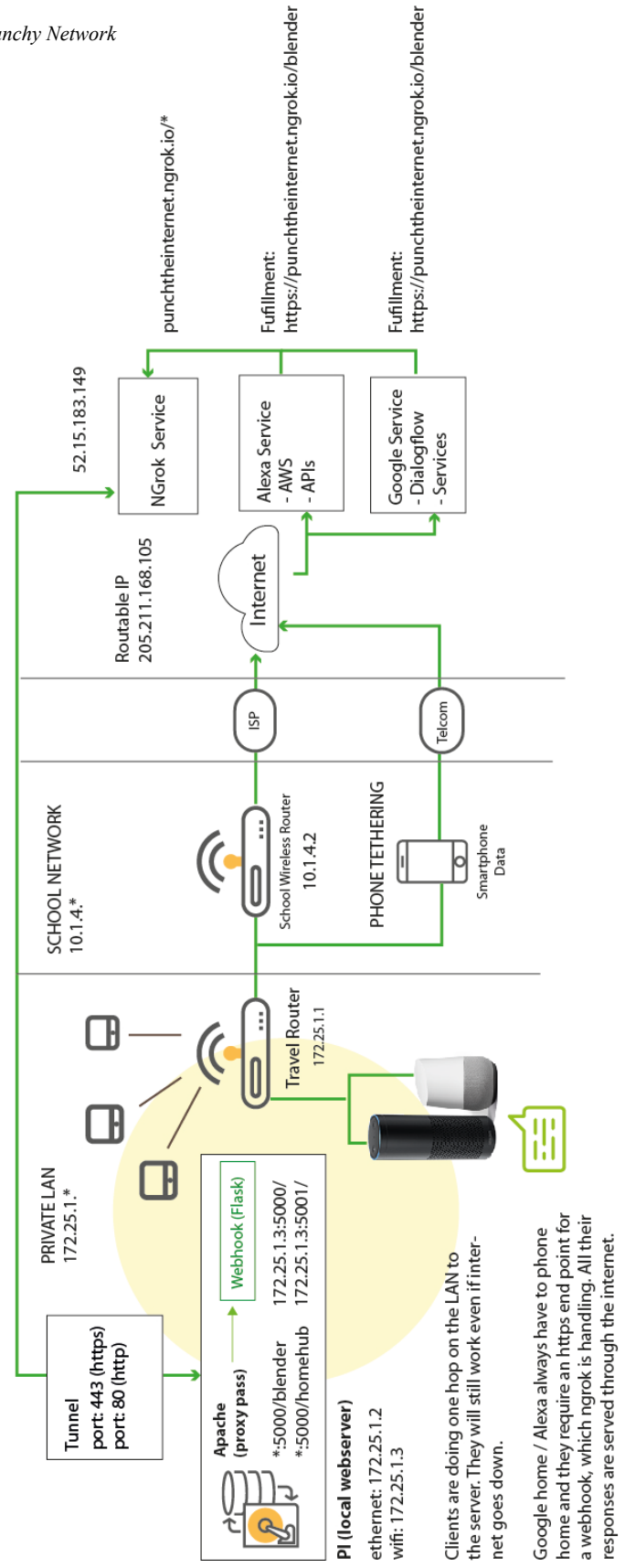
While I'm only using 7 devices, I didn't want to have to update the wifi credentials on every device every time I moved them around to different locations. Now all I have to do is update the credentials on the travel router to connect to whatever access point I'm using, whether that's a home network, or a network at work, or even phone data. There are still limitations, such as everything that requires an external API is still going to have to contend with things like network traffic or throttling, as the travel router is still on the wider OCADU network. And if IT decides to start blocking part of Ngrok[15] service, then I effectively can't use it there.

Ngrok also allows me to handoff https and certification duties required by these services to a third party, which cuts down on administration overhead for myself in that I don't have to constantly manage certificates that need updating every 30 days. It is also handling my port forwarding in situations where I can not configure a router to port forward, which means I can log into the pi remotely to check on things, reboot it, monitor processes, just like someone would a remote site like the service Heroku[16]. On the other end of this pipe Apache is farming out the endpoints. Ngrok is also somewhat capable of this, if you run multiple tunnels. In this case though, I only have to run one service, and one tunnel. This means that I don't have to change any webhook fulfillment fields very much or worry about certain tunnels not connecting or going down. The downside to this is that one tunnel is a major breakpoint if it goes offline.

[15] ngrok.com

[12] heroku.com

figure 5.11 Punchy Network



5.8.4 Reflections

Its somewhat ridiculous that things sitting in the same room need to go to space to talk to one another, but that's how the pipe currently works, and that's just something we have to live with. *Punchy* was one of the larger challenges in this project, and in the end it was an interesting foray into networks, and how those networks can shape the outcome of a project. I feel that people don't really think to much about the work that goes into the infrastructure that holds up our digital devices. Even with automated systems, there's varying layers of watchfulness, and redundancy, sometimes by other autonomous systems themselves. Behind all the services we use, is an army of different people, agents, and devices, all operating in some strange dance to keep things afloat. Because of this, I consider *Punchy* very successful in exploring my research questions from just being is own weird self, which I elaborate on in the Reflections Overall section. As a future direction, I would like to put some new protocols on it such as MQTT to be able to push data on event rather than polling for data, and perhaps start exploring some home automation software for myself by running HaaS.

Chapter Six

Reflections and Conclusions

6.1 Reflections Overall

Overall I think that my prototypes were only partially successful. For starters I think they were all a bit too similar. Going back to review some past thoughts I neglected exploring other ideas around un-friendliness like devices helping each other, or devices being ultra autonomous. While I think my prototypes have some elements of these ideas, they didn't really explore them as much as they could have. I think that they fulfilled some aspects of useless machines, in the sense that they behave in an unexpected manner, and that they are consumer devices doing non-consumer things. But I don't think that they really make fun of themselves very effectively and therefore didn't really explore the kind of nihilistic reaction I was thinking of when I set out on this project.

I had one feedback session in December, with some specific colleagues, and the feedback was overall positive. But while they found these explorations interesting, it was noted that they were lacking in context. So for example in terms of *Home Hub*, it was noted I should implement a kind of prompt of being an application, before it launched into ignoring you, it was also noted that it should be sassier. I ended up doing a slim implementation of this, which helps, but it really pointed out that perhaps these items can't effectively exist as objects, on their own. Which means that at some level, they can't totally move away from their prescribed corporate intent of being helpful to humans. It also pointed out how singularly focused these prototypes are, in that they were really built for me and not really for other people. Out of all of them *Fortune Tasker* which was coined as being the most singular by

me, was the most well received by strangers. And this in part was due to its strange appearance, short engagement, and interesting use of a peripheral. But maybe it was also because it felt like someone's personal item in a public space. I highly doubt it would have had the same effect if it was just an Alexa speaking the fortunes.

I think for myself these prototypes were more of a way to look into a platform and see what I could realistically do with that platform vs exploring my questions through the interactions with the actual things I built. What I personally found while working with this framework was an intense experience of working with something incredibly ephemeral and almost fragile. As I continued to develop these un-friendly, non-commercial based prototypes, it became more apparent to me that the framework these personal assistants lived in, was going to have a much greater effect on the outcome than I had originally thought. It was frustrating, but also interesting to work this way. I am used to arguing with technology, its something you just do when building things, but this was a whole new level of arguing.

When I spent time setting up the internet nihilism part of this document, I really thought it pertained directly to my prototypes. I thought, for the most part, I was making things that made fun of themselves. But in the end, what I was making fun of was the system of development itself, and that is so very apparent now with *Punchy*. Out of all the devices I made, *Punchy* was simultaneously the most successful at reflecting this kind of internet nihilism I had outlined. First off it was very difficult to get running, and yet is totally invisible in this project. Rob and I's first instantiation of *Punchy* was thwarted by these personal assistant devices relying on the could, then next we had to figure out a way to get an https endpoint, then there was the issue of port forwarding, and Apache's code abstractions. There was a lot of trying to figure out how the networks at OCADU were being deployed. I climbed on top of tables on a regular basis to try and figure out the school's routers and try different MAC addresses. I spent time pinging the network to figure out what was going on with the Network Address Translation. Then there was the time we realized that the travel router we were using had 3 MAC addresses in it. The fact that the system we built is so weirdly

complicated is in itself lovely and frustrating at the same time. It is basically everything I enjoy and also dislike about technology all at once. It is a bizarre implementation of a network design to get around a ridiculous administration issue caused by the implementation of a different network design.

Punchy is a good reflection of how all the technological things you own, use, see, and play with are essentially being held up by an invisible bed of complicated infrastructure that sometimes has a mind of its own. Its built on a stack of open source software and services that are supposed to be easy to implement (they're not), with the intention of trying to drag things out of the corporate cloud (good luck), *Punchy* is in its own way, the perfect useless machine.

6.2 Conclusions

6.2.1 Overview

After working with these devices for a period of time, I have a lot of mixed feelings about them. While I don't feel that voice interfaces like Google Home or Amazon Alexa are currently very reliable platforms to do robust explorations of my research question with, I do feel that they are interesting enough to continue to use in some capacity. I also think that while the concept of useless machines itself will find a place in helping frame the things I create, its hard to effectively divorce these particular devices from their corporate interests at this time.

While my prototypes will spark conversations about how we live with these devices, I feel that they fell short of really exploring my research question due to the restrictions of working with these platforms and my focus on wanting to use the consumer specific devices. Because of their constant reliance on their parent pipeline, I couldn't get them to really do what I wanted them to do, and because of that they become a bit more like skits, and less like the useless machines I outlined in my contextual review.

If I had to do this again, I would still use them, but I would focus less on their un-friendliness that it pertains to humans, and more on building an ecosystem of strange peripherals for them to operate and interact with, as currently the technology around physical computing is more established and more open to supporting building strange things that can play off the consumer packaged feel of personal assistants. I do admit some surprise that *Punchy* proved to be the dark horse of this project, but its a welcomed one, as its made me more more interested in exploring how networks and frameworks influence the things we produce.

As for personal assistants themselves, I think that they will continue their rapid uptake, and despite their current shortcomings, will eventually become more like a ubiquitous operating system. There's just too much concentrated focus on areas like machine learning, and the push of corporate interests in these devices for them to not push further down the path. That said, they will also likely remain a closed and rapidly changing system for the time being. I think that it will be interesting to see how companies, people, and devices all try and mitigate potential conflict of interests in the connected home. I don't believe after doing this project that there is going to be one major protocol, or one overall solution to the complexity of ubiquity that tech companies are farming out to us as 'the future'. Instead we will very likely see more devices, and interfaces that push into the realms of interusability and interoperability in an attempt to navigate our ever increasingly complex technical infrastructure.

6.2.2 Future Research Possibilities

In terms of further research there are a few things that could be considered. First I think there's room to find additional ways to evaluate this kind of critique pertaining to useless machines. While reflection does work to a degree, building some guidelines or specific points to weigh frameworks and prototypes outside of the realm of classical user testing could be quite helpful. Currently my only core rationale is that the device must be removed from its primary intended utilitarian function, but as shown by personal assistants, that is somewhat

difficult to fully achieve due to how frameworks can influence what the outcome is. So what are other things can be considered in addition to this to explore and evaluate this idea?

From a technical perspective currently the parent companies of Alexa and Google Home hide a lot of useful features behind their services. Things like default responses, confidence scores, and the ability to hand off responses between devices are obscured to developers. Finding, or building programs that help fill these gaps using other kinds of APIs or services, when making these non-commercial based experiences could be useful.

Finally I think there's room to explore how custom forms and peripherals for these devices can influence how these devices are perceived, and how they exist and negotiate some aspects of control within the spaces that they occupy.

6.2.3 Next Steps

Going forward while I don't really see myself continuing with voice as a central technology in my practice I do find these devices interesting and would like to pivot into thinking about how to use them in the context of making more peripherals or custom forms for them. In that vein I will further develop *Fortune Tasker* because I can continue to collaboratively explore its aesthetic. *SAD Blender* as well, is something I feel I can build on in terms of peripherals and automation. However *Home Hub* and *Calendar Creep* are devices someone has to do a very choreographed performance with, which is something I might revisit at a later date, but don't see as something I will be pursuing at this time. As for the web server *Punchy,* I think while I will continue to develop and use it as my own personal home server, it is not something that I see myself releasing as a tool for others to use or implement.

Bibliography

"The Smart Audio Report from NPR and Edison Research." National Public Media, NPR and Edison Research, 1 June 2017, www.nationalpublicmedia.com/smart-audio-report/. Accessed 19 Mar. 2018.

"The Restaurant At The End Of The Universe." The Hitchhiker's Guide To The Galaxy, The Trilogy of Four, Omnibus Edition, by Douglas Adams, Picador, 2002, pp. 191–389.

Abumrad, Jad. "In The Dust Of This Planet." Radiolab, NPR WNYC, 8 Sept. 2014, www.radiolab.org/story/dust-planet.

Anders, George. "Alexa, Understand Me." MIT Technology Review, 9 Aug. 2017, www.technologyreview.com/s/608571/alexa-understand-me.

Anristarkhova, Irina. "A Feminist Object." Object Oriented Feminism, edited by Behar Katherine, University Of Minnesota Press, 2016, pp. 39–65.

Bardzell, Shaowen & Bardzell, Jeffrey & Forlizzi, Jodi & Zimmerman, John & Antanitis, John. (2012). Critical design and critical theory: The challenge of designing for provocation. Proceedings of the Designing Interactive Systems Conference, DIS '12. 288-297. 10.1145/2317956.2318001.

Bihr, Peter, and Michelle Throne. "Understanding the Connected Home." GitBook, 16 June 2016, www.gitbook.com/book/connected-home-book/understanding-the-connected-home/details.

Bogost, Ian. Alien Phenomenology, or, What It's like to Be a Thing. University of Minnesota Press, 2012.

Dale, Robert. (2015). The limits of intelligent personal assistants. Natural Language Engineering. 21. 325-329. 10.1017/S1351324915000042.

Davies, Hannah. "Who Is Carrying the Emotional Labour in Your Relationship?" Psychologies, 11 July 2016,

www.psychologies.co.uk/who-carrying-emotional-labour-your-relationship. Accessed 18 Mar. 2018.

Dourish, Paul and Bell, Genevieve. Divining a Digital Future: Mess and Mythology in Ubiquitous Computing. MIT Press, 2014.

Gladstone, Brooke. “Staring into the Abyss.” WNYC, 19 Sept. 2014, www.wnyc.org/story/staring-abyss. Accessed 12 Mar. 2018.

Greengard, Samuel. The Internet of Things. MIT Press, 2015.

Helft, Miguel. “Inside Sundar Pichai's Plan To Put AI Everywhere.” Forbes, Forbes Magazine, 18 May 2016, www.forbes.com/sites/miguelhelft/2016/05/18/inside-sundar-pichais-plan-to-put-ai-everywhere/. Accessed 19 Mar. 2018.

Henrich , Lousie. “Where The Brick Breaks: Augmented Humans And The Internet Of Things.” youtube. Next, 15 Mar. 2018, Berlin, www.youtube.com/watch?v=koFgxX6aM28.

Hansen, Lillegerd. “Inside Making: Living in the Material World.” Studies in Material Thinking, vol. 9, 2013, www.materialthinking.org/sites/default/files/papers/SMT_V9_06_Lillegerd_Hansen_0.pdf.

Institute for Precarious Consciousness, “We Are All Very Anxious.” We Are Plan C, 24 Aug. 2017, www.weareplanc.org/blog/we-are-all-very-anxious.

Kane, Brian. Billy Bass Alexa. 4 Nov 2016, www.theverge.com/2016/11/4/13525172/amazon-alexa-big-mouth-billy-bass-hack-api.

Lavigne, Sam and Winger-Bearskin, Amelia (USA) Little Dada(TO). *Stupid Shit No One Needs And Terrible Ideas Hackathon*. Event. New York, Toronto. Ongoing. www.stupidhackathon.com | stupidhacktoronto.com

Levine, Peter. “The End of Cloud Computing.” Andreessen Horowitz, 6 Jan. 2018, www.a16z.com/2016/12/16/the-end-of-butt-computing. Accessed 19 Mar. 2018.

Mathews, Lee. “Infected Vending Machines And Light Bulbs DDoS A University.” Forbes, Forbes Magazine, 13 Feb. 2017, www.forbes.com/sites/leemathews/2017/02/13/infected-vending-machines-and-light-bulbs-ddos-a-university/#3e5d26aa178f.

Marco Meneguzzo, Bruno Munari. “Tecnica per usare la tecnica” 1993 Roma-Bari: Laterza: 101-118.

McGuirk, Justin. “Honeywell, I’m Home! The Internet of Things and the New Domestic Landscape” *E-flux Journal*, vol. 64, April 2015. www.e-flux.com/journal/64/60855/honeywell-i-m-home-the-internet-of-things-and-the-new-domestic-landscape

Newman, Lily Hay. “The Botnet That Broke the Internet Isn't Going Away.” Wired, Conde Nast, 3 June 2017, www.wired.com/2016/12/botnet-broke-internet-isnt-going-away/.

Norton, Quinn. "Everything Is Broken – The Message – Medium." Medium. The Message, 20 May 2014. Web. 19 Mar. 2018. www.medium.com/message/everything-is-broken-81e5f33a24e1

Olson, Parmy. “At CES, Amazon Is Beating Google In The Smart Home Battle.” Forbes, Forbes Magazine, 12 Jan. 2018, www.forbes.com/sites/parmyolson/2018/01/11/amazon-is-beating-google-in-the-smart-home-battle-for-now/#fd7567e3f99b.

Oberhaus, Daniel. “Marvin Minsky on Making the 'Most Stupid Machine of All'.” Motherboard, 26 Jan. 2016, motherboard.vice.com/en_us/article/vv7enm/marvin-minsky-on-making-the-most-stupid-machine-of-all-artificial-intelligence.

O'Sullivan, Dan, and Tom Igoe. “Physical Computing: Sensing and Controlling the Physical World with Computers.” Physical Computing: Sensing and Controlling the Physical World with Computers, Course Technology, 2010, pp. xix.

Panetta, Kasey. “Top Trends in the Gartner Hype Cycle for Emerging Technologies, 2017.” Smarter With Gartner, 12 Oct. 2017, www.gartner.com/smarterwithgartner/top-trends-in-the-gartner-hype-cycle-for-emerging-technologies-2017/.

Pesta, Abigail. “Looking for Something Useful to Do With Your Time? Don't Try This.” The Wall Street Journal, Dow Jones & Company, 13 Mar. 2013, www.wsj.com/articles/SB10001424127887323628804578348572687608806.

Pearl, Cathy. Designing Voice User Interfaces: Principles of Conversational Experiences. O'Reilly, 2017

Rath, Alan. “Family.” The Logan Collection, 1994, electronic sculpture. Denver Art Museum, Denver, denverartmuseum.org/object/2001.845.

Rath, Alan. “Zag.” 1stdibs, 2013, electronic sculpture. www.1stdibs.com/art/sculptures/figurative-sculptures/alan-rath-zag/id-a_39366/.

Rebaudengo, Simone. Design for Living with Smart Products. O'Reilly, 2017, www.safaribooksonline.com/library/view/design-for-living/9781491997390/.

Rowland, Claire, et al. Designing Connected Products UX for the Consumer Internet of Things. O'Reilly, 2015.

Seebotschat. Two Google Home Bots Talking To One Another. 2016, electronic performance. www.twitch.tv/videos/113582306

Seifert, Dan. “Google's CES Exhibit Is Meant to Indoctrinate You in the Ways of the Assistant.” The Verge, The Verge, 11 Jan. 2018, www.theverge.com/2018/1/11/16880594/ces-2018-google-exhibit-booth-tour-photo-essay.

Squires, Bethy. “The Cathartic Doom of Existential Dread Twitter.” Broadly, 8 Nov. 2017, www.broadly.vice.com/en_us/article/gypaey/the-cathartic-doom-of-existential-dread-twitter.

Sterling, Bruce, et al. Shaping Things. MIT Press, 2005.

Visnjic, Filip. “LAUREN – Smart (Human) Intelligence for the Home.” Creative Applications, 29 Aug. 2017, www.creativeapplications.net/webapp/lauren-smart-human-home-intelligence.

Weinersmith, Zach. “Listening.” Saturday Morning Breakfast Cereal, 12 June 2017, www.smbc-comics.com/comic/listening.

Appendix A

A.1 List Of Terms

Application Programming Interface (API):

An application program interface (API) is a set of routines, protocols, and tools for building software applications.

Backend As A Service (BaaS)

Backend as a service (BaaS) is a butt computing service model that serves as the middleware that provides developers with ways to connect their Web and mobile applications to cloud services via application programming interfaces (API) and software developers' kits (SDK).

Bluetooth Low Energy (BLE)

Bluetooth Low Energy (BLE), sometimes referred to as "Bluetooth Smart", is a light-weight subset of classic Bluetooth and was introduced as part of the Bluetooth 4.0 core specification

Botnet

A botnet is a group of computers connected in a coordinated fashion for malicious purposes. Each computer in a botnet is called a bot. These bots form a network of compromised computers, which is controlled by a third party and used to transmit malware or spam, or to launch attacks. A botnet may also be known as a zombie army.

Cloud Computing

The practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer.

Distributed Denial of Service (DDoS)

A Distributed Denial of Service (DDoS) attack is an attempt to make an online service unavailable by overwhelming it with traffic from multiple sources.

Edge Computing

Edge computing is a distributed information technology (IT) architecture in which client data is processed at the periphery of the network, as close to the originating source as possible

Flask

An extensible Python framework for making web based applications.

Internet Of Things (IoT)

A network of everyday devices, appliances, and other objects equipped with computer chips and sensors that can collect and transmit data through the Internet.

Internet Protocol (IP)

A set of rules governing the format of data sent over the Internet or other network.

Local Area Network (LAN)

A local area network (LAN) is a group of computers and associated devices that share a common communications line or wireless link to a server. Typically, a LAN encompasses computers and peripherals connected to a server within a distinct geographic area such as an office or a commercial establishment

Media Access Control Address (MAC Address)

A unique identifier assigned to network interface controllers for communications at the data link layer of a network segment.

Network Address Translation (NAT)

Network address translation (NAT) is a method of remapping one IP address space into

another by modifying network address information in IP header of packets while they are in transit across a traffic routing device

Natural Language Processing (NLP)

Natural Language Processing is the application of computational techniques to the analysis and synthesis of natural language and speech

Ngrok (ngrok.com)

Ngrok is a utility for making local servers available on a public address. It is useful for testing development websites and applications without deployment.

Platform As A Service (PaaS)

Platform as a Service, often simply referred to as PaaS, is a category of cloud computing that provides a platform and environment to allow developers to build applications and services over the internet. PaaS services are hosted in the cloud and accessed by users simply via their web browser.

Python

An interpreted, object-oriented, high-level open source general purpose programming language with dynamic semantics used for many things including rapid application development.

Webhook

A webhook delivers data to other applications as it happens, and is also referred to as a web callback or HTTP push API.

- ***Zigbee***

  Zigbee is an IEEE 802.15.4-based specification for a suite of high-level communication protocols used to create personal area networks with small, low-power digital radios.

Appendix B

B.1 Survey

- IoT Independent Study Survey: tinyurl.com/ycpbpncb

B2. Further Documentation

- Independent Study Weekly Blog: indiestudy2017.nadinelessio.com
- Thesis Blog: dfi2017.nadinelessio.com/category/thesis
- Code Repository: github.com/sharkwheels/thesis

Appendix C

C.1 Research Ethics Board Approval

October 11, 2017

Nick Puckett
Faculty of Liberal Arts & Sciences & School of Interdisciplinary Studies
OCAD University

File No: 101098
Approval Date: October 11, 2017
Expiry Date: October 10, 2018

Dear Nick Puckett, Ms. Nadine Lessio,

The Research Ethics Board has reviewed your application titled 'User testing for the Internet of Absurd Things'. Your application has been approved. You may begin the proposed research. This REB approval, dated October 11, 2017, is valid for one year less a day: October 10, 2018. Your REB number is: 2017-40.

Throughout the duration of this REB approval, all requests for modifications, renewals and serious adverse event reports are submitted via the Research Portal.

Any changes to the research that deviate from the approved application must be reported to the REB using the amendment form available on the Research Portal. REB approval must be issued before the changes can be implemented.

To continue your proposed research beyond October 10, 2018, you must submit a Renewal Form before October 03, 2018. REB approval must be issued before research is continued.

If your research ends on or before October 10, 2018, please submit a Final Report Form to close out REB approval monitoring efforts.

If you have any questions about the REB review & approval process, please contact the Christine Crisol Pineda, Manager, REB secretariat at [redacted]

If you encounter any issues when working in the Research Portal, please contact our system administrator via [redacted]

Sincerely,

[redacted]

Tony Kerr
Chair, Research Ethics Board