Evolving the Ecosystem of Personal Behavioral Data

Jason Stampfer Wiese

September 2015
CMU-HCII-15-108

Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213 USA

Committee:
Jason Hong (Co-Chair)
John Zimmerman (Co-Chair)
Anind Dey
James Landay (Stanford University)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

Funding for this research comes from the Yahoo InMind project, The Stu Card Fellowship, A Google Faculty Research Award, NSF Grant No. DGE-0903659 and ONR N66001-12-C-4196. Any opinions, findings and conclusions or recommendations are those of the authors and do not necessarily reflect those of any of the sponsors.

Keywords: Computer Science, Design Research, Software Systems, Human-Computer Interaction, Human-Centered Computing, Ubiquitous Computing, Context-Aware Computing, User Modeling, Recommender Systems, Personal Information Management, Identity Management, Personalization, Problem Framing, Research Through Design, Personal Data, Unified Personal Data, Small Data, Personal Data Service

Abstract

Personal data is everywhere. The widespread adoption of the Internet, fueled by the proliferation of smartphones and data plans, has resulted in an amazing amount of digital information about each individual. Social interactions (e.g. email, SMS, phone, Skype, Facebook), planning and coordination (e.g. calendars, TripIt, Basecamp, online to do lists), entertainment (e.g. YouTube, iTunes, Netflix, Spotify), and commerce (e.g. online banking, credit card purchases, Amazon, Zappos, eBay) all generate personal data.

This data holds promise for a breadth of new service opportunities to improve people's lives through deep personalization, through tools to manage aspects of their personal wellbeing, and through services that support identity construction. However, there is a broad gap between this vision of leveraging personal data to benefit individuals and the state of personal data today.

This thesis proposes unified personal data as a new framing for engaging with personal data. Through this framing, it synthesizes previous research on personal data and describes a generalized framework for developing applications that depend on personal data, exposing current challenges and issues. Next, it defines a set of design goals to improve the state of personal data systems today. Finally, it contributes Phenom, a software service designed to address the challenges of developing applications that rely on personal data.

Acknowledgements

I am thankful for the advice, support, insights, and feedback from my co-advisors Jason Hong and John Zimmerman. Throughout my time as their student, their patience with me and commitment to me has been steadfast. I am also grateful for the insights of my committee members Anind Dey and James Landay, and of my internship mentors Jacob Biehl, Elizabeth Churchill, A.J. Brush, and Scott Saponas.

The faculty, staff, and students of the Human-Computer Interaction Institute have provided a warm and supportive community of colleagues, collaborators, and dear friends. In particular, I would like to thank Ian Li, Sauvik Das, Derek Lomas, Jenn Marlow, Jenny Olsen, Iris Howley, Scott Davidoff, Eiji Hayashi, Will Odom, and Queenie Kravitz. I am especially grateful to my Friendship House roommates Chris Harrison, Stephen Oney, and Amy Ogan. These years at Carnegie Mellon have been immeasurably better because of the three of you.

Thank you to my family for always believing in me: my parents Evette and Charlie, and my sisters Sarah and Samantha.

Most of all, I want to thank Eliane, my partner in life, who has contributed blood, sweat, and tears to this dissertation.

Table of Contents

List of Figures

List of Tables

1 Introduction

In the last decade, our society has undergone a fundamental shift in day-to-day life starting with the widespread adoption of the Internet and rapidly accelerated by the proliferation of smartphones and data plans. For a large and growing portion of the first world population, an incredible number of people's daily tasks are now mediated by Internet-connected computing technology: social interactions (e.g. email, SMS, phone, Skype, Facebook), planning and coordination (e.g. calendars, TripIt, Basecamp, online to do lists), entertainment (e.g. YouTube, iTunes, Netflix, Spotify), and commerce (e.g. online banking, credit card purchases, Amazon, Zappos, eBay) are all activities that are increasingly digitally mediated. In addition, people are generating increasing amounts of files including documents, media, and contact lists. Fueled by convenience and increased efficiency, the way people do things today is markedly different from the prior decade.

Through this lens, the massive accumulation of data that describes people's behavior in these applications and services is merely a byproduct of this major societal shift: these applications capture who their users communicate with, what their users purchase, and what content they consume. But these large caches of data are hardly a coincidental byproduct: Facebook, Google, Amazon, and Netflix each owe their continued success in large part to the massive stores of personal data they have amassed that describe their users' behaviors. These companies employ their users' data to sell advertising, recommend content, and personalize interfaces. Often, companies use the amassed data while users remain uninvolved. Most users do not understand what data is being used, how it is being used, how they might be at risk, and how they might benefit from applications that use their data.

People are understandably concerned, distrustful, and feel helpless when it comes to their data. Other than withdrawing altogether from our technology-drive society, what choice do they have? As a result, most people have a fairly distanced relationship with the data about them: the typical person has effectively no relationship with their data. The concerned person tries to minimize what is collected, to say "no" whenever offered a choice that lets them still receive service without surrendering their data. Thus, the ecosystem of personal data appears quite dysfunctional: the people who are the subject of that data have limited access to it and try to minimize its existence while companies vie for users so that they can have unrestricted access to the data users will generate in their services.

Simultaneously, there is a sense that this data holds immense value that when combined could unlock an exciting new future of highly personalized, meaningful personal computing experiences. Many applications and services have begun to demonstrate the personalized, holistic, and user-centric potential that individuals' data has to offer. Personal assistants like Google Now, Siri, and Cortana use the data collected within their platforms to suggest contextually relevant information and answer queries. The Nest thermostat adapts to a user's behavior and makes adjustments auto-magically. Gmail's priority inbox feature uses a variety of heuristics like which emails the user reads first and who the user sends emails to in order to guess which emails the user wants to be prioritized.

Yet, these examples feel like they fall short of the real potential of personal data. Researchers motivate their papers with promises for the awesome, intelligent, personalized future of computing. Science fiction envisions personal assistants that understand complex situations[1], learning environments that can relate lessons to our actual life experiences[2], and technology that can automatically assess and treat mental health conditions[3]. With a little imagination, there is the clear potential for technology to support tasks that are difficult for people to do today: Where should I go on vacation? How can I live more sustainably? Who should I room with in college? What thing should I buy to make my life better? What should I do differently to be a better boss/employee/ spouse/parent/friend? A future where technology can help us in these situations seems more plausible than it has ever been before. Following the path to realizing this vision will require major advances across computer science: speech interfaces, machine learning, robotics, sensing hardware, database systems, privacy and security, distributed systems. Furthermore, beyond computer science much of this personalization will require domain-specific knowledge and will likely require advances in those fields as well.

To be able to attempt the advances required to enable this promising future requires engaging with the present-day dysfunctional landscape of personal data characterized above, itself a daunting task. Even worse, beneath the surface of the societal and social issues surrounding personal data is a similarly dysfunctional technological landscape. Science and engineering research answers well-defined

[1] Her (2015)
[2] Star Trek (2009)
[3] Card, O. S. (1985). Ender's game (Vol. 1). St. Martin's Press.

questions, but in the case of personal data, the goal state is ill defined. Making an advance under these conditions first requires specifying a new frame for understanding what could and should be; a vision for the future of personal data.

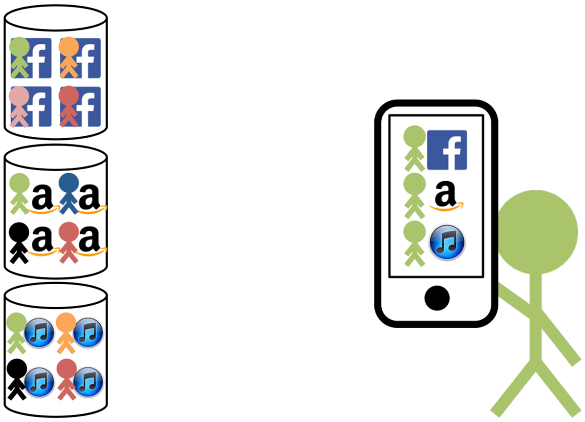


Figure 1: Personal data today is separated across the applications and services where each type of data originated (left). To unlock the full potential of personal data, it should instead be structured to prioritize the coherence of the heterogeneous data around each individual who is the subject of that data (right).

Recent work by Pentland has proposed “a new deal on data,” specifying that users should be the owners of the data that describes their own behavior (Pentland, 2009). Following this theme, Estrin has proposed a vision of “small data” wherein each individual can leverage the traces of data about them in order to build insights about themselves (Estrin, 2014). These visions offer components of an intriguing future: who should own a person’s data (people should own their own data) and what people ought to do with their own data (people should be able to build insights about themselves from their own data).

Building on this recent work, this thesis proposes the framing of **unified personal data** as an opportunity and a goal state for advancing the landscape of personal data. The unified personal data vision claims that an individual’s heterogeneous personal data should be tightly integrated and represented all together on the level of the individual (Figure 1 right), rather than each user’s personal data being disparate, disjoint, and siloed across each of the particular services and devices that an individual uses. This framing of unified personal data as a goal state for personal data signifies an important design contribution that advances many areas of computing.

An entire host of challenges must be overcome to bring about unified personal data. Personal data is siloed within the services and devices where it was collected. Companies independently determine what data to collect, whether or not data can be accessible outside of the service, how long data will be kept, the terms of use for the data, and what format that data can be accessed in. Even if a user has the power to grant a developer access to her data, the challenges continue: bringing data together from multiple sources, doing something to process that data (e.g. machine learning), and applying the data are all a massive undertaking. Furthermore, there is very little structure or support for this process today.

Advancing the state of personal data will require a fundamental shift in the way that personal data is managed. Today, personal data is stored separately by each company that collected it, and then within each application or service it is separated by user. This approach is a natural fit for "big data" analysis: a company can use the data they have amassed across all of their users to gain insights on user behavior.

If the goal is to gain insights about individuals, the current approach is a bad fit. The amount of effort required to participate in the quantified self movement helps to illustrate just how inhibitive the current approach is: to get even a partial view of one's own data requires technical skills, and a fair amount of invested time in order to write the code that brings together data from these disparate sources and do something interesting with that data. While motivated individuals are able to draw together some of their data and even generate their own insights from it[4], these systems tend to be built in an ad hoc fashion (e.g. connecting to specific sets of services, designed to run in specific programming environments). Even for individuals with technical skills, these can be difficult to build on, and for those without technical knowledge most tools that bring together multiple sources of heterogeneous data are out of reach.

Though these observations might seem obvious in retrospect, they were not. In my early days as a doctoral student at Carnegie Mellon, colleagues and I would hypothesize many ideas of the form "I bet if you had [X], [Y], and [Z] data, you could infer [A]." Finally, I tried one, a comparatively simple one: "I bet if you had a person's communication logs you could infer the strength of their relationships with all of their contacts." In fact, we expected it was going to be so simple that the real research contribution would not be the relationship model, but instead the contribution was going to be "inferred relationship strength can be used to set sharing preferences." As chapter 3 details, this was in fact not a simple task, and the result left much to be desired. Furthermore, following up with more data from more sources was not feasible. The resources required to make even some simple additions were too great. Why was this so difficult, so resource intensive? What changes are necessary to improve this state of personal data?

This dissertation seeks answers to those questions by stepping back to take a holistic look at the ecosystem of personal data. To accomplish this, I employ a multidisciplinary human-computer interaction approach, integrating inquiry techniques from both computer science and design to make advances in both disciplines.

[4] See http://blog.stephenwolfram.com/2012/03/the-personal-analytics-of-my-life/ and http://feltron.com/ for two notable examples.

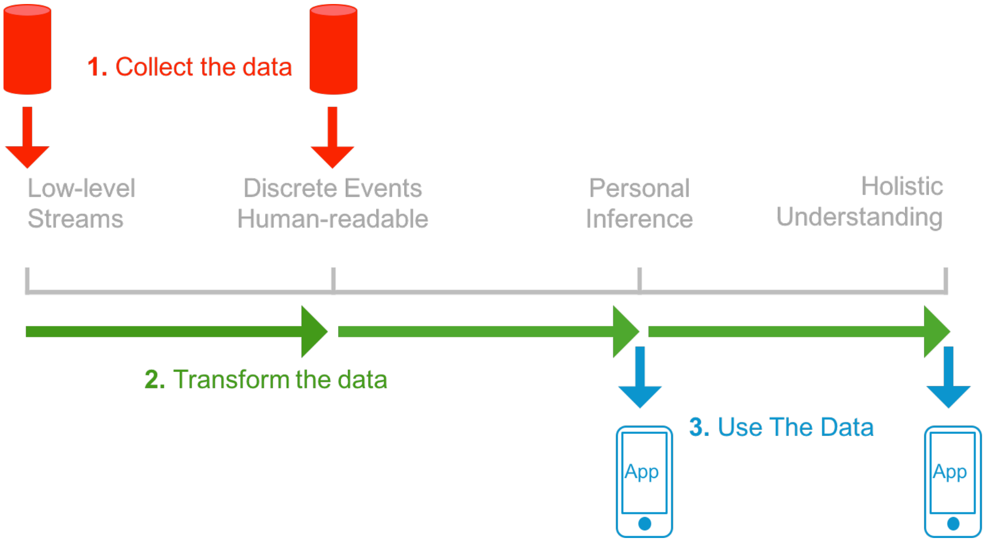


Figure 2: The conceptual framework of the steps involved in developing software that depends on personal data: data is collected from some number of data sources, the collected data is transformed into the appropriate level of abstraction or meaning for the target application, and the data is incorporated in that target application.

Another important outcome of this dissertation is a conceptual framework that describes the general process that is required to develop software that depends on personal data (see Figure 2). The conceptual framework consists of two components. The first component is a continuum of personal data from very low-level (e.g. raw sensor data) to very high level (e.g. is the user experiencing major depression?). The second component is a set of three steps that are required to develop applications that depend on personal data. First, capture or collect the necessary personal data. Second, transform the collected personal data into the required level of abstraction or meaning for how it will be used. Third, apply the transformed data to the target application. While these three steps may seem simple on the surface, chapter 4 highlights a variety of challenges that highlight the complexity of engaging in this process. The conceptual framework is a useful tool for engaging a conversation around the process of developing applications with personal data, and exposes some critical issues for working with personal data, which limits what is reasonably possible for researchers and application developers to accomplish today. By distilling these challenges, this dissertation proposes a set of goals for achieving the vision of unified personal data.

Finally, this dissertation describes the design and implementation of Phenom, a service designed to make progress towards these goals by modularizing the process of working with personal data. Phenom dramatically reduces the effort that is required for a developer to incorporate personal data in an application. By employing a semantic data store and focusing on an integrated, flexible, and modular approach to handling personal data, Phenom radically changes how easy it is for a developer to program applications that depend on personal data, demonstrating a first step towards the vision of unified personal data.

1.1 Research Contributions

This dissertation offers the following technical and design contributions to HCI:

1. A proposal for unified personal data; a reframing of many HCI challenges, human needs, and technical opportunities that can all be advanced through a more holistic view of all of the individual data amassing around people as their personal data that should be brought together and structured such that it works for them and remains under their control.
2. The notion of personal data as a continuum, and a conceptual framework that unpacks the implicit process involved in working with personal data.
3. A set of design goals for improving the ecosystem of personal data.
4. The design of Phenom: a service that supports software development with personal data. Phenom modularizes the collection, interconnection, processing, and querying of personal data to solve a key set of challenges involved in developing applications that use personal data.
5. The implementation of a proof of concept of Phenom, which demonstrates its viability and utility as a personal data service.

1.2 Dissertation Overview

Chapter 2 highlights many research domains that have incorporated personal data (often implicitly) in their work, including a variety of my own projects across those domains. Personal data underlies multiple threads of research, and in many cases progress in those domains appears stifled because of the limitations of the state of personal data today. Despite these limitations encountered and the commonalities across fields, it appears that no efforts have been made towards connecting these domains and engaging holistic thinking on the ecosystem of personal data. The vision of unified personal data offers a new frame for viewing people's collections of personal data, one that offers benefits to the various research domains that have helped to define personal data and that employ it to offer an advance.

Chapter 3 offers a detailed case study of the process and findings of my own research to connect communication behavior to social sharing preferences. The practical challenges faced in that work highlight many of the shortcomings of engaging in research with personal data.

Chapter 4 synthesizes the landscape of personal data mapped out in the previous two chapters to engage personal data from a holistic perspective. This synthesis produces a set of general steps that are required for making use of personal data: collecting the data, making higher-level sense of the data, and applying the processed data to an application. It identifies a set of challenges and issues that inhibit work with personal data, using the framework to illustrate these challenges. Finally, it proposes a set of design goals that offer an agenda for improving the personal data ecosystem.

Chapter 5 describes Phenom, a service that I developed to support the process of developing applications based on the vision of unified personal data. Phenom addresses some of the most prominent challenges of working with personal data by offering a modular approach that separates the steps of the personal data development process. Phenom unifies personal data on the level of the individual, supporting rich interconnections in the data and reuse of components across completely independent applications.

Chapter 6 concludes the dissertation with an eye towards the future of personal data research.

2 Situating Unified Personal Data within the Landscape of Research that Leverages Personal Data

Traditionally, scientific and engineering research both focus on answering well-formed research questions; the mantra "what is your research question?" is universally familiar and relevant. And yet sometimes identifying what the question should be is itself a major research challenge. These situations can be daunting from the perspective of science and engineering research. Design research, in contrast, focuses on the search for a question that is worth answering. Design researchers refer to this as framing a goal state that supports an advance toward a preferred state of the world. It asks about relevance and improvement to the world as the most critical criteria. To understand this approach requires a basic understanding of the concept of design thinking.

When describing design, Herbert Simon wrote "To design is to devise courses of action aimed at changing existing situations into preferred ones," (Simon, 1969). This places design thinking in a subjective space with a focus on what might be better for the world. Rittel and Webber advanced this idea with their work on "Wicked Problems," large-scale social issues like urban crime, that are not easily addressed through science or engineering inquiry, but that are approachable through design thinking (Rittel & Webber, 1973). These challenges cannot be accurately modeled (and thus cannot be solved by scientific or engineering methods alone) because of the conflicting perspectives of the stakeholders involved. The complex open system of multiple stakeholders with conflicting goals and innumerable possible solutions described by Rittel and Weber are applicable when considering the ecosystem of personal data[5]. They describe how design thinking makes advances on these types of problems by proposing solutions that offer a unique framing of the problem to be solved, and that it is only through the articulation of a solution that researchers can even know the problem they want to address.

Speaking on how design functions as a reflective practice, Donald Schön argues for the importance of framing problems, and specifically that the process of design thinking is about picking a specific frame to engage (Schön, 1983). Design thinkers employ a process of reflecting-in-action and reflecting on action as they generate and assess many possible frames (i.e. the futures they might want to achieve). More recently, Kee Dorst considered different forms of reasoning (deduction, induction, and abduction) to position design thinking with respect to the kinds of reasoning frequently encountered in science and engineering (Dorst, 2011). Dorst builds on Schön's concept of framing, identifying that framing is a form of perspective-taking, with many different perspectives possible. He discusses how designers systematically cycle through many possible desired outcomes in order to discover a path forward that can resolve a problematic situation.

This thesis proposes unified personal data as a mechanism to unlock the promised future of personalized computing experiences. The articulation of this goal (a preferred future) and this mechanism (unified personal data) is a match to Dorst's conception of framing in design thinking. Framing the vision and the opportunity of unified personal data is a core contribution of this dissertation that unfolds through chapters 2, 3, and 4. The messy and iterative nature of this process does not easily lend itself to the serial format of this dissertation, and this chapter relies in part on the overview offered in Chapter 1 to offer a structure to this framing.

To begin this framing, this chapter provides a broad survey of research domains that have led to the conception of unified personal data as a solution. Across computing research, researchers have been implicitly examining the need and benefit of personal data from a variety of perspectives over many years. Despite the common interest in personal data shared by each of these domains, and perhaps because of the lack of a broader cross-discipline unified personal data framing, the contributions between them have been mostly disconnected: progress in one personal-data-focused sub-discipline typically has little impact on the work in other sub-disciplines. Treating personal data holistically as a research community rather than as a disconnected (or loosely connected) combination of research topics may provide the long-term support necessary to push forward the evolution of personal data.

The bulk of this chapter highlights each of these domains and connections between them (see Figure 3), focusing on challenges that each domain has encountered

[5] see chapter 4 for a discussion of some of the stakeholders for personal data

related to personal data. Highlighting these challenges serves several purposes in this dissertation. First, understanding the personal-data-related challenges in each field offers multiple perspectives that contribute to the problem framing. Additionally, understanding how each of these fields relates to personal data offers the ability to contextualize advances made in the space of personal data with respect to each discipline.

The end of this chapter offers an overview of various personal-data-related research projects that I have worked on. While many of these projects also have separate research contributions of their own, in the context of this dissertation these projects can be seen as design probes. Through this lens, each of these projects has offered a different perspective towards framing the opportunity of unified personal data, which I have synthesized in chapter 4.

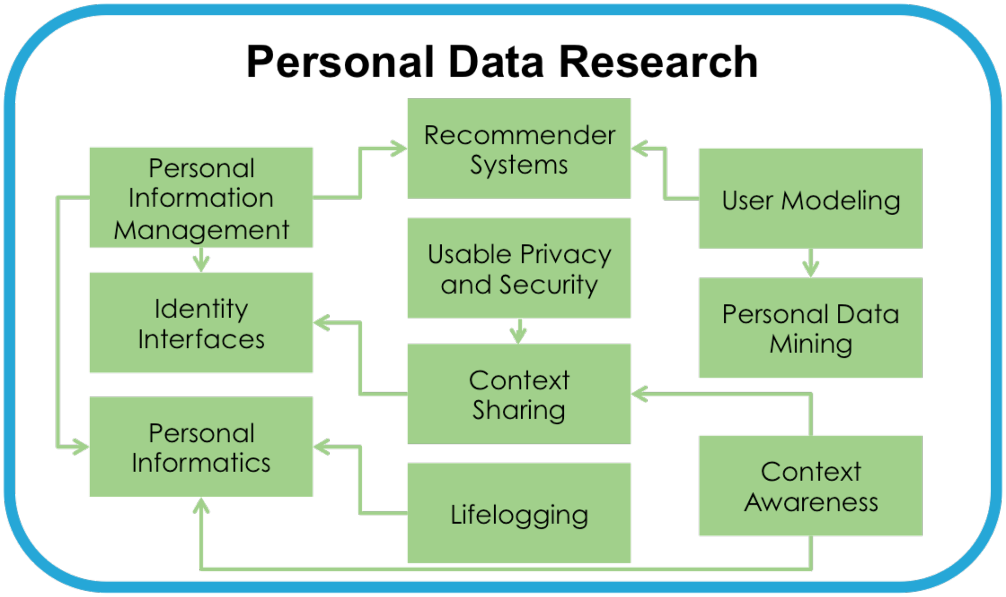


Figure 3. A map highlighting research domains across HCI that generate, make use of, or investigate personal data, and the interconnections between them.

2.1 Personal Information Management

The research area of Personal Information Management studies how people acquire, organize, maintain, and retrieve the many different types of information that they use in their day-to-day lives. In many ways, the field draws inspiration from Vannevar Bush's seminal paper "As We May Think" (Bush, 1945). Bush describes the hypothetical Memex, a microfilm-based system for storing and retrieving the multitude of information that people handle throughout their lives. The first work actually referencing PIM appeared in the 1980s[6], evolving as a research area around the same time as HCI.

Jones' survey chapter of PIM (W. Jones, 2007) describes three "senses" of Personal Information:

1. The information people keep for their own personal use (e.g. contact lists, financial records, time-tracking logs)
2. Information about a person but possibly kept by and under the control of others. (e.g. invoices from purchases made on Amazon, electronic medical records)
3. Information experienced by a person even if this information remains outside a person's control. (e.g. the news stories a person views online, the items a person browses on Amazon but does not buy)

As Jones identifies, the study of PIM primarily focuses on the first sense, but acknowledges the relevance of the second and third as well. Put another way, PIM is primarily a study of how humans use the tools available to them in order to store information that they would like to access in the future, including contacts (Whittaker, Jones, & Terveen, 2002), calendar appointments (Starner, Snoeck, Wong, & McGuire, 2004), to do items (Bellotti, Dalal, Good, Flynn, & Bobrow, 2004), email (Ducheneaut & Bellotti, 2001), and the myriad pieces of unstructured information that we collect (Bernstein, Van Kleek, Karger, & Schraefel, 2008) and the possibility of finding the structure in that data (Chang et al., 2013). Other recent work in PIM has explored the concept of unifying different types of heterogeneous personal information (Karger & Jones, 2006). The problems cited in this work offer support for the design goals specified in chapter 4.

PIM research offers an important component to the broader context of personal data research. First, PIM research provides an important examination of the interface between the user and the storage and retrieval of their personal data. In PIM research the specific interaction is about users explicitly storing pieces of their personal data for the purposes of retrieving it themselves later: there is no additional processing happening on the data while it is stored. However, PIM research can contribute insights into how best to collect hand-labeled ground truth data. This is particularly important for training models on personal data. With personal data the ground truth labeling task is more constrained than in the general case because with personal data models the person labeling the data typically needs to be the person who that data is about.

Today, there is much more personal information for users to manage than ever before. The volume of data has grown both because there are more types of digital personal information (e.g. media collections, shopping behavior, and taxes), and also

[6] See (W. Jones, 2007) for a helpful explanation of the study of PIM that offers some connections between Vannevar Bush's "As We May Think" and the modern study of PIM starting in the 1980s.

because there are more data in existing channels (e.g. growing histories of email interaction, and more and more email each year). The result of this is that people have more personal data than ever to keep track of and many of the "things" are stored on different third-party services. A major challenge for PIM is to provide people with easy and relevant access to their data. This dissertation focuses on offering new ways of connecting relevant data together, even across different third-party services, which is one component of the challenge facing PIM. Furthermore, this dissertation proposes a unified approach to track and store people's interactions with their data, which is another aspect of PIM.

2.2 User Modeling

Research related to the goal of creating software that responds and reacts to characteristics of the user first appeared in the 1970s in several different application domains. Some of this work had a goal of creating intelligent tutoring systems that would use the student's behavior within a tutoring system to personalize the software's behavior for that student (Burton & Brown, 1979). Other work was focused on dialog systems that would tell different things to different users based on what the software could infer the user already knew (Allen, 1979; Cohen & Perrault, 1979; Perrault, Allen, & Cohen, 1978), or by using characteristics of the user to guess what the user's intention was (Rich, 1979a, 1979b). In these early modeling systems, the modeling components were not distinct from the rest of the application, but as the field grew user modeling systems were made more modular. The first wave of modularization was in the form of shell applications that would be a part of the application. Fueled by the advent of the internet, the next wave was server-based user models that could support multiple distributed client applications (Kobsa, 2001).

User modeling has grown with the rest of computing to include modeling in mobile and ubiquitous contexts. The info-bead user modeling approach (Dim, Kuflik, & Reinhartz-Berger, 2015) is one such system, which represents different pieces of user context as *info-beads* that can be connected together through *info-links* to form *info-pendants*. The complete collection of *info-beads* and *info-pendants* can be combined to form user models and group models, which can be used to personalized specific systems. This modular approach can enable the reuse of *info-beads* and *info-pendants* in different deployments of the *info-bead* user modeling approach. Though the architecture and the approach to modularity in the *info-bead* approach are different from the implementation of Phenom described in chapter 5, the value placed on modular components that can be used across different applications and Phenom's bots (see section 5.1) share a common inspiration.

User modeling has also begun to expand to include the idea that user data can come from across the user's lifetime. PortMe (Kay & Kummerfeld, 2010) is a user model framework that is designed to support models that are based on the user's lifetime of personal data. PortMe provides an interface so that users can view and interact with details of user models that are based on their own data, and relies on the PersonisAD user model server (Assad, Carmichael, Kay, & Kummerfeld, 2007) for the underlying user model representation. This concept of holistically thinking about personal data that spans a person's entire life is core to engaging some of the fundamental issues with personal data, and it informs the design goals in chapter 4.

Where PIM research was mostly focused on a user-centric perspective of explicitly-collected personal data, User Modeling is different in many ways. Instead User Modeling takes a primarily system-centric perspective, focusing instead on developing domain-specific models based on user behavior.

User modeling brings to personal data research a demonstrated process for making end-to-end systems that leverage a user's behavior to model a specific item, and incorporate that model into the application. Though these tend to be closed systems (i.e. data is collected from, modeled by, and applied to a single application), work in user modeling represents concrete examples of leveraging personal data to create models and apply those models to specific applications.

One important challenge facing work in user modeling is deploying the models. This is essential for being able to build on the models (either in research or in commercial contexts), and also for understanding the real-world validity of the models beyond the more controlled environment of a traditional study. Phenom, the system described in this dissertation, offers an architecture that supports deploying models that depend on a user's personal data.

2.3 Recommender Systems

Recommender systems emerged in the early 1990s, growing out of user modeling into a space that was more directly focused on user experience. Specifically, early recommender systems set out to address a clear problem: as more and more people started using the internet, the amount of content was growing considerably and information overload was setting in (Konstan & Riedl, 2012). Tapestry (Goldberg, Nichols, Oki, & Terry, 1992), the first recommender system, targeted email overload. That work also introduced the phrase collaborative filtering, which has been an essential technique used by many recommender systems. In the twenty years since, recommender systems have grown massively and are deployed across many commercial systems offering personalized recommendations and predictions across many different domains from media consumption (i.e. news, movies, books) to recommending social relationships (i.e. dating websites, following people on Twitter).

For personal data research, recommender systems (similar to user modeling) demonstrate the potential for closed systems to leverage personal data to enable real-world personalization. One interesting dimension that recommender systems bring to the conversation around personal data is the concept of collaborative filtering: dynamically using labeled data captured from a breadth of users to predict another user's behavior or interests. This also draws together one aspect of the research on personal information management: the manual labeling of data by users. Recommender systems, particularly those based on collaborative filtering, demonstrate interactions where users can provide labels for their personal data and receive direct benefits in turn for that labeling.

A key challenge facing recommender systems is to integrate different kinds of recommender approaches including content-based approaches (i.e. using information about the content), collaborative approaches (i.e. collaborative filtering), and contextual approaches (i.e. situational information about the user) (Konstan & Riedl, 2012). This goal is an important component of the personal data vision as well. In most recommender systems, the personal data that is used by the system is personal data that was generated within the system (e.g. ratings, viewing behavior, sharing behavior). However, to fully realize the potential of contextual approaches, recommender systems will need to start to depend on data from outside of their systems as well. Work on context-aware recommender systems represents a movement in that direction (Abbar, Bouzeghoub, & Lopez, 2009; Adomavicius & Tuzhilin, 2011). Most context-aware recommender systems to date focus on immediate context, like the time of day or location (Matyas & Schlieder, 2009; Oku, Kotera, & Sumiya, 2010). Moving forward, recommender systems will need to broaden their focus to include a more holistic view of the user's data and begin to make use of logs of personal data that show routines, changes in behavior, and trends over time (Bobadilla, Ortega, Hernando, & Gutiérrez, 2013). As a result, the work of this dissertation is of direct interest to recommender systems.

2.4 Lifelogging

The research topic of lifelogging first emerged in the mid-1990s and in many ways started as a combination PIM, multimedia, and ubiquitous computing, sharing the same basic PIM inspiration of Bush's Memex (Bush, 1945), but dramatically increasing the amount and kinds of data that might be captured in such a system (i.e. location, video, workstation logging) (Lamming et al., 1994). Other early work in the topic of lifelogging includes Lifestreams, which proposed a new metaphor for dynamically organizing a person's data (Freeman & Fertig, 1995; Freeman & Gelernter, 1996). While the Lifestreams work was particularly focused on documents, it sets out a list of six observations that motivated their work, and remain relevant today:

1. Storage should be transparent
2. Directories are inadequate as an organizing device
3. Archiving should be automatic
4. The system should summarize multiple related documents in a concise overview
5. Computers should make reminders convenient
6. Personal data should be accessible everywhere

With a small amount of interpretation, when placed in the context of today's landscape of personal data many of these observations remain applicable and the spirit of the goals expressed through those observations have not been met.

Unified on the concept of "total capture" with a primary goal of serving as a memory aid, a variety of lifelogging systems appeared to focus on capturing as much data as possible about individuals' behavior (Hodges et al., 2006; Hori & Aizawa, 2003) and providing usable ways of accessing that data (Adar, Karger, & Stein, 1999; Dumais et al., 2003; Gemmell, Bell, & Lueder, 2006; Gemmell, Bell, Lueder, Drucker, & Wong, 2002). This style of lifelogging work attracted criticism and lost favor in the research world when it became apparent that the collection of these huge archives of disparate data did not lead to compelling applications (Sellen & Whittaker, 2010). Despite these criticisms, a number of commercial systems have emerged in recent years that enable the "capture" portion of lifelogging (e.g. Narrative camera[7], Saga mobile app[8]). One notable exception to this criticism is in more specific populations where there has been demonstrated value in lifelogging, for example in people with memory impairment (Browne et al., 2011; Lee & Dey, 2008), or serving as a tool for helping and understanding children with autism (Marcu, Dey, & Kiesler, 2012).

In many ways, lifelogging is an attempt at finding a solution (developing the technology) without fully understanding the problem (validating the application area). lifelogging as a research area communicates an underlying hunch that there must be value in the data that characterizes our lives. However, it lacks a clear need that collecting this data will fill.

Lifelogging is a different perspective on personal data: the idea that individuals will drive the collection of their own personal data, perhaps without a specific purpose in mind. This contrasts against User Modeling and Recommender Systems where the user may not even know that data is being captured and used by the system. lifelogging research is in some ways similar to PIM: a user-centric focus on the collection and retrieval of personal data. However, where lifelogging and PIM differ is in the volume and use of the data: PIM collects a comparatively small amount of data that the user expects to need later, where lifelogging collects as much data as possible, typically without a specific use in mind.

Lifelogging research faces a difficult duality. On one hand, there is a general hunch that there is value contained within personal data, and it is impossible to harness that value (or even to understand what that value is) without first collecting large amounts of data. On the other hand, lifelogging is a cautionary tale of finding a solution without knowing what problem it solves. One way for personal data to address these issues is by making it easier for developers and researchers to experiment with different ways of finding value in personal data. Lowering the barrier to entry is likely to surface many more ideas and enable a real-world validation of their utility. This dissertation explores opportunities for reducing the burden on developers for carrying out these steps.

[7] http://getnarrative.com/

[8] http://www.getsaga.com/

2.5 Context-Aware Computing

The field of context-aware computing is a research domain that was established in parallel with and closely related to ubiquitous computing in the early 1990's to develop computing systems that could capture, process, and react to a person's immediate context (Schilit, Adams, & Want, 1994). A widely used definition of context comes from (A. K. Dey, 2001): "Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." Dey also offers a definition of context-aware: "A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task." These definitions are quite general, and Dourish (2004) argues that full context-awareness is intractable, as the relevance of context surely changes from moment to moment.

The Context Toolkit is a very prominent piece of work in this domain (A. Dey, Salber, & Abowd, 2001). The Context Toolkit was a response to the problem that developing context-aware applications was far too difficult because the components of a context-aware system were not modular enough to facilitate reuse. This observation and the resulting requirements for dealing with context are in many ways analogous to the observations made in this dissertation with respect to personal data: developing with personal data is also very difficult, and one of the sources of that difficulty is also the lack of modularity and the exposure of too much complexity.[9]

Context-aware computing is largely motivated by vision proposed by Weiser's "Sal" story (Weiser, 1991) where as a user moves through her day, technology is seamlessly integrated into her day, to the point where the technology becomes unremarkable and invisible in use. Despite the fact that an important component of realizing this vision will require longer-term knowledge, for example knowledge of a person's routine behavior (Tolmie, Pycock, Diggins, Maclean, & Karsenty, 2002), context-aware computing has traditionally focused on immediate context based on data that was collected in the short term, often based only on instantaneous sensor readings. This represents an entire category of data that describes information about a person, and thus fits into the category of personal data.

From the perspective of personal data, context-aware computing research contributes technical solutions for transforming sensor data into more meaningful personal data. In this way, context-aware computing makes possible the automatic collection of new kinds of personal data, or of improving the accuracy or coverage of that data.

[9] There is far too much work in the realm of context-aware computing to describe in this chapter, however (Baldauf, Dustdar, & Rosenberg, 2007) and (Chen & Kotz, 2000) offer surveys of the field.

One important problem that the field of context awareness faces is the challenge of reconciling instantaneous data, which captures part of a person's context in the moment, with the much more expansive logs of heterogeneous data that provide the information necessary to correctly interpret that instantaneous context. To engage this challenge, context-awareness research will either need to access and interpret those disparate logs of heterogeneous personal data themselves, or build on the work of others who have done this. The framing of unified personal data includes the concept of collecting long-term logs of data about the user, simplifying this challenge.

2.6 Personal Informatics and Quantified Self

Personal informatics is defined as a class of systems that help people collect personally relevant information for the purpose of self-reflection and gaining self-knowledge (Li, Dey, & Forlizzi, 2010). Personal informatics has emerged as a research area simultaneous with the widespread adoption of smartphones and increased consumer interest in fitness-related wearables. At a high level, personal informatics involves two major steps: collecting data and reflecting on that data. On the collection side, personal informatics has its roots in PIM, lifelogging, context-aware computing. On the reflection side, personal informatics has its roots in information visualization (Pousman, Stasko, & Mateas, 2007). One example of an early personal informatics system that combines these steps is the Ubifit Garden (Consolvo et al., 2008), which used mobile phones to collect and visualize physical activity information.

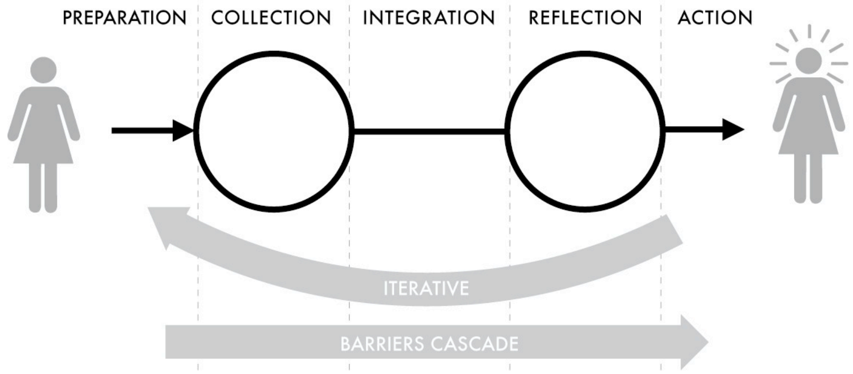


Figure 4: Stage-Based Model of Personal Informatics Systems (Li et al., 2010)

Li, et al. (2010) proposed a stage-based model of personal informatics systems (see Figure 4) targeted toward behavior change that identified five stages: preparation, collection, integration, reflection, and action. This work highlighted two important features that the model emphasized: this process is iterative, and that barriers in earlier stages of the process (e.g. difficulty collecting data, difficulty integrating data from different sources) cascade to impact later stages, perhaps even making those later stages impossible.

The quantified self movement, a group of people interested in self-tracking, has appeared and has gained some traction across the world over the last several years, with a small but loyal following. While a few notable individuals have attracted some press for reporting on their own findings from examining their own personal data (e.g. Stephen Wolfram's "The Personal Analytics of My Life"[10] and Nicholas Felton's "Feltron Annual Report"[11]), for the most part examining one's own data remains a fairly uncommon activity that requires the user to have logged her own data and also have the knowledge, skills, and motivation to turn that raw data into something meaningful or consumable.

Research in personal informatics, and the growing quantified self movement demonstrate that the process of collecting personal data is a challenge, so much so that it inhibits what information can be collected even if the data already exists. Recently, there has been a sort of call to arms towards data liberation throughout the community. For example, Deborah Estrin's concept of small data envisions a future where individuals have access to and control over all of their data own data, for use however they choose (Estrin, 2014). For personal informatics to continue to grow, it needs to be easier for people to bring together and synthesize their own data, and to do this from more sources. The vision of unified personal data offers one way of accomplishing this.

2.7 Computational Social Science and Data Mining To Understand Human Behavior

With the widespread adoption of cell phones, it has become possible to collect large-scale datasets that capture the dynamics of human movement and social behavior over large populations and/or long periods of time. A major differentiator in this work is whether or not the researchers have access to individuals in the population.

Typically, if the researchers have access to individuals in the population, it is because the researchers have recruited the population directly and have collected the data themselves. One early example of this style of work is the reality mining project (Eagle & Pentland, 2006). In this work, 9 months of data was collected from 100 participants that included call logs, Bluetooth proximity to other devices in the study, cell tower IDs (providing rough location), and application usage. Through the collected dataset, the researchers were able to examine individual routines, dyadic behavior across individuals, and organizational behaviors across the entire dataset. Since the time that this original data was collected, subsequent studies have been conducted to examine routines within families (Davidoff, Ziebart, Zimmerman, & Dey, 2011), location dynamics of a heterogeneous sample (Kiukkonen, Blom, Dousse, Gatica-Perez, & Laurila, 2010), shifting behaviors in a residential community (Aharony, Pan, Ip, Khayal, & Pentland, 2011), and mental health in the students of a college class (R. Wang et al., 2014). By engaging in these data collections, researchers have the opportunity to collect participant responses focused

[10] http://blog.stephenwolfram.com/2012/03/the-personal-analytics-of-my-life/
[11] http://feltron.com/

on a particular research question, which can then be used as ground truth when developing a model based on the automatically sensed data. These data collections each represent a massive effort on the part of the data collector.

An alternative approach taken by many researchers in the space of computational social science is to obtain and analyze a dataset that already exists, for example (Conti, Passarella, & Pezzoni, 2011; González, Hidalgo, & Barabási, 2008; Onnela et al., 2007; D. Wang, Pedreschi, Song, Giannotti, & Barabasi, 2011). The perspective taken by this work mimics the broader "big data" trend: using anonymized logs, typically from a single data source, to build some insight or observe a broad scale phenomenon (Lazer et al., 2009). However, there are some major drawbacks to this approach, many of which stem from the lack of access that researchers have to the individuals whose data comprise the dataset. In particular, this lack of access means that the data is often homogeneous (e.g. only call log data no other data) because connecting multiple sources of data would require knowing who those individuals are, linking data together, and perhaps even requesting permission to do this work. Additionally, this means that this style of work also does not have access to explicitly-provided data (e.g. survey responses), only implicitly-provided data (e.g. phone call logs from a telecommunications provider). Even if individuals wanted to provide additional information to researchers, there is no mechanism by which to provide that information. As a result, this leads researchers to define proxies based on the data that are used to represent the desired data. However, these proxies are not necessarily validated before use, which can lead to systematic problems when interpreting the data (Wiese, Min, Hong, & Zimmerman, 2015).

These challenges (lacking the ground truth and real-world understanding of what these data actually represent) limits the conclusions that can be drawn from this kind of research. The ability to have anonymized unified data with useful ground truth labels from a large population could have a massive impact on the world. This kind of data could produce important scientific results, and also build insights about the population that can affect city planning, health, and public policy. The challenge of bringing this data together today, even for a single individual, inhibits this progress. The vision of unified personal data described in this dissertation represents an important step towards these goals by bringing together a user's heterogeneous personal data.

2.8 Identity Interfaces: Virtual Possessions and Self-Reflection

As technology has become increasingly integrated into people's everyday lives, many aspects of everyday life that were previously well established in the physical world have begun to bridge into a hybrid physical/digital space, or even a fully digital space. The effects of this shift are incredibly broad and far reaching, and have changed the way that people communicate, collaborate, create, consume, and collect. One important effect of this shift is the transition of many different kinds of possessions that were previously physical possessions into virtual possessions (Odom, Zimmerman, & Forlizzi, 2010). This shift is significant in part because people's possessions both reflect and contribute to their identities (R. W. Belk, 1988). In contrast to material possessions, virtual possessions are placeless, spaceless, and formless (Odom, Zimmerman, & Forlizzi, 2014). These qualities affect the circumstances under which people manage their possessions including the process of curating and archiving their possessions (Kaye et al., 2006), their process of spending time with and reflecting on their virtual possessions (Odom et al., 2010), and the legacy that they leave through their possessions (Gulotta, Odom, Forlizzi, & Faste, 2013).

Research on virtual possessions stands distinct from that in personal informatics, though they are in some ways related. Personal informatics is a user-driven goal-oriented process for collecting personal data (often through explicit action) that describes a user's own behavior. Virtual possessions is also focused on users interacting with their own data, but the user's motivation for interacting with this data and the provenance of this data is a much more fluid component of the user's life: even without explicitly collecting virtual possessions, people have them and interact with them on a regular basis.

The metadata that captures people's interaction with their virtual possessions, and in many cases the virtual possession itself, are all personal data. Furthermore, this data could be used as a component of a personal informatics system. As such, many of the challenges, research questions, and exploratory systems within virtual possessions and personal informatics inform each other and the study of personal data at large. For example, the finding that fragmented virtual possessions are problematic for end users (i.e. that they are stored in different non-compatible applications and services) (Odom et al., 2014) is an important issue for personal data at large. Similarly, the importance of and challenges with leaving a digital legacy (Gulotta et al., 2013) is an important point of consideration for the whole of an individual's personal data archives, even the aspects of that archive that might not be considered "virtual possessions.

Research on identity interfaces is an examination of the way people relate to the personal data that has become an integral part of their everyday lives. Having personal data in a digital format, when contrasted with the previous era where this data was either a material object or did not exist affords a different way of interacting with that data.

Fragmented personal data is a major challenge for identity interfaces. While people present fragmented identities to different social groups (Farnham & Churchill, 2011), the fragmentation of personal data is service-based, not identity-based. Thus, an essential component of moving identity interfaces forward is to bring together service-fragmented data, which will enable research in this domain to continue. Unified personal data offers one solution to this very issue.

2.9 Sharing Context

The widespread adoption of the Internet over the last twenty years has brought the general public the ability to digitally share personal data socially with other people. One major reason to share personal data is to facilitate awareness across colleagues, family members, and close friends. Sideshow (Cadiz, Venolia, & Jancke, 2002), Community Bar (McEwan & Greenberg, 2005), MyVine (Fogarty, Lai, & Christensen, 2004), and ConNexus (J. Tang et al., 2001) collected some awareness information, such as IM status and calendar, and automatically shared that information with contacts in a side bar interface on the desktop. Awarenex (J. Tang et al., 2001), ContextContacts (Oulasvirta, Raento, & Tiitta, 2005) and Connecto (Barkhuus et al., 2008) are all mobile awareness systems with various representations of location and other data, such as calendar information, ring tone profile, and Bluetooth neighbors.

Location is one type of personal data sharing that has been the subject of a great deal of research. While early location sharing was focused on instrumenting an office or workplace (Want, Hopper, Falcão, & Gibbons, 1992), subsequent studies explored location sharing with colleagues and friends IMBuddy (Hsieh, Tang, Low, & Hong, 2007), or with family Whereabouts Clock (Brown et al., 2007). Tang et al. explored location sharing from the perspective of the motivation behind sharing location data, focusing on the difference between purpose-driven sharing and social-driven sharing (K. P. Tang, Lin, Hong, Siewiorek, & Sadeh, 2010). They found that where purpose-driven sharing typically focuses on an exact location, social-driven location sharing was more likely to favor semantic place names to specific geographic location.

Context sharing brings the technical contributions of context-aware computing toward a user-centric space. Research on context sharing offers insights into mechanisms for sharing personal data, and how people interact with that data are important dimensions of the broader domain of personal data.

Context sharing depends on bringing together the information required to do a contextual inference and making that inference. The challenges of doing both of these steps are major, and this challenge inhibits more complex context sharing scenarios (e.g. the "in-common" sharing scenarios described in (Wiese, Kelley, et al., 2011)). To enable these more complex context-sharing scenarios that depend on multiple pieces of context requires a significant development effort if developers cannot easily build on context inferences that have been developed by others. In this dissertation, Phenom offers an architecture that supports the integration and reuse of many different kinds of abstractions (including inferences) that could be made on personal data.

2.10 Privacy

Privacy is perhaps the most dominant topic when it comes to collecting, using, and sharing personal data. Academic discussions about data protection and personal privacy date back to the late 1960s and early 1970s and have expanded considerably in scope since then. A major challenge in privacy research, and in designing privacy-sensitive systems, is that expectations and perceptions of privacy co-evolve with technology, (Iachello & Hong, 2007) and vary across individuals whose opinions may also change over time (M. S. Ackerman, Cranor, & Reagle, 1999; Westin, 2001). Thus, protecting users' privacy is both a moving target, and also must allow for some dimension of control across individuals. Furthermore, privacy is often viewed as a tradeoff, for example trading off the risks and benefits of disclosing some information, or the tradeoff between privacy and the public interest (Iachello & Hong, 2007). The disclosure of personal data can offer benefits (either tangible or intangible) for people who disclose the data and also for the companies that hold the data, but can also be costly for either or both parties (Brandimarte & Acquisti, 2012).

Personal data privacy has been a particularly active topic in recent public discourse in large part because of the exposure of the mass-data-collection of the NSA's PRISM program[12], but also because of discomfort caused by behavioral advertising and a rash of recent data breaches. A major concern within this space is that even in the cases where individuals are explicitly paying attention to the permissions that they are granting to the applications and services that they use, they often do not fully understand the permissions that they are approving (Kelley et al., 2012).

One way of thinking about user-centric personal data privacy research is through the following categories:

- Understanding the potential privacy risks of disclosing personal data, especially cases where disclosing some data inadvertently leaks other data (e.g. (Acquisti & Gross, 2009))
- Understanding the potential benefits of disclosing personal data (e.g. (Lindqvist, Cranshaw, Wiese, Hong, & Zimmerman, 2011)
- Helping users to understand the meaning behind different privacy options and the tradeoffs of granting or denying access to different kinds of data (e.g. (Kelley, Bresee, Cranor, & Reeder, 2009))
- Providing users with interfaces that enable them to easily express their privacy preferences (e.g. (Klemperer et al., 2012))
- Guaranteeing enforcement of a user's privacy preferences (e.g. (Yang, Yessenov, & Solar-Lezama, 2012))

[12] http://www.washingtonpost.com/investigations/us-intelligence-mining-data-from-nine-us-internet-companies-in-broad-secret-program/2013/06/06/3a0c0da8-cebf-11e2-8845-d970ccb04497_story.html

Privacy is an extremely important, and also extremely challenging, topic in user-centric research on personal data, and must be an integral part of ongoing research on personal data.

One major challenge towards implementing usable privacy controls is the lack of continuity in setting those controls. Users are forced to specify privacy settings in a fragmented way, specifying these preferences per-application. While in some cases this granular control might be desirable, in many cases users would benefit from a simpler, unified interface for specifying these controls. The vision of unified personal data offers the possibility of this kind of unified interfaces for considering privacy concerns and specifying privacy preferences in a unified, coherent way.

2.11 Research Examples

The previous sections have offered brief overviews of the breadth of research sub-disciplines that contribute to a broader understanding of personal data. This section highlights research projects related to personal data that I have engaged in across a variety of these research areas. These projects demonstrate in more detail how research in some of these different domains connects to personal data. Additionally, these projects have served as research probes that have greatly contributed to my task of framing the opportunity of unified personal data.

2.11.1 Personal Information Management: The Contact List Name Field

I examined contact lists with an initial goal of leveraging the structured data within the contact list entries of users' smartphones to infer aspects of the user's relationship with her contacts (Wiese, Hong, & Zimmerman, 2014). To understand the feasibility of this, I collected the contact lists of 54 participants, containing 35,599 contacts. However, to my surprise 67% of the contact entries that I collected contained either no contact information, or only an email address. Most of the remaining 33% of contacts only contained one piece of information, usually a phone number. The majority of contact list features were unused.

Despite the apparent lack of information contained in these lists, a deeper exploration of the content uncovered more subtle structures within the data. Analysis of the contact name field yielded twelve distinct and unexpected naming strategies.

This analysis of contact lists from a broad range of 54 participants found that those lists were used in surprising ways and revealed consistent patterns. The behaviors we identified present both a challenge and an opportunity: though usage patterns prevent simple automated approaches for data mining or contact-list merging, they also suggest alternative directions for data mining to understand the behavior of individuals and their relationships with others. More broadly, the results of this work point to a mismatch between the expected use and actual use of the contact list, a very common interface for interacting with personal data.

2.11.2 Context Awareness: Inferring Phone Placement

One example of context awareness from my own work is using sensors to infer the placement of a device (Wiese, Saponas, & Brush, 2013). Enabling phones to infer whether they are currently in a pocket, purse or on a table facilitates a range of new interactions from placement-dependent notifications setting to preventing "pocket dialing." Phone placement data may not seem to be personal data at first glance, but over time phone placement data can be used to characterize the behavior of individuals.

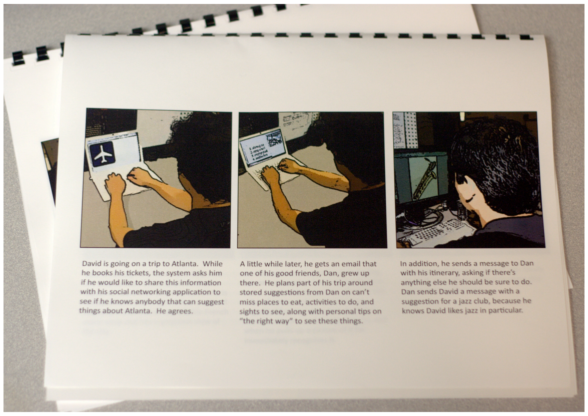
In this work I collected two weeks of accelerometer data from 32 participants' personal mobile devices. Using the experience sampling method (ESM), participants recorded how their devices were being stored in-situ. To evaluate algorithms for inferring the placement or proprioception of the phone, I built and evaluated models using features from the in-situ accelerometer data. These models achieve accuracies of 85% for two different two-class models (Enclosed vs. Out and On Person vs. Not) and 75% for a four-class model (Pocket, Bag, Out, Hand).

I also explored opportunities to improve the accuracy of the accelerometer-only models, using prototype sensors that leverage capacitive sensing (previously unexplored for this task), multi-spectral properties, and light/proximity sensing. I compared data gathered with these sensors in a laboratory setting, with resulting models achieving top accuracy levels of 85% to 100%.

This work represents one example of developing a context-aware component of an application. To the extent that a smartphone is associated with a primary user, the place that they put their phone is a form of personal data: it describes something about the user's behavior. Furthermore, over time logs of this data can reveal trends that might offer even more information on the user's behavior.

2.11.3 Lifelogging and Identity Interfaces: Evaluating Applications that Make Use of Long-Term Location History

A major shortcoming of early Lifelogging research was the general lack of applications for collecting large logs of personal data. The process of testing application ideas and finding value can be a difficult one, but is an important component of human-centered computing.



David is going on a trip to Atlanta. While he books his tickets, the system asks him if he would like to share this information with his social networking application to see if he knows anybody that can suggest things about Atlanta. He agrees.

A little while later, he gets an email that one of his good friends, Dan, grew up there. He plans part of his trip around stored suggestions from Dan on can't miss places to eat, activities to do, and sights to see, along with personal tips on "the right way" to see these things.

In addition, he sends a message to Dan with his itinerary, asking if there's anything else he should be sure to do. Dan sends David a message with a suggestion for a jazz club, because he knows David likes jazz in particular.

Figure 5: An example of a scenario presented during one of the sessions.

In one example from my own work, I developed a set of scenarios that illustrated some potential use cases for applying histories of a user's location and conducted a needs validation session, following guidelines from the "speed dating" design technique (Davidoff, Lee, Dey, & Zimmerman, 2007). Needs validation uses storyboards of different scenarios, in our case to depict different concepts for location histories and future history (see Figure 5), to provide participants with many quick views of possible futures. During a session, a researcher presents the storyboards one at a time to an individual or to a small group of participants. The researcher then follows the storyboard with a lead question that focuses the discussion on the underlying need and away from the specific way the technology in the storyboard shows the need being addressed. By presenting the participants with storyboards showing people like themselves in situations that seem common, this method helps participants draw on their own experience as they visit and reflect on an imagined near future.

Participants were invited to share their reactions to the storyboards and to address the corresponding questions, which ask them specific ways that their own experiences have led them to a similar need as the one addressed in the scenario. Furthermore, participants were told not to think about the technology that would be used to implement these scenarios, but just to assume that the technology could work.

I brainstormed 36 scenarios, which I refined down to 18 based on redundancy of the underlying need we were addressing and the level of convincingness for each scenario. Once made, I thematically clustered the scenarios based on content. These clusters, and results from the needs validation, are described here:

- **Icebreaking (3 scenarios)**: These scenarios describe situations where location history is used to strengthen existing relationships or build new ones. Participants strongly identified with the needs implicit in the icebreaking scenarios (e.g. needing a good topic of conversation when talking with a new person). However, participants were also concerned that the usage of a conversational aid that supplies conversational topics might be awkward, make them look socially inept, or even come across as "stalkerish". Participants felt that this kind of technology would be best suited for professions where social relationships are brief (e.g. nurse, taxi driver), so building commonality earlier is better. Recent work has continued to explore this concept of computer-supported icebreaking (Nguyen, Nguyen, Iqbal, & Ofek, 2015), though not necessarily using location history.

- **Future – intersections and obstacles (3 scenarios)**: These scenarios address different aspects that affect one's plans in the short or long term, including things that might inhibit them or that they may want to include in their plan. There are fewer future location scenarios compared to location history because of the differences in how easy it is to obtain that data accurately, which affects technical feasibility. Participants reflected a clear need for monitoring how different logistics might affect their future plans, and they also strongly identified with being able to take advantage of opportunistic serendipitous overlap with friends that they had not seen for a while. One concern expressed by some participants was that they take pride in "having it together" and being prepared for different situations, which they feared this kind of technology might diminish.
- **Identifying a person by time and place (2 scenarios)**: These scenarios explore the idea that there are some situations where one would want to contact the people that were around them in a particular place some time in the past, but do not have contact information for them. Scenarios here provide a functionality that is otherwise not available in the real world. One place where this functionality is slightly available is through Craigslist "Missed Connections"[13], which allows people to post a message, hoping that the person they saw at a particular place will get that message. While participants were not at all interested in the scenario for supporting missed connections, but they were however much more interested in situations where they had spoken with somebody, but had not exchanged contact information. Also, when the scenario motivation was functional rather than social, (e.g. who left the meeting room messy, or who saw the car accident), then it was no longer a problem if they hadn't spoken. However, one important issue here was that participants did not want the barrier for a stranger to contact them to be too low.
- **Personal traces (7 scenarios)**: These scenarios build on data from interviews with early adopters, which suggest that there is value in having access to your own location history. Participants responses to these scenarios were in many ways neutral: there was no problem with the scenarios, but they also were not really sure how much value the scenarios offered. This differed from the early adopters who had expressed that these kinds of scenarios are a major motivation for their usage of location logging applications. One explanation for this disparity is that the real value in this scenario actually comes from being able to see your own data, so speed dating might not be the best way to evaluate this because participants are not looking at their own data. Additionally, these scenarios are usually easier to implement: they only depend on the location logs of an individual, while

[13] http://pittsburgh.craigslist.org/i/personals?category=mis

many of the other scenarios would require wide adoption in order to realize their potential.

- **Mining existing social networks for location overlap (3 scenarios)**: These scenarios use existing social connections to share information, experience, or interest in a place. They differ from icebreaking in that their primary goal is not to strengthen the relationship, though it could certainly be a byproduct. Participants expressed a strong desire for these scenarios, particularly social place recommendations. Participants expressed that this would reduce the need to read and write reviews (i.e. if you see that somebody you know has been there, you can just ask them). Additionally discussion of these scenarios revealed an additional unmet need: the need to identify common interests with friends that were previously unknown.

The results from this work demonstrate the importance of creating low fidelity prototypes of the future, in particular where personal data is concerned. At a high level, this exercise demonstrated that location history logs have the potential to offer real value to consumers. Even today, 6 years after that study was run, location data is mostly used in the form of "present location", very little with past or planned/projected future location. With longer location logs and better ways of managing this data, a lot more seems possible. On the other hand, there were numerous issues and concerns associated with many of these scenarios that really reveal the challenge and the complexity of working with personal data.

2.11.4 Identity Interfaces: Mailing Archived Emails As Postcards[14]

Recent research speculates that changes to the form or behavior of virtual things might increase people's perceptions of value (Odom, Zimmerman, & Forlizzi, 2011). To investigate this further, we designed and deployed a technology probe that radically altered the form and presentation of potentially valuable elements within people's massive email archives by sending them physical postcards of email snippets. We interviewed participants, probing to understand the properties of cards that did and did not encourage self-reflection, a behavior shown to be associated with value creation (Odom et al., 2011) and a behavior that reflects the meaningfulness of an item.

For the technology probe, we created a piece of software to extract potentially meaningful snippets from a person's archive based on several heuristics, and chose photos from Google Image Search using the generated snippet as a search term. We conducted the study over a three-month period. We sent a postcard (with the image and the snippet) to each participant at a random interval between 7 to 10 days so that they likely received each new card on a different day of the week. We conducted three in-home interviews with each participant at the beginning of the study, one months, and three months into the study.

[14] Work done in collaboration with Jennifer Olson, Dan Tasse, David Gerritsen, Tatiana Vlahovic, William Odom, and John Zimmerman

During the interviews with participants the postcards caused them to reflect on events, people, and humorous memories or jokes that were related to the snippets. However, many cards also left the participants feeling bemused or disinterested because they could not place the card in context. In these instances participants looked to the images on the postcards, but because of the loose coupling between snippet and image these were not helpful aids. Most participants also felt uncertain about where to place a card after it arrived. Contrasted with email, when the snippet arrived as a material postcard, the fact that the message was in their hand forced them to evaluate it from a new perspective in order to determine its next location in the world.

This technology probe offers evidence for a variety of insights that further our understanding of how people relate to their personal data. At a high level, participants had clearly not thought about or engaged with their vast email archives. Even with email archives, personal data that is amongst the more accessible to end users, it is still a mostly untapped archive full of rich memories. Another insight offered by this work was the fragmented state of personal data today, brought into sharp relief by how difficult it was to contextualize the email snippets. If personal data were less fragmented it might have been easier to select meaningful snippets, and we could have better contextualized the snippets for users. Perhaps the postcard photos could themselves have had more contextual meaning, even coming directly from the participants' photo archives.

Finally, this technology probe demonstrates strong limitations on being able to interpret personal data. Many systems attempt to draw insights from personal data automatically. In this probe, even the participants, who should theoretically be the gold standard for interpreting information from their own email archives, often struggled to interpret those archives. Moving forward, to realize value by interpreting personal data the people who are the subject of that data must be involved.

2.11.5 Context Sharing: Facilitating Workplace Awareness through MyUnity

I deployed myUnity, a cross-platform awareness system designed to support awareness, communication, and collaboration for an office worker environment (Wiese, Biehl, Turner, van Melle, & Girgensohn, 2011). Where previous systems were platform-centric, myUnity supported both mobile and desktop environments both for sensing information and also for presenting that information. myUnity brings together personal data from disparate sources including: vision-based office activity, mobile phone location, desktop location via network, calendar, IM presence, and phone call status. The myUnity server aggregated this data, and also aggregated these disparate data sources into a higher-level abstraction of "presence", which was a proxy for how accessible an individual in the system was. Results from a deployment of myUnity highlighted the value of connecting to multiple data sources and of automating the sharing process.

From the perspective of personal data, there several valuable takeaways from this work. First, the value of a service (in the case of myUnity the service is awareness) can be amplified by including multiple data sources, rather than a single one. Furthermore, even some simple processing to make a higher-level inference (e.g. presence) can be very useful for helping people find value in the data. Finally, much of the value in myUnity came from the automated nature of the sharing process.

2.11.6 Context Sharing: Understanding User Motivations for Sharing Location Using Foursquare

We examined social location sharing on the check-in-based location sharing site Foursquare (Lindqvist et al., 2011). Foursquare is typically considered to be the first successful social location-sharing service. In this work we conducted interviews with early adopters and deployed two surveys to understand the reasons why people use Foursquare. One major goal of this work was to gain perspective on the factors that led to the success of this site in the domain of location sharing where many others had previously failed. In this work, we found a variety of reasons why people used Foursquare: to have fun and earn badges, to facilitate social connection, to discover new places, and to keep track of where they had been previously. While it seems that over time the novelty of the social gaming wears off, the value of other motivations persisted over time.

Perhaps most notably from these findings is that there was not one specific "killer app" that led to the success of Foursquare where so many location-based applications had previously failed. Instead, there were a variety of reasons that people were using Foursquare, which combined to make the site successful. This combination of motivations seems to have helped Foursquare overcome the difficult chicken-and-egg problem that plagues many services: a service can offer an exciting new feature once it has built up users and data, but it can only build up the user base if it offers value to users to begin with. This is a challenge that extends beyond location data and applies more generally to applications and services that depend on personal data.

2.11.7 Privacy: Understanding Privacy preferences by investigating self-censorship

I have explored user's privacy decisions was by investigating their decisions not to post content on Facebook, a phenomenon we termed self-censorship (Sleeper et al., 2013). In this work we asked participants to take note of when they considered sharing a piece of content on Facebook, but instead decided not to. Participants were instructed to send a quick text message whenever this occurred with a few words to describe the situation (Brandt, Weiss, & Klemmer, 2007), and then to complete nightly surveys that described the situations in more detail. We conducted and coded semi-structured interviews with participants. The findings of this work indicated that in many cases, participants chose not to share content because it would have required too much effort to specify the subset of people that they wanted to share with. Instead of Facebook's manual list-based sharing controls, participants wanted to be able to specify a target sharing group more dynamically, using factors such as: life facet (specific work/school, family), demographics (age, gender, geography, race), tie strength, and the person's relationship with the post (i.e. will this person be interested).

This work has broader implications for personal data. First, if privacy controls are inadequate for capturing user's preferences within a service, it may lead to decreased usage of the service, and hence less overall value for the user. This may be especially significant for new and less established applications and services which do depend on a critical mass of engaged users in order to succeed. The second implication for personal data is that personal data could make possible the kind of sharing controls specified above. Specifically, the dynamic factors above that specify the target sharing groups refer either to information about the person being shared-with, or about the relationship between that person and the sharer. These are both types of personal data, and thus if a system had access to this personal data it might enable a new class of privacy controls.

2.12 Discussion

This chapter has offered a brief overview of a variety of research domains that relate to personal data. Particularly striking through this chapter is the broad variety of perspectives and research that relate to personal data: active research is taking place with personal data across many different domains, often with a tenuous or even totally absent connection between those domains. This lack of coherence across personal data research is counter-productive, and even potentially harmful. Research in one domain wrestles with issues and challenges that have already been explored in a different domain. Today the canon of personal data research is extremely scattered, if a researcher wanted to think about her research project holistically with respect to personal data, it would be very difficult to even know where to start. How does a particular piece of research relate to the broader landscape of personal data? What are the major issues that exist around personal data? Who are the stakeholders involved? What solutions already exist to challenges that I'm encountering? Answering questions like these requires considering personal data as its own topic, from a holistic perspective.

In the early days of research that involved personal data, before smartphones and the Internet, a research system could be completely self-contained. In this way, research with personal data was simpler then than it is today. For example, user modeling researchers could collect data within the context of their particular experimental system, and that data was sufficient for pushing the field forward. Users did not have other data that could potentially be added to the system, it simply didn't exist yet. One research system was unlikely to need to use the data from a different system, and the participants in an experiment with one system were unlikely to be participants with another system anyway. In the case of context aware systems, researchers could focus only one type of data: the data in their systems.

Furthermore, they could focus only on immediate context, data that had been sensed immediately or in the short term.

Current research with personal data has stagnated in large part because it continues to follow these trends. Researchers use data from one or two sources. They often have to collect the data themselves using ad-hoc, one-off systems designed specifically for their study. When the study is done, the infrastructure used to collect that data, application that was built on top of personal data, it all dies with that particular study. However, where this was once acceptable and a reasonable approach to conducting research, it no longer makes sense. Today, people have large amounts of personal data built up across a variety of applications, devices, and services, and failure to take advantage of this is at best a missed opportunity.

Across the board, work on personal data is pushing up against the limitations of this approach: virtual possessions, context sharing, personal informatics, and user modeling are all dealing with various formulations of the problem that there is no uniform way of accessing or working with personal data. The solution to these challenges will not come from a single research project or line of inquiry. The space of personal data is complex and multifaceted, and advancing the way that it is handled today will require a holistic approach with advances across many disciplines.

In short, personal data needs to be established as a separate research area, bringing together researchers of many backgrounds to focus on and solve the challenges present in this very important aspect of computer science. A holistic, multi-disciplinary approach to personal data will lead to stronger research contributions across the board, and establishing standard tools, approaches, and protocols for working with personal data will benefit the entire research community.

3 A Case Study: Inferring Sharing Preferences Using Communication Data

A central claim of this dissertation is that working with personal data is an unnecessarily arduous process. The previous chapter lays the groundwork for this at a high level: the current ad-hoc process of working with personal data inhibits many research areas. Working with personal data is challenging for a wide variety of reasons, and efforts to improve the state of personal data require an understanding of these challenges. How, specifically, does the current ecosystem of personal data inhibit research or application development?

Building this understanding is itself a challenging task. Application developers and their companies make many decisions (often implicitly) throughout the software development process and even before it begins based on myriad issues and considerations around personal data. Even with complete access to the entire software development process, there is no expedient way to capture that data in order to understand these challenges.

Exploring this question from the perspective of research offers a different view with some tradeoffs. Research applications can ignore or temporarily solve issues around developing with personal data (e.g. user adoption, privacy concerns), which distance them from the reality of deploying a production system. Research is often intended to push the boundaries of what is possible, which can offer a stronger perspective of how the current state of personal data may be limiting the imagined future of that research vision.

This chapter documents my work to infer social sharing preferences using people's communication history. In the context of this thesis, the ability to infer social sharing preferences from automatically collected communication logs is an example of translating low-level personal data into a higher level understanding of the user (i.e. who is she willing to share sensitive personal information with?). The process documented offers numerous concrete examples of the challenges of working with personal data.

On a functional level, the ability to infer sharing preferences from automatically collected data offers real value for users. It is often reported that people are unlikely to adjust the default privacy settings, sometimes even choosing not to share at all rather than adjusting their sharing preferences (Sleeper et al., 2013). Automatically inferred sharing preferences have the potential to avoid sharing too much or too little information, both of which can be harmful for users: while over-sharing can annoy others, cause embarrassment, or even lead to job loss, under-sharing has social consequences as well, including missed opportunities for connection and social support.

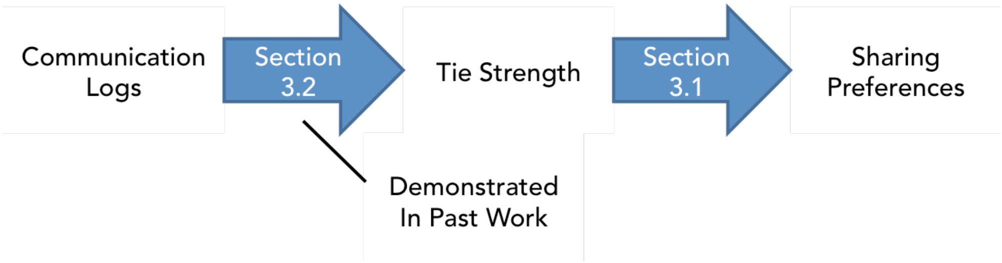


Figure 6: The goal of the research in this chapter is to use communication logs (call and SMS logs) to infer sharing preferences, using tie strength as an intermediate representation. Theoretical literature supports the connection between communication logs and tie strength and also the connection between tie strength and sharing preferences. Additionally, past work has demonstrated that communication behavior corresponds to tie strength. Therefore, I focused first on the connection between tie strength and sharing preferences before attempting to replicate the prior finding connecting communication behavior to tie strength.

The insight that communication behavior has the potential to predict sharing preferences is based on a combination of two different findings in the HCI and social science literature. The first finding connects communication behavior with the social science construct of *tie strength* (informally this is the strength of the relationship between two people): more communication between two people indicates a stronger tie between them (Granovetter, 1973). This finding applies across all communication between two people, including in-person communication. Not only does social science theory support the connection between tie strength and amount of communication, but recent work has demonstrated this connection in social media (Gilbert & Karahalios, 2009). Furthermore, a number of recent research projects have use communication frequency as a direct proxy for tie strength (Conti et al., 2011; Miritello et al., 2013; Onnela et al., 2007; D. Wang et al., 2011). Automatic detection of strong ties has many potential benefits. Social support from strong ties has been associated with mediating the occurrence and severity of depression (N. Lin & Dean, 1984) as well as finding employment after losing a job (Burke & Kraut, 2013). Automatic detection of strong ties could also be useful for a variety of user interface personalization: determining notification preferences, sorting contact lists, or setting sharing preferences.

The second finding relevant to communication behavior and sharing preferences comes from theoretical literature on sharing. Belk distinguishes two sharing motives. When "sharing-in" people share things with people they feel close to or desire to feel closer to, as a way of strengthening this relationship. "Sharing-out" involves interactions with people outside of close social boundaries and is generally more like gift-giving or commodity-exchange (R. Belk, 2010). However, unlike tie strength and communication, the HCI literature had not explored the connection between tie strength and sharing preferences.

With the connection between communication and tie strength already established in the literature, this chapter demonstrates a connection between features of social relationships and users' preferences for sharing different kinds of personal information (Section 3.1). However, using phone and SMS logs as communication data, this work could not predict the entire chain (going from communication preferences to tie strength to sharing preferences; Section 3.2). Specifically, phone and SMS log data was not sufficient to accurately predict strong ties. Altogether, this process highlights many of the challenges and complications inherent in working with personal data.

3.1 Connecting Features of Social Relationships to Sharing Preferences

The study presented in this section explores salient features of interpersonal relationships that predict the user's preference for sharing personal information, such as location, proximity to another person, and activity. Specifically, this study examines the association between several factors (e.g., collocation frequency, communication frequency, closeness, and social group) with preferences for sharing specific kinds of information. In this online study, participants provided basic demographic information and a list of friends. They then associated each friend with relevant social groups, rated their perception of closeness with each friend (tie strength), and stated a willingness to share information with each individual for 21 different sharing scenarios.

3.1.1 Method

To recruit participants, I posting ads in several nationwide online bulletin boards and through two study recruiting websites. Prospective participants were selected based on several criteria:

- *Age (20 - 50):* This age range includes different life stages, especially with respect to being a parent or child within an immediate family.

- *Occupation (non-student):* Students were excluded because they do not easily allow distinctions between work and school groups.
- *Social network membership (members of Facebook with at least 50 Facebook friends):* This was a source for generating friends' names for the study. Additionally, membership in a social networking site indicates that participants are more likely to want to share information about themselves with people they know, allowing us to observe differences in their sharing preferences (as opposed to a person who does not want to share at all).
- *Mobile device usage (must have a smartphone):* Participants with smartphones were more likely to understand the potential values and risks of the sharing scenarios.

Participants were compensated $20 for completing task 1, and $60 for completing tasks 2 and 3 (listed below, and described in more detail in the following sections). The data collection took place online and participants were given two weeks to complete all parts of the study.

Participants completed three distinct activities:

1. Generating a lists of friends
2. Describing each friend in terms of closeness and affiliation with different groups
3. Stating willingness to share different kinds of information with each friend

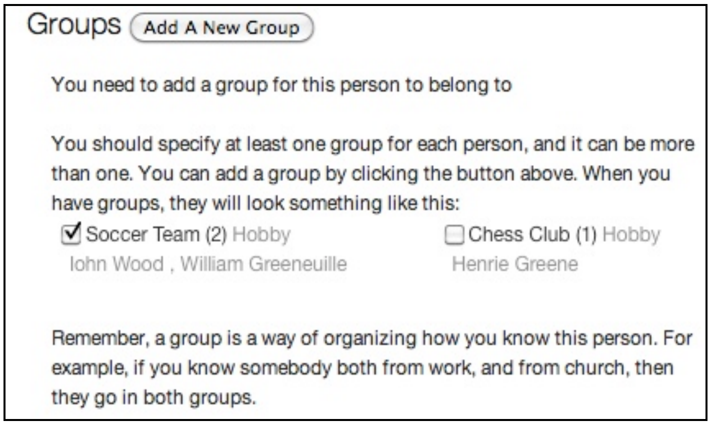
Generating lists of participants' friends

To ensure that participants would answer questions about friends who varied in social group and in closeness, I asked participants to provide two lists. The first list was intended to target potential strong ties, and was generated from categories which I derived from qualitative work on relationships (McCarty, 2002; Spencer & Pahl, 2006). The categories were:

- People you currently live with (5 people maximum)
- Immediate family members (5)
- Extended family members (10)
- People you work with (10)
- People you are close to (10)
- People you do hobbies or activities with (10)

I instructed participants to avoid duplicates. The second list consisted of all of their Facebook friends. I provided participants with instructions on how to download this information from Facebook.

The final friend list included everyone from the first list (typically less than 40 people), plus a random sampling from the Facebook friend list to reach 70 total friend names. Each list was checked for duplicates and for names that the participant did not recognize. If any were found, they were replaced with randomly selected names from the Facebook friend list. This final list of 70 names is referred to as the "friend list."



Groups (Add A New Group)

You need to add a group for this person to belong to

You should specify at least one group for each person, and it can be more than one. You can add a group by clicking the button above. When you have groups, they will look something like this:

☑ Soccer Team (2) Hobby
Iohn Wood , William Greeneuille

□ Chess Club (1) Hobby
Henrie Greene

Remember, a group is a way of organizing how you know this person. For example, if you know somebody both from work, and from church, then they go in both groups.

Figure 7: The instructions for the grouping activity.

Describing each relationship

Next, participants provided information about their relationship with each person on their friend list. The complete list of data collected per friend is in Table 1. I organized this information into two categories: data that would be easily observable from within a UbiComp system or social networking site, and data that would require more work either to infer from observable features or for the user to express manually. Participants indicated tie strength by answering the question "How close do you feel to this person?" on a 1-5 Likert scale. This approach is similar to the one taken in work by McCarty (2002).

	Data collected	Data type
Observable features	Friend sex	Male/Female
	Friend age	Rounded to nearest year
	Years known	
	Frequency seen	Likert 0-7: Less than yearly (0), yearly, yearly-monthly, monthly, monthly-weekly, weekly, weekly-daily, daily (7)
	Frequency communicated with electronically	
Non-observable	Closeness (strength of tie)	Likert 1-5: very distant (1), distant, neither distant nor close, close, very close (5)
	Group	Participant-dependent, however each group was put in a pre-specified category

Table 1 Data collected for each friend. Data in the top half of the table ("observable features") is data that was potentially observable by a UbiComp system or social networking site. Data on the bottom half of the table would either be inferred from the observable features or manually inputted by the user

Next, participants detailed their mutual affiliations with each friend by placing them into groups. The interface (see Figure 7) allowed participants to create personalized groups. In addition, it required them to classify each group into one of 12 pre-determined categories: neighborhood, religious, immediate family, extended family, family friend, know through somebody else, work, school, hobby, significant other, trip/travel group, and other. I developed these categories based on a combination of literature sources (McCarty, 2002) and data from previous work on grouping friends in social network sites (Kelley, P.G., Brewer, R., Mayer, Y., Cranor, L.F., Sadeh, 2011). I instructed participants to indicate at least one group affiliation for each friend, and encouraged them to indicate multiple group affiliations when relevant. For example, if a person and their friend went to college together, and they both attend or attended the same church, the participant would place that friend in two groups. The result is a set of affiliations, and all of the people on the friend list who are associated with each affiliation.

Sharing scenarios

Finally, participants indicated their willingness to share information with each friend in the context of 21 different information-sharing scenarios (see Table 3).

To develop the final list of scenarios, I first brainstormed over 100 different UbiComp scenarios in which individuals could share information, such as location, activity, calendar, history, photos, etc. I grouped scenarios into 11 categories based on the type of information being shared. I assembled these scenarios in a survey and posted it on Amazon's Mechanical Turk, with two questions for each scenario:

- How often do you currently share this information now (whether with one person or with many people): never, seldom, sometimes, frequently, constantly
- How useful is it to you to share this information with somebody you know, answering for maximum usefulness: totally useless, somewhat useless, neither useless nor useful, somewhat useful, totally useful

I used the results from this survey as a guide to reduce the list of 100 scenarios down to 21 specific scenarios. Survey results allowed me to pick scenarios with information that respondents found was more useful to share. Further, for that information that would be useful to share, I selected for a range in current sharing practices, including a mix of information that people currently do and do not share. The resulting list fit into five different categories: current personal location (7), personal location history (5), calendar and location plans (7), communication activity (1), and social graph information (1). See Table 3 for a list of the final set of scenarios used.

For each of the 21 scenarios, I asked participants to indicate their willingness to share information with each of their 70 friends using a 5-point Likert scale (labels: 1-definitely not, 3-no preference, 5-definitely). I adapted this method based on past work (Olson, Grudin, & Horvitz, 2005).

3.1.2 Findings

Forty-two participants completed the study. Their occupations ranged from education and engineering to administration and legal. I eliminated three problematic respondents who each demonstrated no variance for more than 65 out of the 70 friends; each individual friend had the same rating for each of the sharing scenarios. These participants seemed to have simply rated the sharing scenarios as quickly as possible. Of our remaining 39 participants, there were 28 female and 11 male, with ages ranging from 21 to 49 (M=29.8, SD=6.4).

n=2370	Mean (SD)	Sharing M=2.83(0.66)							closeness
friend sex = female	55.5%					0.01	0.01	0.01	0.01
friend age	32.7 (12.3)					-0.01***	-0.004***	-0.005***	-0.01***
frequency seen	1.8 (2.2)					0.06***	-0.03	-0.03*	0.21***
frequency comm	2.5 (2.4)					0.17***	0.03**	0.04***	0.34***
years known	10.5 (9.8)					0.03***	0.02	0.01**	0.03***
user age × person age						-0.0004	-0.0004*	-0.0002	0.0001
user sex = female × friend sex = female						0.02	0.01	0.02	0.02
freq seen × freq comm						-0.02***	0.004	0.003	-0.05***
user age × years known						- 0.0007**	-0.0004	0.00005	-0.001**
friend closeness	2.7 (1.4)		0.45***		0.40***		0.41***	0.37***	
is family	23.2%			0.59***	0.24***			0.22***	
is social	66.9%			0.14***	0.03			0.03	
is work	18.0%			0.18***	-0.02			-0.001	
Intercept		2.86***	1.62***	3.25***	1.89***	2.33***	1.66***	1.99***	1.61***
R^2 (variance explained)		0.36	0.63	0.48	0.65	0.57	0.65	0.66	0.70
Model Name		User	Close	Mode	Non-Obs	Obs	Obs+close	All	

Table 2: Linear regression models predicting sharing and closeness (last column only), controlling for each participant. Each column is a different model and data in the table are non-standardized β coefficients, except for R^2 in the last row, which can be compared across models to demonstrate the variance explained. For example, the "close" model (fourth column) includes one effect, friend closeness, and this model accounts for 63% of the variance in sharing preferences. Gray cells indicate effects that were not included for that particular model. The data indicate both that closeness is the best predictor of sharing, and that observable features can predict closeness. Significance: *p<0.05; **p<0.01; ***p<0.001

Modeling sharing preferences

The differences in participants' mean sharing answer indicated a range of individual privacy/sharing preferences (M = 2.83 out of 5 where 5 is "definitely willing to share this information with this person", SD = 0.66). To address the question of which relationship characteristics predict sharing preferences, I conducted a mixed-model analysis of variance predicting sharing as the outcome variable (see Table 2, note that the variables 'user age' and 'user sex' refers to our study participants). This analysis accounts for the non-independence of observations within each participant. Running this analysis with different models allows for an exploration of which combinations of independent variables explain the most variance in participants' sharing preferences.

All of the regressions in Table 2 were done on a per-friend level of analysis; the models use the mean sharing value across all scenarios for each friend (n=2730) as the dependent variable, and the features that described each relationship were effects in the models. The models included the participant as a random effect to account for non-independence of ratings within each participant. The first column of Table 2 shows means and standard deviations for all continuous effects in the model.

The second column in Table 2 (model name = user) is a model that has no effects except for the effect of the participant (which accounts for individual differences among participants). The result shows that certainly some amount of the variance relates to individual differences, indicating preferences for sharing in general (R^2 = 0.36). Models that additionally accounted for participant-level effects of sex and age performed poorly.

Modeling sharing preferences with non-observable features

The third, fourth, and fifth columns in Table 2 show models with effects that only include the non-observable data. For these analyses, I sorted group categories into the three descriptive "life modes" identified by Ozenc and Farnham, (family, work, and social) (Ozenc & Farnham, 2011), which they suggest are the primary areas of a person's life.

Closeness by itself turns out to be a very strong predictor of sharing preferences (model name = close, R2 = 0.63) with each 1-point gain in closeness accounting for a 10% increase of the sharing outcome. This means that a friend who is at closeness 5 (top closeness) is 40% more likely to be shared with than a friend at closeness 1 (bottom closeness). The regression that only had life modes as a predictor did not account for as much of the variance as closeness alone did (model name = mode, R2 = 0.48), with membership in family, work, and social modes accounting for a 12%, 3%, and 3% increase in likelihood to share respectively (note that all friends were categorized into at least one of these modes). This means that just knowing which of these categories a contact is in is not particularly helpful in predicting sharing preferences. Finally, adding life mode to closeness resulted in only a slight increase in performance over just closeness (model name = non obs, R2 = 0.65), and resulted in a loss of significance for the "social" and "work" effects: closeness and family were all that mattered in this model, with participants being more likely to share with contacts they are closer to and with contacts that are family members.

Modeling sharing preferences with observable features

The previous section discussed models based on relationship features like closeness that are not immediately observable. How well do observable features predict sharing? These observable features (see Table 2) include friend age, sex, years known, frequency seen, and frequency communicated with. I call these features observable because current UbiComp systems are capable of capturing them from existing social network data, or from sensor and communication logs. As such, by testing these features, I can evaluate how well a fully automated system might perform for predicting sharing preferences. This model performed well (model name = obs, R^2= 0.57), though still not as well as the model with just closeness. Significant effects included friend age (0.2% less likely to share per year), frequency seen (1.4% more likely to share per point increase), frequency communicated with (3.6% more likely to share per point increase), years known (0.6% increase per year known). The only feature that was not predictive was friend sex.

The model also included four interactions. First, I included the interactions between participant and friend sex and the interaction between participant and friend age to see if homophily accounted for sharing preferences (are people more likely to share with others of the same gender or others who are closer in age?), but neither of these interactions were significant.

The next interaction was between years known and participant age, which I included because I hypothesized that the duration of a person's life that they have known another person might be a useful indicator. This did have a very small effect, indicating that younger participants were more greatly influenced by how long this person had known them.

Finally, the model included an interaction between frequency seen and frequency communicated with. I hypothesized that some strong ties are communicated with much more often than they are seen (e.g. family who do not live nearby); similarly, some weak ties are seen often but not communicated with particularly frequently (e.g., one might see coworkers often, but not communicate with them outside of work). This interaction was also significant, revealing that communication is a stronger indicator of willingness to share when collocation is less frequent.

Modeling sharing preferences with observables and non-observables

The next model includes both observable and non-observable features. This model (model name = obs+close, R^2 = 0.65) includes all of the observable features (and the interactions described in the previous section), and also includes closeness. This model explains 65% of the variance in sharing preferences, an improvement over the

57% explained by the model that only included observable features, without closeness. Closeness has nearly the same effect in this model as it does in the closeness only model, with each point in closeness increasing the likelihood to share by 8.8%. Frequency seen is no longer significant in this model, neither is the interaction between frequency seen and frequency communicated with. Additionally, frequency communicated with has less of an effect in the model (0.8% more likely to share per point increase, down from 3.6%).

The final model (model name = all) added life mode to the obs+close model described above. Including all features in the model led to almost no difference in the variance explained (R^2= 0.66, compared with R^2 = 0.65 for the obs+close model), and the model effects were nearly identical to those in the previous model. A model that kept all 12 group categories distinct instead of grouping them into the 3 life modes was comparable (R^2= 0.67).

Overall, the models with closeness explained more of the variance in sharing preferences than any of the models without closeness, and adding closeness results in the loss of significance for other effects in the model.

Predicting closeness using observables

Since closeness is such a predictive feature, it is worth examining how well the observable features of each relationship predict closeness. I used the same approach as before, with a mixed-model analysis of variance controlling for participant as a random effect, but this time with closeness as the outcome. I included all observable effects from the other models. This model was quite effective (R^2 = 0.70, last column of Table 2). Significant effects in this model included: friend age (0.2% less close per year), frequency seen (4.2% closer per point increase), frequency communicated with (6.8% closer per point increase), years known (0.6% closer per year). The interaction between frequency seen and frequency communicated was also significant, showing that communication has a much stronger effect when collocation is infrequent. The interaction between participant age and years known was significant with a small effect as before. The friend's sex and the interactions of the participant's and friend's age and participant's and friend's sex were not significant.

Scenario	Pearson's r with closeness	Mean Sharing	Std Dev	Tukey-Kramer HSD			
The next calendar event that we have in common	0.39	3.45	1.42	A			
All calendar events that we have in common	0.39	3.40	1.42	A			
I am with a person who we both know	0.43	3.36	1.39	A			
I'm within 1 mile of this person	0.49	3.26	1.46		B	C	
Details of who my family connections/family relationships are	0.46	3.17	1.39		B	C	D
My personal travel plans that mean we will be in the same place	0.43	3.17	1.54			C	D
My location when I am closer to this person than we normally are	0.35	3.06	1.60			C	D
Everywhere I have travelled to	0.47	3.02	1.36	E			D
My location when I am on vacation	0.53	3.02	1.38	E	F		
I've been to the place that this person currently is	0.42	2.94	1.43	E	F		
My work travel plans that mean we will be in the same place	0.36	2.94	1.62		F	G	
All places that I've been to that this person has also been to	0.42	2.92	1.42		F	G	H
My location when this person has been here before	0.41	2.84	1.37	I		G	H
Everywhere that I have gone out to eat	0.38	2.80	1.34	I			H
I'm at home during a normal weekend	0.50	2.71	1.31	I			
When I am usually at work	0.40	2.45	1.29		J		
My location wherever I am	0.39	2.39	1.34		J		
My tentative plan for the day	0.36	2.23	1.24			K	
I'm in a call on my cell phone	0.25	2.19	1.30			K	L
When the next thing on my calendar starts	0.33	2.10	1.18				L
All details of the next event on my personal calendar	0.33	1.93	1.19	M			

Table 3: Summary of data for each sharing scenario, sorted by overall mean sharing. The first column reports the correlation with closeness, and all correlation coefficients are significant to p<.001. The Tukey-Kramer test compares the overall means for sharing in each scenario: scenarios that have the same letter are not significantly different from each other.

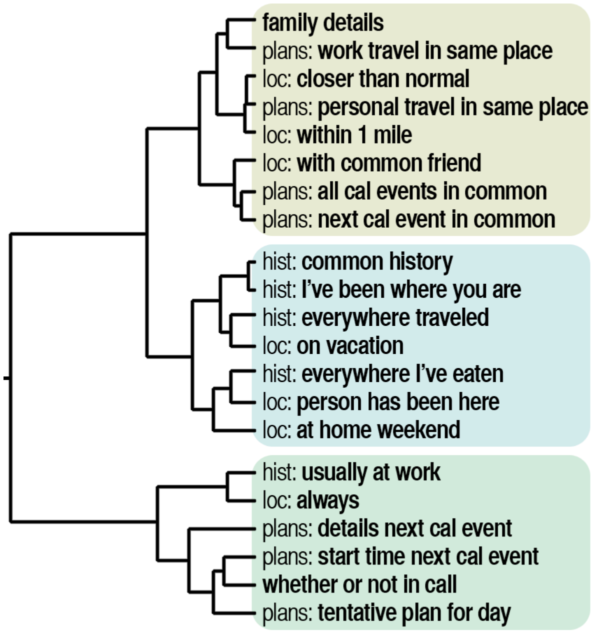

family details
plans: work travel in same place
loc: closer than normal
plans: personal travel in same place
loc: within 1 mile
loc: with common friend
plans: all cal events in common
plans: next cal event in common
hist: common history
hist: I've been where you are
hist: everywhere traveled
loc: on vacation
hist: everywhere I've eaten
loc: person has been here
loc: at home weekend
hist: usually at work
loc: always
plans: details next cal event
plans: start time next cal event
whether or not in call
plans: tentative plan for day

Figure 8: Hierarchical clustering using average linkage distance. Horizontal position of the branches is directly proportional to the calculated distance between each cluster. Scenarios are shorthand for the same ones in Table 3.

Willingness to share across different scenarios

The previous analyses examined sharing preferences in general, and found that participants were more willing to share with closer ties. Are there differences in how well closeness predicts sharing between the different sharing scenarios? Is closeness a strong predictor for certain scenarios only, or for all scenarios? Correlations between closeness and willingness to share are significant for all of the sharing scenarios, with Pearson's correlation values ranging from r=0.25 to r=0.53, all p<0.001 (see Table 3 for all values).

By asking about sharing across 21 different scenarios, I was able to investigate differences in sharing as a function of scenario type. Willingness to share in all scenarios were significantly and positively correlated with each other (r=0.40 to 0.96, Cronbach's α = 0.97).

I examined these similarities further by performing a hierarchical cluster analysis using the average linkage distance formula, a standard technique for examining groupings among items which Olson *et al.* also used in their analysis of privacy and sharing (Olson et al., 2005). I chose to use mean sharing per level of closeness as the input because of the strength of closeness in explaining the variance of sharing responses. The dendrogram in Figure 8 shows the clusters. The horizontal scale for the dendrogram is linearly related to the cluster distance at each point where a pair of clusters was merged. For example, in the middle of the dendrogram "hist:common hist" and "hist:I've been where you are" were more closely clustered than the next two "hist:everywhere traveled" and "loc:on vacation": this is indicated with the horizontal distance, with the first cluster formed closer to the right side than the second one. Note that the scenario names are shorthand for the scenarios in Table 3.

The three clusters in the dendrogram can be roughly labeled as categories of scenarios: 1) scenarios with information about something that the participant and friend have in common (see Figure 8 top, e.g. loc:within 1 mile); 2) location-history-related scenarios (see Figure 8 middle, e.g. hist:everywhere traveled); and 3) scenarios that reveal sensitive information (see Figure 8 bottom, e.g. loc:always).

To ensure that the means for willingness to share were in fact significantly different across clusters, I performed a Tukey-Kramer HSD across all of the means (see Table 3; there was no significant difference across scenarios that are connected by the same letter). This revealed 13 groups (some of which overlap) of scenarios with no mean difference. Table 3 shows that the seven highest-mean sharing scenarios all involve sharing personal information that has something in common with the friend's information, for example shared calendar events or location proximity with the friend.

3.1.3 Discussion

The main focus of this study was to understand which of the collected features are most useful for predicting individual sharing preferences, with the ultimate goal of being able to automatically predict sharing preferences from that information. The results show that the simple 1-5 Likert scale for closeness was clearly the most useful feature for predicting sharing, outperforming grouping and all other models that do not include closeness.

Despite the relative success of closeness as a predictor when compared with life modes, the literature has favored privacy controls that focus on grouping (Danezis, 2009; Fang & LeFevre, 2010; S. Jones & O'Neill, 2010). In addition, commercial OSNs all seem to either provide grouping controls (e.g. Facebook and LinkedIn), or else require users to specify sharing preferences on a per-friend basis (e.g. Google Hangout's "send my location" feature). While a grouping paradigm does not prevent individuals from constructing groups based on closeness, it may be more useful to explicitly ask users to do so.

One advantage of using closeness to aid in the specification of sharing controls is that closeness is ordinal; providing the closeness for two friends also indicates if one is closer than the other. In contrast, a weakness of group-based privacy controls there is no natural ordering between groups; they are nominal. The ordinal nature of closeness can be useful for expressing privacy controls, as users could simply express "don't share with anybody below medium closeness" (closeness = 3). Closeness can also support tiered rules, such as "closest friends (5) can always see my location, medium-close friends (3 and 4) can only check up to twice a day, nobody else (1 and 2) can see it without requesting."

Additionally, closeness is a useful intermediate between communication frequency and sharing controls because it offers intelligibility to users (i.e. a user can understand that a sharing preference was specified based on closeness, and even fix incorrect inferences of closeness). This also benefits any other applications that might use closeness for various features: closeness ratings will be improved for all applications.

3.1.4 Limitations

One limitation of this data is that it is entirely self-reported. Additionally, further work is required to demonstrate the real-world application of these findings. By conducting the study online and anonymously, experimenter effects were likely minimized. Furthermore, individual self-report data is the ground truth on some measures such as felt closeness. However, some of the participants' answers may have been idealized responses (e.g. people they call less frequently than reported), or participants may have been unable to answer thoughtfully for every sharing scenario (e.g. cannot answer for all places I've been to).

3.2 Using Communication Data To Infer Tie-Strength

According to social science theory, features of communication such as frequency of contact (Granovetter, 1973) and communication reciprocity (Friedkin, 1980) are reliable proxies for tie strength, and these have been increasingly used as proxies for tie strength in the research literature. Following the findings of the previous study, that self-reported tie strength predicts sharing preferences, the goal of this next study was to connect communication behavior to sharing preferences, using automatically inferred tie strength as an intermediate step in that chain. Since communication behavior should predict tie strength, and tie strength was just shown to predict sharing preferences, the results of this study were expected to be straightforward. Instead, the main result was surprising: communication behavior was not a reliable predictor of tie strength, in particular for strong ties.

3.2.1 Method

How well can tie strength be inferred from contacts, call logs, and SMS logs? These data sources can be found on nearly every smartphone, and I chose them to validate an assumption in the research community that communication frequency and duration from these channels can work as an effective proxy for the strength of a relationship. Further, I planned to use the inferred tie strength to predict sharing preferences. I collected data from participants' Android smartphones and asked them to manually categorize and rate their relationships with individual contacts as ground truth for tie strength.

Participants

I recruited 40 participants (13 male and 27 female) living throughout the United States by posting ads in several places: on Craigslist in 6 major US cities, on a nationwide site for recruiting study participants, on a website for posting social relationship research studies, and on a participant pool within our university. Participants met three selection criteria. First, to avoid privacy concerns with minors, participants had to be at least 18. Second, to focus on people who could benefit from a more computationally sophisticated representation of relationships, participants had to use Facebook and have at least 50 friends through the service. Third, to ensure a sufficient amount of log data, participants had to have used the same Android phone for at least six months prior to the study. 55% of the participants were students (graduate or undergraduate), 35% were employed in a variety of professions, and 10% were unemployed. Participant ages ranged from 19 to 50 years (*mean* = 28.0 years, σ = 8.9). Participants were instructed to complete the ground truthing within two weeks, and were compensated $80 USD. Of the 40 participants, four were excluded from our analysis: each had fewer than two weeks of data and fewer than 100 phone calls. Findings are based on the remaining 36 participants.

Procedure

Participants downloaded an Android app that copied their contact list, call log, and SMS log to a database file. Participants then uploaded this file, in addition to their Facebook friends list, to the study server through a custom website that was designed for this study. The entire study was conducted through this website. Participants could stop and resume whenever they wanted, and were given two weeks to complete the entire process. By default, Android phones limit the call log to the last 500 calls and typically have a default limit of 200 SMS messages per contact. This resulted in broad differences in how many days the logs represented (range: 21-369; median: 80; mean: 108).

Participants' contact and Facebook lists were much too long for participants to completely ground truth. Through pilot testing, we found 70 contacts to be a reasonable number for participants to rate before the task became overly burdensome. To maximize participant retention, participants were asked to rate 70 contacts.

The vast majority of any individual's contacts will be weak ties. However, for this study it was necessary to collect information on strong ties as well. To ensure that strong ties were included in the list of 70 contacts, participant generated a list of contacts that fit specific social categories, regardless of their appearance in the phone contact list or Facebook list. Participants listed five people in each of the following categories: *immediate family*, *extended family*, *people they live with*, *coworkers*, *people they feel close to*, and *people they do hobbies with*. Past qualitative work suggests these categories will contain an individuals' strong ties (McCarty, 2002; Spencer & Pahl, 2006; Wiese, Kelley, et al., 2011). This process resulted in approximately 25 unique names per participant (some names were repeated across the categories). In addition, each participant's top 15 contacts with the highest communication frequency for calls, SMS, and Facebook were included in the list. The characteristics of the contacts on the list allow for an examination of the assumptions that communication is a direct proxy for tie strength: participants provided ground truth data for all of their high-communication contacts, and also for all of their self-reported strong ties. If call and SMS communication is a perfect proxy for tie strength, these two groups should be the same.

The final list of 70 contacts was comprised of the category list and the frequency list, after removing duplicate names. In cases where this process yielded fewer than 70 contacts, I added randomly selected contacts from the participant's phone's contact list and Facebook friend list. Afterward, participants manually inspected the list for duplicates, since automatic detection using contact names alone does not reliably identify all duplicates (Wiese et al., 2014). This process repeated until each participant had a list of 70 distinct contact names (hereafter called the *70-person* list).

Participants provided demographics for each contact in the 70-person list, such as sex, age, and relationship duration. Participants also answered four questions about their relationship with each contact, adapted from (Marin & Hampton, 2007):

1. How close do you feel to this person?
2. How strongly do you agree with the statement "I talk with this person about important matters"?
3. How strongly do you agree with the statement "I would be willing to ask this person for a loan of $100 or more"?
4. How strongly do you agree with the statement "I enjoy interacting with this person socially"?

Participants answered questions using a discrete 5-point scale, following previous work on tie strength (J. M. Ackerman, Kenrick, & Schaller, 2007; Burke, 2011; Cummings, Lee, & Kraut, 2006; Roberts & Dunbar, 2011). I used a discrete rather than continuous scale to reduce cognitive load and fatigue – participants provided a large amount of data for many contacts, and a continuous slider may have been an additional burden. To protect privacy, I did not collect the content of SMS messages. However, I did collect descriptive information such as email domain name, first six digits of phone numbers, and city/state/zip code.

3.2.2 Dataset

The dataset consisted of logs for 24,370 phone contacts, 16,940 calls, 63,893 SMS messages, and 1,853 MMS messages. Note that Android phones can be set to automatically sync the phonebook with online contact lists (e.g. Gmail and Facebook), so phonebooks may have included these contacts in addition to ones entered manually.

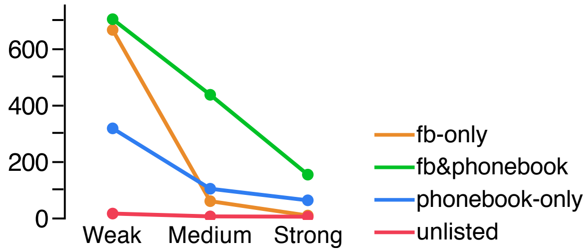


Figure 9. Total number of friends within each tie strength level across all participants, separated by the number of contacts who only appeared in the contact list, only in the Facebook friends list, appeared in both, or neither. The data indicates that there are a notable number of strong ties that appear only in the phonebook and not in Facebook, but there are few strong ties who appear only in Facebook and not in the phonebook.

3.2.3 Tie Strength and Basic Properties of the Dataset

As a first step to explore the validity of using information available on a smart phone (contact list, call logs, and SMS logs) to infer tie strength, I analyzed participants' answers for the four tie strength questions (questions 1-4 listed in the procedure section). The questions were highly reliable (α = 0.91), so I added all four responses together to form a scale. This is a standard practice that increases the reliability of a measure (Gliem & Gliem, 2003). Using the scale, I generated a ranked list of each participant's contacts based on relationship strength.

Next I partitioned each participant's contacts into three levels of tie strength. I explored several approaches for identifying these levels. An assessment of the distribution of Z-scores from the combined tie strength metric both across all participants and per-participant revealed no obvious gaps in ratings on which I could split strong and weak ties. Instead, I based these levels on previous work by Zhou *et al*, which finds that "rather than a single or a continuous spectrum of group sizes, humans spontaneously form groups of preferred sizes organized in a geometrical series approximating 3–5, 9–15, 30–45, etc." (Zhou, Sornette, Hill, & Dunbar, 2005). They found that the top group represents a person's closest relationships (support group), and the second group represents the next closest set of relationships (sympathy group). The larger sized groups of 50 and 150 people are considered to be less stable, and are referred to as clans or regional groupings.

In constructing each participant's 70-person list, I took multiple steps to increase the likelihood of capturing many of a participant's closest contacts. Therefore, since the 70-person list likely included the majority of a participant's strong ties, I assigned the contacts into their respective groups based on the numbers from Zhou *et al.* By identifying relative tie strength for contacts within each participant instead of setting absolute ratings as a cutoff points, I normalized out individual differences between participants (e.g. a tendency for some participants to use 3 as the baseline and others to use 1, or a participant's negative reaction to a particular question).

I partitioned each contact list into three groups:

- ***strong tie*** - the top group (rank 1-4)

- ***medium tie*** - the middle group (rank 5 – 19)
- ***weak tie*** - the remaining contacts

In cases where multiple contacts tied for a rank, all of those contacts were assigned to the same tie strength level, resulting in a slight variation in group sizes per participant.

With these tie strength groupings, I began to investigate communication patterns as a proxy for tie strength. First, I discuss simple features and their relationship to the tie strength groupings. Next, I describe machine learning models for inferring these tie strength levels.

Contact Source and Tie Strength

The properties of the *70-person* list allow me to estimate an upper bound for the percentage of a user's close contacts who could be detected from the two contact sources: only Facebook, only the contact list, or both. As Figure 9 shows, overall 99% of people on the *70-person* list showed up in either a phonebook or Facebook list (range: 95-100%, med: 100%). Overall, 19% of contacts existed only in the phonebook (range: 4-57%, med: 18%); 29% were only in Facebook (range: 0-56%, med: 31%); and 51% were in both (range: 20-90%, med: 52%). Looking across the tie strength categories reveals distinctive trends. I used Spearman's rho (ρ) to measure the non-parametric correlations between tie strength group and presence in the phonebook and Facebook friend list. Being a Facebook-only contact was negatively correlated with tie strength ($\rho=-0.32$, $p < 0.001$). Being a phonebook-only contact was not correlated with tie strength ($\rho=0.03$, n.s.), although percentage-wise, more of the closer contacts were only in the phonebook. Being a phonebook-and-Facebook contact was positively correlated with tie strength ($\rho=0.27$, $p < 0.001$).

The red points in Figure 9 represent the 21 people that were neither in the phonebook nor Facebook list. They were people whom participants identified as immediate and extended family members, housemates or roommates, or people they worked with, felt close to, or did hobbies with. The orange points in Figure 9 represent Facebook-only contacts and the blue points represent the phonebook-only contacts. 29% of contacts would be missed if using a phonebook-only list to classify

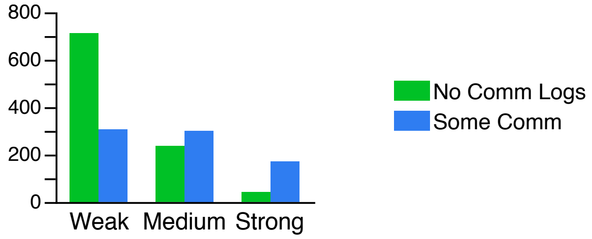


Figure 10. Number of friends in the mobile contact list who exchanged zero (No Comm Logs) vs. at least one (Some Comm) SMS or call with our participants (determined from call log data). There are a number of strong ties with zero communication logs in the dataset. Any classifier that is based on this communication behavior will misclassify those strong ties as weak ties. This issue is even more pronounced for medium tie-strength: nearly half of those contacts have no communication in the collected dataset.

tie strength and 19% would be missed if using a Facebook-only list. Both a Facebook-only and a contact-list-only approach would miss some strong ties; however, the Facebook-only approach would miss a notably larger number of strong ties (29% vs. 4%).

Tie Strength and Phone/SMS Communication

To establish an upper bound for the accuracy of inferring tie strength from phone and SMS communication, I divided the phonebook contacts into two groups by communication history (none vs. some). A reasonable baseline expectation would be that contacts with no communication history would have weak tie strength. Figure 10 shows that most contacts with at least one communication in the dataset have higher levels of tie strength. Additionally, as the tie strength level increases, the percentage of contacts with some communication with the participant also increases (ρ=0.35, p < 0.0001). Still, several contacts with strong tie strength have no communication history in the dataset. Thus, attempts to classify tie strength using only call and SMS data could not correctly classify these contacts.

Having at least one communication in the call and SMS logs increases the likelihood of a contact having higher tie strength. However, this is not an absolute rule: there are counter-examples in both directions - strong ties without communication history and weak tie contacts with it.

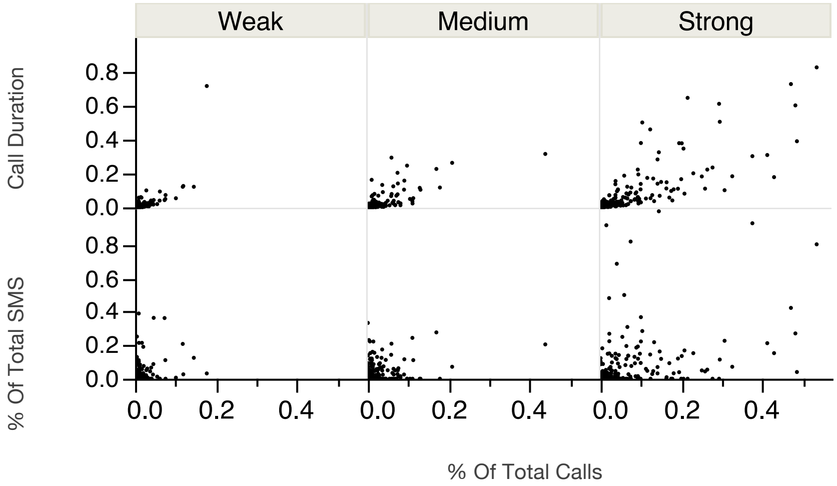


Figure 11. A grid of six plots showing communication frequency and total talk time. The top 3 graphs plot each contact's aggregate call duration (y-axis) against number of calls (x-axis). The bottom 3 graphs plot each contact's number of SMS messages (y-axis) against number of calls (x-axis). For both top and bottom, the columns separate the contacts by tie strength group. The graphs include data for contacts with at least one call or SMS. All numbers are represented as the percentage of a participant's total communication frequency/duration.

Next I explored the relationship between communication frequency and duration with respect to tie strength. Figure 11 shows six plots in a grid, with each dot representing a contact in the dataset. The graphs in the top row show aggregate call duration (y-axis) against the total number of calls (x-axis) for each contact. The bottom row shows the total number of SMS messages (y-axis) against the total number of calls (x-axis) for each contact. Each column indicates the contact's ground truth tie strength level. Both aggregate duration and frequency are represented as a percentage relative to the total call duration or number of calls/SMSs per participant. I expected some close contacts (appearing in the two graphs on the right column) to stand out with long call durations (high y-axis value), and others to stand out with high frequency (high x-axis value) when compared with medium tie strength contacts (middle column) or low tie strength contacts (right column). For example, a person might call an old friend infrequently, but chat for a while each time. Conversely, one might regularly make short calls to a roommate to coordinate.

As expected, contacts with more frequent or longer duration communications were more often in the higher tie strength levels. Number of calls, duration of calls, and number of SMS are all positively correlated with tie strength ($\rho = 0.42$, 0.43, and 0.20, all $p < 0.0001$). Surprisingly, many people in all tie strength levels had very little communication. Weak ties generally had few calls and short durations. For strong ties, the ranges increase for number and duration of calls, with a clump of few-and-short contacts.

Summary of Simple Features

This section established a basic upper bound of accuracy for inferring tie strength with smartphone communication logs. The data shows that using Facebook as the only data source would miss 29% of strong ties, either because they are not Facebook friends, or because these contacts do not use Facebook at all. Next, there are some strong ties without any record of communication within the phone logs. Finally, while communication frequency and duration of calls can help indicate strong tie strength, low frequency and duration are not clear indications of weak tie strength.

These trends are consistent with tie strength theory: more communication on more channels indicates a strong tie. However, our dataset has a number of counterexamples, pointing to critical challenges for automatically inferring tie strength from communication behavior.

3.2.4 Classifying Tie Strength

While the above findings already indicate significant issues for using call and SMS logs to indicate tie strength, perhaps a combination of more subtle features than frequency and duration might indicate tie strength. To explore this prospect, I developed several machine learning models to classify tie strength based on call and SMS log data.

Features Used for Inferring Models

I defined a total of 153 machine learning features: 17 from the contact list, 66 from call logs, 36 from SMS logs, and 34 from combined calls and SMS. These features are based on (Min, Wiese, Hong, & Zimmerman, 2013), and more details on the specific features can be found in that paper. These features include:

- *Intensity and regularity:* The number of and duration of communications has been used to infer tie strength in past work (Hill & Dunbar, 2003; Roberts & Dunbar, 2011). I modeled this factor using features such as total number and total duration of calls.
- *Temporal tendency*: In their friends-acquaintances work, Eagle, Pentland, and Lazer observed the temporal tendency in contacting people (2009). For example, calling particular contacts at different times of day and days of the week.
- *Channel selection and avoidance*: People favor a certain communication medium based on the person they are communicating with (Mesch, 2009). I modeled this using features such as the ratio between SMS and phone calls.
- *Maintenance cost*: Roberts and Dunbar (2011) found that people apply different amounts of effort in maintaining different kinds of relationships. This effort is measured with the time to last contact. To model maintenance cost, I used the number of communications in the past two weeks (short-term view) and in the past three months (longer-term view).

Inferring Tie Strength Using Communication Logs

Using all of the features described above, how well can a model infer tie strength? The nature of tie strength poses a challenge for building this model. Tie strength could be treated as a numeric class value based on the answers to the tie strength questions. However, the difference between a rating of 1 and 2 is not necessarily equal to the difference between a rating of 2 and 3. Additionally, early iterations treating tie strength as a continuous value tended to push scores closer to the middle, with very few people classified as being weak ties. Therefore, I used the tie strength levels of *very strong tie, medium strong tie,* and *weak tie* as nominal class values in these models.

I evaluated the models using the Weka Toolkit's ("Weka 3: Data Mining Software in Java") implementation of a support vector machine (SMO). I conducted a leave-one-participant-out cross-validation (each fold contained data from one participant). This prevents any anomalies within a particular participant's data from causing a performance overestimate. I trained 9 models, varying two aspects of input data. First I varied what the model was classifying (First column of Table 4):

- ***3-class***: classifies as very strong, medium-strong, or weak
- ***2-verystrong***: binary classifier that combines medium strong and weak ties into one class, with very strong as the other class
- ***2-mediumstrong***: binary classifier that combines very strong and medium strong ties into one class, with weak ties as the other class

				Strong ties		Medium strong ties		Weak ties	
Class Condition	Dataset	Accuracy	Kappa	Precision	Recall	Precision	Recall	Precision	Recall
3-class	all	69.60%	0.279	0.503	0.399	0.399	0.209	0.759	0.907
3-class	contactlist	61.11%	0.251	0.491	0.423	0.414	0.242	0.677	0.845
3-class	somecomm	46.28%	0.179	0.449	0.473	0.440	0.425	0.496	0.498
2-verystrong	all	91.55%	0.361	0.537	0.323	➡		0.937	0.973
2-verystrong	contactlist	88.64%	0.361	0.545	0.343			0.914	0.961
2-verystrong	somecomm	75.46%	0.297	0.480	0.432			0.829	0.855
2-mediumstrong	all	75.00%	0.367	0.693	0.420	⬅		0.764	0.920
2-mediumstrong	contactlist	68.06%	0.317	0.683	0.460			0.680	0.843
2-mediumstrong	somecomm	63.11%	0.192	0.707	0.724			0.488	0.467

Table 4 The results of 9 classifiers constructed using SMO. The prediction classes are tie-strength categories. For 2-verystrong, the medium strong and weak tie strength classes are combined and for 2-mediumstrong the medium strong and very strong tie strength classes are combined.

I also varied the input data for the classifier (Second column of Table 4):

- **all** includes all contacts on the 70-person list
- **contactlist** includes only contacts from the 70-person list who appear in the user's phonebook (see Figure 9)
- **somecomm** includes only contacts from the 70-person list with at least one logged SMS or call (see Figure 10)

Classification results vary considerably (Table 4), ranging from 46.28% (κ=0.179), to 91.55% (κ=0.361). The Kappa statistic measures the agreement between predicted and observed categorizations, correcting for agreement that occurs by chance. Table 4 The results of 9 classifiers constructed using SMO. The prediction classes are tie-strength categories. For 2-verystrong, the medium strong and weak tie strength classes are combined and for 2-mediumstrong the medium strong and very strong tie strength classes are combined. These results reveal clear trends. First, within each of the class conditions, classifiers perform best for *all*, second best for *contactlist* and worst for *somecomm*. Figures Figure 9, Figure 10, and Figure 11 provide some insight into these results. Most of the contacts who are not in the contact list (thus excluded from *contactlist* models) or who have no communication history (thus excluded from the *somecomm* models) are not strong ties, and thus are easier to classify. As a result, the models that include them perform better.

The most successful class condition is *2-verystrong*, followed by *2-mediumstrong*. *3-class* performs the worst. This is typical of multi-class models, which usually take a performance hit compared to binary classifiers.

More often than not, the models classified strong ties incorrectly − they were more likely to classify a strong tie as a weak tie than as a strong tie (in Table 4, the recall values for the strong tie class are the percentage of strong ties correctly classified, and are under 50% for all but the *2-mediumstrong* model). Also, about half of ties that were classified as strong were actually not strong (in Table 4, the precision values for strong ties is the percentage of contacts that were classified as strong ties who were actually strong ties − they are under 55% for six of the nine class conditions). The plots from Figure 11 offer insight into these errors. These misclassifications emphasize the weakness of using call and SMS logs to infer tie strength, and thus the problem with using those logs as direct proxies for tie strength. This result is even more pronounced in recall values for the strong tie class of the *2-verystrong* models in Table 4. The *2-verystrong-all* model, which has the best overall accuracy, only detects 1/3 of strong ties correctly.

3.2.5 Error Analysis Participant Interviews

Motivated by the particularly low recall of the *very strong* tie class in these models, I conducted semi-structured interviews with 7 of the participants. For each participant, I selected 5 to 10 contacts they had labeled as strong ties that were misclassified as weak ties (58 contacts total). I focused on this type of misclassification based on an error analysis of the data. In the error analysis, I referenced tie strength theory to consider communication expectations for medium and weak ties. People do not *only* communicate with strong ties, so the presence of some communication with weak ties is reasonable. However, if participants had more communication with more of their strong ties, the model would have been better able to distinguish between strong and weak ties. This led me to focus on very strong ties with little or no communication (who were misclassified as weak ties), rather than weak ties with some communication (who were misclassified as very strong ties).

Interviews took place over the phone, lasted about 30 minutes, and were recorded to facilitate note taking. I asked participants open-ended questions about the nature of their relationship and communication with each selected contact:

- When and how did you meet this person?
- What led to this being a close relationship?
- Has anything changed between the time that you became close and now?
- Was there anything different about the channels that you used to communicate with this person or the frequency of communication that you used with this person between then and now?

I iteratively coded participants' responses about each contact for themes to provide insight into the misclassifications. Several themes surfaced that help explain the discrepancy between communication frequency and tie strength. I present them in two categories: *Communication Channel* and *Relationship Evolution.*

Communication Channel

We used to talk on the phone more when we first became close (7 of 58 contacts). In these cases, participants indicated that they used to speak on the phone more frequently, but do so less frequently now, mostly just to catch up. In some cases, this seemed to be a result of a change in life stage (either for the user or for their contact) and/or a change in their geographic location, replicating findings from prior work (Spencer & Pahl, 2006). For example, one participant complained that he used to keep up with a friend much more regularly before that friend got married, and now they hardly speak at all. Change in life stage and change in geography are discussed further in the *Relationship Evolution* section below.

Other contacts in this category appear to be in relationships in decline, yet the feeling of closeness lingers. One participant spoke about reaching out to a friend multiple times without reciprocity: "I'd like to be friends, but it doesn't work unless we both put in the effort."

In-person communication (11 of 58 contacts). Participants also identified contacts whom they mostly interacted with in person. A contact's close proximity to the home seems to play an important role in tie strength. One participant described talking to her neighbor opportunistically, when they see each other. Another detailed how she spoke with her 11-year-old son regularly, just not over the phone. Three participants described friends from classes and their dorm with whom they spoke when they saw each other.

Extended family often fell into this category. Many participants reported primarily speaking with parents, siblings, and other family members in person. In one case, a participant reported going to her parents' house a couple times per month, but mostly not calling her dad on the phone. In these cases, lack of communication logs did not mean lack of effort in maintaining the relationships. In discussing these contacts, some participants specifically mentioned making an effort to travel once a year to see each other, or making a special effort to get together when they do happen to be in the same place.

Other communication channels (25 of 58 contacts). For some strong ties, participants noted that they communicate regularly, but not via phone calls or SMS. For several participants, communication with a contact happened almost exclusively using Facebook. Other participants used instant messenger, email, Skype, or SMS replacements such as WhatsApp to stay in touch with close contacts.

Relationship Evolution

Different location or different life stage (27 of 58 contacts). When asked what was different about their relationship between when they became close and now, many participants responded immediately that either they or their contact had moved. As in the literature (Spencer & Pahl, 2006), participants said that with the change in geography, the communication frequency had changed, but not the perception of closeness. The move was often triggered by a change in life stage (e.g., going to college, graduating, getting a new job). However, even without moves, a significant life stage change could trigger a communication change on its own (e.g. getting married or having a child).

Family is close regardless of communication (17 of 58 contacts). Many misclassified participants were family members. Several participants described specific familial relationships from the perspective of obligation, which hinted at a greater underlying complexity. For example, one participant said that she refused to take her grandmother's phone calls, stating that she calls too frequently and repeats herself. Yet, the participant still reported feeling very close to her grandmother. Another participant, the mother of an 11 year old, said "of course I am close to him," but that it is not necessary for them to talk on the phone. Another participant said her uncle was "definitely close, but he's different from the other close people. He's that really strict uncle that wants to tell me how to live my life, so I don't talk to him too much, maybe every couple months."

Interview Summary

These interviews highlight the limited effectiveness of the tie strength models. One issue that limits the effectiveness of these models is the way that relationships change over time. In particular, the circumstances under which two people became close are not necessarily the same as the current circumstances of the relationship, even if the two people remain close. Since the communication logs only capture relatively recent behavior, they do not contain the data that would indicate a strong long-term relationship. The other main component that limits these models' effectiveness is that much interpersonal interaction occurs outside of phone calls and text messages, including communication in other media as well as face-to-face communication. Call and SMS-based models do not account for these interactions.

3.2.6 Discussion

This section (3.2) investigates the growing practice of using communication frequency and duration as a proxy for social tie strength. While social psychology theory holds frequency and long durations across all communication channels as indicators of strong ties, the research community has used behavior across a few communication channels and over relatively short time windows as a tie strength proxy. This study examined if the call and SMS logs stored on a smartphone held enough information to infer tie strength.

Communication Is an Indicator of Tie Strength, But...

These results support the tie strength theory literature, showing a strong relationship between tie strength and communication patterns (Gilbert & Karahalios, 2009; Roberts & Dunbar, 2011). Higher levels of communication frequency, call duration, and, in particular, communication initiated by the phone's owner are all indicators of a strong tie. However, when operationalizing this theory with call and SMS logs, the signal is very noisy. Low levels of communication do not accurately identify weak ties: participants had many strong ties who they rarely called or SMSed. The interviews probing strong ties with little communication revealed several explanations for this pattern, each of which pose fundamental challenges for inferring tie strength.

First, a person's communication via phone and SMS does not capture all of their communications. Interactions happen through many other channels (e.g., Skype, instant messenger, landline phones), in some cases replacing communication via phone or SMS. Second, face-to-face communication remains a primary form of communication for some very close contacts, but capturing this kind of communication is difficult with current technology. Third, strong ties may form in an earlier life stage and persist across stages even as communication frequency diminishes. Even if one could capture data across multiple channels and do so for long periods of time, it is not clear that this would be sufficient to improve the models of tie strength.

A breadth of recent and highly-cited research has assumed that call and SMS behavior is a good proxy for tie strength (Conti et al., 2011; Miritello et al., 2013; Onnela et al., 2007; D. Wang et al., 2011). These contributions do not attempt to identify all strong ties exhaustively. Rather, they only identify strong ties who use a specific communication channel. Our *contactlist* and *somecomm* datasets best match this task. The models for these datasets produce similar errors, and also indicate that communication frequency and duration are an incomplete signal for determining tie strength. While theory supports the relationship between communication frequency and duration and tie strength (Hill & Dunbar, 2003), these communications should not be operationalized only through the call and SMS logs stored on a person's phone.

Alternatives for Identifying Tie Strength

Researchers looking for a way to separate strong ties and weak ties need to consider alternatives to using short term communication logs from one or two channels, such as those available of today's smartphones.

One alternative is to collect data from more communication channels. This approach has several challenges. First, beyond the most popular additional sources (i.e. email, Facebook), researchers are likely to face diminishing returns when adding additional data sources. For example, some people use Skype, while others use Google Hangouts. Similarly, there are many text message replacement apps (e.g., WhatsApp, GroupMe, Kik). The number of communication channels is growing, people have different preferences for which channels they use and for what purposes, and people switch between services based on fads, or on what services their friends are using. Second, many of these services offer no API for accessing log data. Third, correctly linking contact identities across multiple communication sources is non-trivial and error-prone.

Another way of augmenting this process while still using communication data to separate strong and weak ties is to use *a lot more data*: data that extends back to when close relationships first began, which could be on the order of years or even decades. Since this data does not exist for current close relationships, the only way to evaluate this method would be to start collecting the data now and see if it predicts the presence of strong ties, which may only be formed several years from now. Current data collection and retention practices are not conducive to long-term data collection. For example, Android devices by default only store the last 500 calls and 200 SMS messages. Furthermore, there are no standard APIs to access one's data, and no unified structures for storing user data and maintaining history as users change devices and services. For work on long-term communication history to be possible, these practices will have to change.

Investigating message content might also help to improve the separation of strong and weak ties. It is possible that in cases where there is some communication, the content of the communication with strong ties is different from weak ties in a systematic way. A drawback to this approach, and the reason that we did not explore this avenue, is that many people are uncomfortable with the privacy implications of granting content level access to calls and SMS.

Another approach is to differentiate relationship-maintenance communications with strong ties (which can be infrequent but very important) from other types of communication. One way to do so is to see whom a person calls or visits when traveling (factoring in time of day to differentiate between a likely work contact versus a social contact). Another way might be to use age or the inferred life stage of individuals and incorporate that into tie strength models. For instance, college students, 40-year-old parents, and senior citizens likely have different kinds of people in their strong ties. This method would require much deeper investigation into how people's friendships change over time and how life stage affects these relationships.

The most reliable option for distinguishing strong and weak ties is to include users in the process through interviews (Spencer & Pahl, 2006), or a survey (as I did). Some research has considered computer supported tools for collecting this kind of data (Ricken, Schuler, Grandhi, & Jones, 2010). The primary challenge here is that, even in the case that labeling is efficient, this approach still requires the time and effort of the user.

The primary drawback to all of these approaches is that they require data that is hard to obtain. In general, researchers who use communication frequency as a tie strength proxy do so because it is easily available. Many of the research datasets that are being analyzed were collected and anonymized for a different purpose, often by a third party such as a telecommunications company. Researchers using such datasets do not have the possibility of collecting more data, or have any access at all to the actual participants. Furthermore, many of these datasets contain data from far too many users for a non-automated approach to be possible.

Using Communication Frequency as Tie Strength

Researchers will likely continue to use communication frequency as a tie strength proxy because, with the rise of smartphones, the log data is increasingly available. Here, I offer some implications for those that make this choice.

Researchers should carefully consider how the imperfect proxy of communication frequency as tie strength limits the strength of their claims. A strong tie might have some in-channel communication (meaning that they would be included in the experiment), but may still have less communication in that channel than some weak ties – does this hurt the strength of a claim being made on that data? It will depend on the claims being made, and to what extent those claims rely on a clear separation between strong and weak ties.

One solution for researchers in these situations is to modify their claims so that instead of relating claims to *tie strength*, they relate the claims directly to *communication frequency*. For example, the existing work (Conti et al., 2011; Miritello et al., 2013; Onnela et al., 2007; D. Wang et al., 2011) that equates tie strength and communication frequency are valuable contributions. However, their findings are explained directly in the context of tie strength, which over-estimates the reliability of inferring tie strength from communication frequency. This can negatively impact the reader's ability to correctly interpret their findings. If tie strength is important to an argument, researchers should also explain how they believe tie strength and communication frequency are related to each other within their dataset, and should explicitly identify that communication frequency is a limited proxy.

This work has not yet explored the possibility of systematic per-user differences based on demographics, behavioral characteristics, or life stage that may affect classification accuracy in separating strong ties from weak ties. If any such effects exist, they may affect the claims that can be drawn from using communication frequency to classify tie strength. Similarly, communication frequency may be useful for detecting other dimensions of interpersonal relationships. In turn, the influence of per-user differences and other dimensions of personal relationships may further the definition of tie strength and the understanding of the nuances of tie strength as a concept.

3.3 Case Study Discussion

The goal of the studies presented in sections 3.1 and 3.2 was to use the communication behaviors of users with their contacts to predict their preferences for sharing different kinds of information with those contacts, using tie strength as an intermediate representation. At the outset this logical chain seemed well supported by theory, especially since communication behavior is known to predict tie strength. However, operationalizing this theory revealed fundamental challenges for working with personal data.

First, obtaining the participants' communication data was a difficult process. It required me to write custom applications to scrape the data from participants' devices and additional effort to gather and process the data from Facebook. Collecting data from more sources would have significantly increased the complexity of this task. The process for transforming this data so that it could be used in the machine learning models also required linking the data across the two different data providers (phone and Facebook) and merging duplicate contacts. Since simple name matching did not identify all duplicates, merging duplicates required significant manual effort both by the researchers and by the participants.

Further, the type and amount of data available varied widely by participant. Android restrictions limited the total number of call logs (and SMS logs for some participants as well). For other participants the data was limited by how recently they had obtained their current phone, because call and SMS logs are not automatically synced from an old device. In some cases this meant the timespan of the dataset did not cover enough time to make the tie-strength inferences. Additionally, the irregularity of the contact list entries and the amount of contact data that was completed made it practically impossible to use data from the contact lists for any of the model's features at all.

Improving these models by including communication logs from additional data sources is not trivial. The effort need to collect, link, and merge the additional data sources would be equal to or greater than the effort needed to do so for the original data, especially since applications for scraping data are not re-usable across data sources. In the two studies presented in this chapter, manual steps were required, both for the researcher and the participant.

Finally, the models developed here still offer potential value for inferring tie strength to some noise-tolerant applications where the cost of an incorrect inference is small. Unfortunately, even in this case deploying these models remains a significant challenge. This is in part because the entire machine learning pipeline here was static and one-off. Formatting the data, extracting features from the data, collecting ground truth, and classifying instances from the dataset all happened with limited automation to complete this research project. Furthermore, there is no mechanism to deploy inference models. Deploying this model (e.g. as an Android library) would require significant additional development effort to automate this process and generate a usable API.

Additionally, deploying a machine learning model of tie strength raises privacy concerns. In particular, inferring tie strength information for contacts requires permission to access the user's communication metadata. One solution would be to require that any applications using the library declare these permissions in their manifest. However, this may give developers pause: for some applications, adding permissions declarations for call logs, SMS logs, and the contact list in order to obtain tie strength may not be worth the additional scrutiny of a user. Some users may ultimately choose not to download an application that accesses too much data (J. Lin et al., 2012). It seems unnecessary for an application to need to require these permissions if all they are accessing is the resulting tie-strength inference.

As a case study, this work illustrates some of the challenges in using personal data to make high-level inferences: the availability of the raw data, the limited effectiveness of automated approaches for identifying duplicates, and dispersion of one behavior across many channels (in this case, communication across phone calls, SMS, and channels not captured in the study). Most importantly, personal data was found to be an unreliable indicator for a higher-level inference even in an area where strong theoretical work already existed. These challenges point to deeper issues that affect the way that personal data can be used. While they are illustrated with communication logs and inferences of tie strength, these challenges are not unique to this specific area: researchers and developers in many situations are sure to encounter the same issues.

4 A Conceptual Framework for Personal Data

The previous chapters have offered a broad set of insights that speak to the complexity of personal data from the perspectives of end users, researchers, and application developers. This chapter begins by examining the ecosystem of personal data today from a macro level, identifying breakdowns between stakeholders. Next the chapter offers a synthesis of issues highlighted here and in previous chapters to extract more general systemic issues with the ecosystem of personal data. This synthesis leads to a conceptual framework for understanding the breadth of personal data, a range of applications that could use that data, and the process for working with that data. The conceptual framework (described in section 4.4) consists of two components. The first component is a continuum of personal data (described in section 4.2) from very low-level (e.g. raw sensor data) to very high level (e.g. is the user experiencing major depression?). The second component is a set of three steps that are required to develop applications that depend on personal data (described in section 4.3). This framework serves as a boundary object to facilitate shared understanding of this domain of personal data and the process of working with that data to serve some client application. Finally, this chapter offers some design goals for improving the ecosystem of personal data from its current state to address the many issues highlighted throughout this thesis.

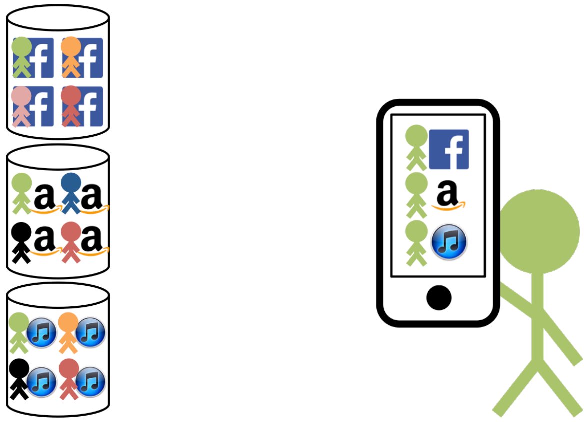


Figure 12: Personal data today is separated across the applications and services where each type of data originated (left). To unlock the full potential of personal data, it should instead be structured to prioritize the coherence of the heterogeneous data around each individual who is the subject of that data (right).

4.1 The ecosystem of personal data today

Despite its "personal" nature, personal data today is organized and stored separately within each service or application where that data was collected, rather than all of an individual's personal data being stored together. This "siloed" approach to storing personal data introduces significant problems. At a high level, these problems are:

- **Provides poor service**: Each service has an incomplete view of the user, which limits the service's offerings. It is impossible for the user to manage the access and usage of their data across these distributed silos.
- **Facilitates customer lock-in**: Users are bound to their services that hold their data, and leaving these services would cause the user to lose all of the value that they get from their data. For example, if a user wanted to stop using Netflix and start using Amazon Instant Video, he would have to leave behind the value of recommendations based on his viewing history.
- **Chicken-and-egg**: New services that rely on rich personal data are subject to a chicken-and-egg problem for procuring that data: the service is not valuable without the data, but the data is hard or impossible to obtain without the user using the service.
- **Ground truth data labels**: If a user tracks her location using one service and labels "home" and "work" in that service, those ground truth labels do not propagate to other services that the user wants to have access to that information (for example, a direction-finding service). This diminishes the value for the user to provide ground truth and further increases the challenges for leveraging personal data.

For an independent application developer to incorporate personal data into a new application today, she must follow all of the steps outlined below in section 4.3 and faces many decisions along the way. In all but the most trivial straw man examples, following this process requires a significant investment in development resources. Through this system, many applications of personal data are simply not feasible, or are even impossible. To make matters worse, because each developer solves these challenges on their own, these bespoke solutions are unlikely to be reusable for other developers. This further impairs a successful outcome when working with personal data.

4.1.1 Stakeholders

To fully understand the current state of personal data, it is important to acknowledge different stakeholders and their goals. Together, these stakeholders and their goals form an ecology of personal data. Considering the entire ecosystem is helpful for understanding why things are the way that they are today, and also what kinds of effects any changes in this ecology might have. The stakeholders include:

Data-logging companies and service providers: Products, services, and applications that generate logs of user data. This can include large companies that have many products (e.g. Apple, which includes iOS, OSX, Apple-written applications, iCloud, iTunes, the iOS App Store). This can also include small companies and companies with fewer products (e.g. Dropbox, Netflix). Truly any service that a person uses has the ability to collect rich data on that person's actions.

Data-consuming applications and services: These are services that take the user's personal data and apply it in some way to provide a service to the user. In many cases, one organization is both of these first two stakeholders (data consumers and data loggers). For example, Netflix collects a user's viewing data, and uses that data to make recommendations. However, services may also make use of many different kinds of personal data from a multitude of different data loggers.

End users: Everybody as individuals. These are the people whom the data is about and the people who are using these applications and services.

4.1.1.1 Relationship Between Data-Logging Services and Users

Data-logging services would not be meaningful without people who use that service. This relationship is mutually beneficial: users get to use the product or service and the services get the data that describes the individual's usage of the service. The data that users generate while using a service is inherently under the direct control of the data-generating service. Data-generators decide which data to collect and not to collect, how long to store collected data, and how accessible that data is to users and third-parties. Services typically do not give a user complete access to the data that has been collected about them. Even in notable situations where data is made easily accessible to end users (like Google Takeout[15]), there is still valuable data that is not included, but that the company still collects (e.g. a Google user's search history and Chrome browsing history).

[15] https://www.google.com/settings/takeout

A user's data is often essential to the business model of a data logger. In many cases, services are provided to the user for free or at a price that is below the amount of money that it costs the service provider to provide that service. This is made possible by selling information gleaned from the user's data (i.e. market insights), or incorporating that data into a service (e.g. facilitating targeted advertising by employing the user's data). In other cases, the user's data is what can differentiate a service to make it more attractive to a user (e.g. Netflix relies on a user's ratings and viewing history). As a result, some service providers are likely to behave in ways that users are unlikely to switch service providers. One way that service providers can do this is by locking a user into their particular service by withholding access to the user's data or limiting portability to different services.

Finally, privacy issues for users arise when data-generators capture data that users did not want to be captured and/or did not know was being captured.

4.1.1.2 Relationship Between Data-Logging Services and Data-Consuming Services

As mentioned above, in the current ecology of personal data, an individual service is often both a data-generating service and also a data-consuming service. For example, the dialer application on an Android Smartphone as a data-generator collects data about whom the user calls. As a data-consumer, the dialer application shows the user whom they have called most frequently and also most recently.

However, despite sometimes being the same entity, Data-consuming services are separate from data-logging services because data-consumers may also want to access data from a data-provider that is a different service. For example, the Android dialer may want to also use data from Facebook and Skype to show people that the user communicates with frequently across different communication media.

In some cases, data-generators charge data-consumers money for access to a user's data. For example, Facebook, Google, and many smartphone applications make a lot of money by leveraging a user's data to deliver targeted advertising. Here, privacy issues can arise for users when data is made accessible to data-consumers without the knowledge or explicit consent of the user.

4.1.1.3 Relationship Between Data-Consuming Services and Users

As we said above, many data-consumers are companies that are consuming the data that they created themselves as data-generators. Privacy issues arise when data consumers use data in a way that a user did not intend, or when the data reveals (either directly or indirectly) information that the user did not want revealed. One example is the recent story of a teenage girl who received advertising in the mail for maternity clothing and cribs based on her shopping habits, even though she had not told her father that she was pregnant[16].

In some cases, a user might wish to provide their own data to a data-consumer. For example, a user wants to use their data from one service to help personalize a different service. Another possibility is a user might want to donate their data to researchers that are going to analyze it. In general, users are fairly limited in their ability to do this.

There is typically very little communication/interaction between data-consumers from different organizations, (except in formal business relationships like Facebook & Advertisers from above), so if the user provides some information to a data-consumer (e.g. this location is my home), then the user probably needs to provide that information separately to other data-consumers, even if they were okay with that information being used by other data consumers as well.

4.1.2 Summary

The way that these stakeholders interact today is troubling and leaves much to be desired. Data-loggers wield a lot of power in deciding what data is collected, how it is stored, and who has access to it. With this power also comes the responsibility of maintaining the user's trust, and obeying laws. Data-consuming services want to provide their users the best possible service, which can rely in part on access to the user's data. Users want to receive the best services possible, but also want to be comfortable with how their data is being used, which requires a combination of transparency and trust. Finally, all stakeholders are seeking to minimize costs, even at the expense of other stakeholders (e.g. storing data and providing it through an API can cost money, so loggers may avoid it).

Examining the ecosystem of personal data in this way highlights clear problems with how personal data is managed today. The current ecosystem stifles innovation, facilitates lock-in, and offers a sub-optimal user experience. Considering these issues holistically offers the potential to improve personal data for all stakeholders.

4.2 The Personal Data Continuum

In this thesis, data is considered personal data if it describes something, anything, about an individual person: her behavior, her interests, her social relationships. So far in this thesis, personal data has been discussed as a single concept: either something is personal data or it is not. However, to fully engage how personal data is collected, stored, and used, it is useful to think about different kinds of personal data and how they are related to each other.

[16] http://www.nytimes.com/2012/02/19/magazine/shopping-habits.html?pagewanted=7

This section develops the idea that personal data can be thought of as falling somewhere along a continuum. This continuum is a core component of the conceptual framework. It ranges from very low-level data (e.g. a log of accelerometer data, latitude and longitude coordinates, audio levels) to extremely high-level data (e.g. my behaviors that do not support sustainability, the set of skills that I don't have which would be most beneficial to learn, an inference of the state of my mental health). Personal data can exist at various points along the continuum. The personal data continuum is intended to be continuous rather than discrete, however it's also important to keep in mind that continuum is a conceptual tool, not an absolute dimension. The personal data continuum described here is one of two components (the other being the steps introduced in section 4.3) of the conceptual framework for working with personal data (described in section 4.4). The following examples offer additional perspective into various points along the personal data continuum.

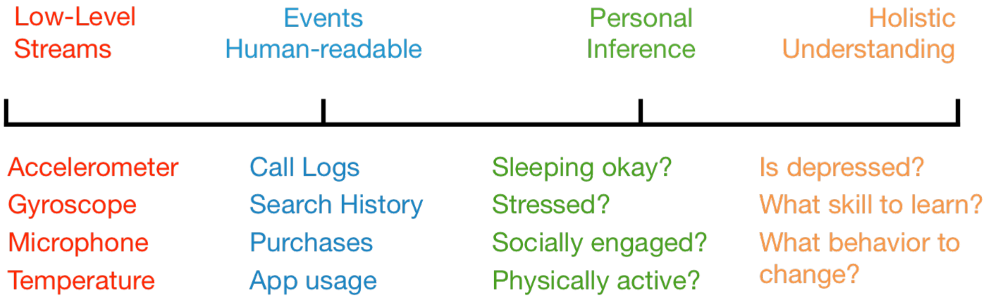


Figure 13: The personal data continuum ranges from very low-level data (far left side) like sensor data that describes the user's behavior and surroundings to very high level data (far right side) that describes information about individuals that they might not even know about themselves. Information in the lower levels can often be directly sensed, but data higher on the continuum has to be provided manually or inferred from a combination of lower level data.

4.2.1 Points along the continuum

Low-level data

Low-level data is often sensor data such as data produced from an accelerometer, light sensor, temperature sensor, and microphone, but this data might also be log data like key presses or mouse movements. One characteristic of this low-level data is that it typically does not mean very much if a human is looking at it on their own. For example, the readings from an accelerometer mean very little to a human without additional processing to interpret this data, usually on a time series.

At first glance, it's not always immediately obvious what low-level data is personal data and what low-level data is not. For example, accelerometer data from a smartphone can vary in its functionality as personal data: if a user is in possession of their smartphone then the accelerometer data describes something about the user's behavior. However, if the user has lent their smartphone to somebody else, the accelerometer is no longer generating personal data for the owner because that data does not describe anything about that person, instead the accelerometer is now generating personal data about the person currently in possession of the phone.

Person-Readable Logs

At this slightly higher level, personal data begins to have more clear meaning. The kind of data that exists at this level are often logs of user behavior from different applications: phone call logs, text messages, emails, browsing history, sleep logs, physical activity logs, purchase history, a log of places visited, media consumption history (music, news stories, TV shows, movies). The list of data that roughly fits into this category is very long.

This category is what most people likely think of when they think of personal data. In general the common concerns about personal data, privacy, and control relate to this kind of data. The data collected as a part of the NSA's infamous PRISM program[17] generally fits into this category.

Personal Inferences

At the level of personal inferences, personal data is less about individual moments in time and more about a general higher-level kind of knowledge about an individual, the kinds of things that change over the course of weeks, months, or even years. For example, social relationship data such as tie strength and life facet (Farnham & Churchill, 2011) from the previous chapters are examples of personal inferences. Other examples of these kinds of inferences might include: how physically fit the user is, how well she has been sleeping, or how social she has been. This level is completely removed from the set of things that can be instantaneously observed and automatically collected by a computer system.

Holistic Understanding

This category represents the upper limit of personal data. These are very high-level inferences that describe things about individuals that they might not even know about themselves. It is easier to think about the items in this category in terms of questions: Am I becoming depressed? What should I be when I grow up? What skill should I learn? How can I live more sustainably? What item should I purchase to make my life better? These are all questions that would require an incredible amount of data to answer. These questions require more information than simply personal data, but personal data is a very important component to the answers to these questions.

[17] The PRISM program is a clandestine surveillance program run by the NSA that collected large amounts of Internet communications. For more information see: http://www.washingtonpost.com/investigations/us-intelligence-mining-data-from-nine-us-internet-companies-in-broad-secret-program/2013/06/06/3a0c0da8-cebf-11e2-8845-d970ccb04497_story.html

One vision of the far-off future of personalized computing systems imagines that technology might be able to answer these questions for users automatically, or at least help lead people to these answers themselves.

4.2.2 Detecting Depression: Tracing an example through the continuum

Part of the value of thinking about personal data on this continuum comes from thinking about how data at different points along the continuum relate to each other. Inferring the onset of clinical depression is one type of very high level inference that has been increasingly explored recently (Doryab, Min, Wiese, Zimmerman, & Hong, 2014; Saeb et al., 2015). This example offers a useful example for reasoning about the personal data continuum.

At the high end, of the spectrum the goal is to know whether or not an individual is depressed. Obviously this is very high-level and not directly observable, especially not by today's technology. However, it is indirectly observable (American Psychiatric Association, 2013). Criteria include:

- Patient has been less social
- Patient has been doing fewer things that he enjoys
- Patient hasn't been sleeping well
- Patient has been less physically active
- Patient has had a significant change in weight
- Patient has been experiencing high stress

Each of these represents a characteristic that is certainly further down on the continuum: these are closer to things that can be easily observed (though not necessarily instantaneously). Each of these items informs the top-level inference of depressed/not depressed.

Going another level lower, each of those items can be broken down into lower-level data that can generate output that answers those questions. For example, some data that might feed into how social a person has been could include data about how much time they have spent talking on the phone, how many text messages and emails they have exchanged, how many total people they have spoken on the phone with, what percentage of their time they've spent speaking with other people face-to-face. It would be easy to brainstorm many other kinds of data here. To infer that the patient has been experiencing high stress, a model might again employ features of the patient's communication behavior, stress indicators in their speech patterns, perhaps even the content of communication exchanges. The model might also take into account how full the patient's calendar is and how many of the events are routine.

Finally, at the lowest level, a variety of sensors provide data that inform the higher-level types of data. For example, accelerometer and gyroscope data can be used to infer what kinds of activities the individual has been engaging in. Those sensors, combined with the microphone and light sensor can be used to infer sleeping behavior (Min et al., 2014).

This single example of detecting depression in an individual has demonstrated aspects of personal data all along the continuum, from the very lowest levels of personal data as sensor logs through several layers of inferences up to a high-level inference of detecting depression.

4.3 The steps for working with personal data

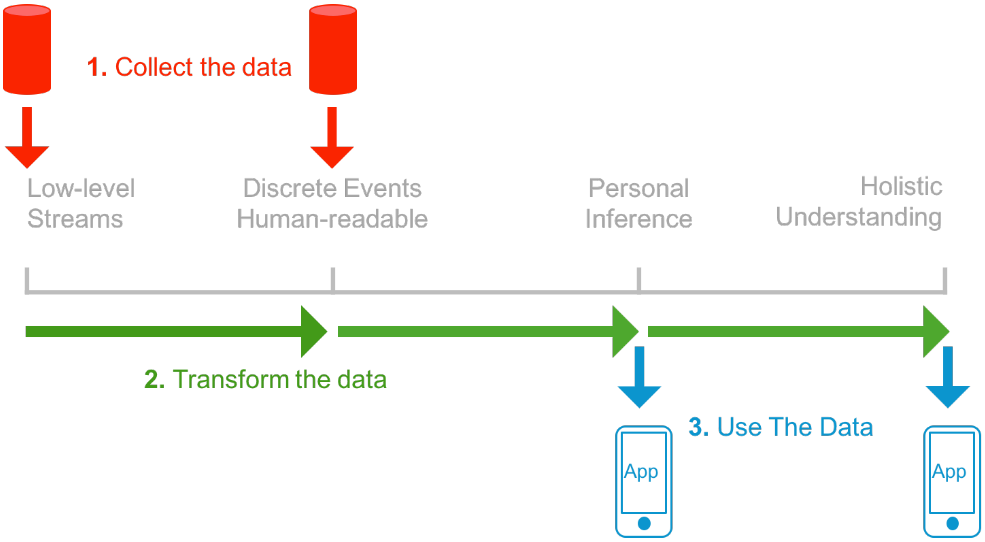


Figure 14: The personal data pipeline breaks down the steps of working with personal data. At a high level, using personal data means collecting the data, inferring some meaning from that data, and then applying the data to the target application. However, these steps are deceivingly simple. In reality each of these steps is complex with many components and a host of implicit challenges.

The illustration of inferring depression in the previous section, as well as the case study of inferring tie strength and sharing preferences in chapter 3 are both examples that offer some insights into the process of working with personal data with the ultimate goal of applying it to some target domain. This section expands on these to establish generalized steps that capture the process of applying personal data to an application, leveraging the continuum from the previous section. Together, the continuum and the steps described in this section combine to form the conceptual framework of personal data.

At a high level, the steps are:

1. Collect the personal data from one or more sources where the data has been recorded and stored.
2. Transform the collected data on the continuum from the point where it was when collected to the point where it needs to be in order to be applied to a particular target application.
3. Use the transformed personal data in the target application.

It is easy to look at that list and infer that this is a simple process: each step is short and concise. However, this process is actually much more complicated than what it would seem at first glance.

4.3.1 Collecting the Data

The first step of the process is to collect the source personal data. This data tends to be towards the lower parts of the continuum, but it might be anywhere along the continuum. This can include collecting data directly from sensors, usage logs, or from other systems that have already processed the data in some way. Collecting this data is really broken down into several steps.

1. **Choose services that allow programmatic access to user data:** The discussion of stakeholders highlighted the power that data-loggers have in this ecosystem: they are the gatekeepers to personal data, so if they don't collect the desired data or don't provide programmatic access to it, nothing else can be done.
2. **Authenticating the user:** Most personal data is protected through some authentication mechanism that the user must authenticate with in order to provide access to the developer.
3. **Obtaining permission**: The application or service that is collecting the data must obtain permission from the user to access the data. This can take many forms. On Android, this is done at install time. If connecting to a REST API (e.g. Fitbit, Email, etc.) then the user must authenticate and grant permission to access the desired data at runtime.
4. **Representing the data**: In almost all cases, different data sources represent their data in different schemas, even when the underlying type of data is similar.
5. **Linking the data together**: When combining data from multiple sources, making use of the data often means linking it together in some way. For example, when collecting different kinds of communication data, it is often necessary to connect the communication based on the person that the communication was with.
6. **Cleaning the data**: In some cases, data from one data source could be duplicated by the data from a different source. The complexity here can vary considerably. For example, Gmail offered the ability to archive Google Talk conversations within Gmail. So, collecting data from Gmail as well as Google Talk would result in a double-counting of those communications for the users who had enabled that archiving feature. In other cases, individual pieces of data or all data from a data source may be biased or incorrect in some way.

Whether a developer thinks about these steps consciously or just implicitly, each of these steps is essential for collecting personal data. Furthermore, the complexity increases considerably when collecting data from multiple sources, whether they are different data sources for the same type of data or for different types of data.

For many developers, the challenges present here severely limit what they do with personal data. They may support fewer sources or they may choose not to attempt an ambitious idea because of these limitations. Furthermore, decisions made at this stage, whether purposeful or implicit, will affect what can be done with the data later on and also the ease with which additional data sources can be added in the future.

4.3.2 Transforming the Data

The next step of the process is to transform the data from the point on the continuum where it was collected to the point on the continuum that it needs to be in order to apply it in the application. Again, there are multiple steps involved here.

1. **Deciding what the target data is:** There are many possible ways to abstract and transform the data, and there are tradeoffs between them. (e.g. Does the application need a representation of tie strength, or just communication frequency? Communication frequency is much more explicit and easier to obtain than tie strength, but for a particular application tie strength might be the right level of abstraction). Is the target numerical? Nominal?
2. **Deciding the transformation mechanism:** Is the transformation going to be machine learning-based? Rule-based? A mathematical transformation? Part of this step will depend on the resources available to the developer. Does the developer know how to apply machine learning? Does the developer have a way of collecting the ground truth data that will be necessary to train machine learning models?
3. **Assembling the input for the transformation:** This step involves preparing the source data. One aspect of this step is strongly related to how the data was collected: is it in a format where it is easy to prepare for input, or does it require additional processing? If the transformation involves machine learning, the developer needs to determine the feature set and calculate the features. The developer must also consider what will happen for radically different inputs (e.g. if there is no data available from a particular data source for a particular user, or if the data is too sparse or over too short of a period of time for a particular user).
4. **Collecting training data:** Having a good dataset is key to developing a good machine learning algorithm. In the case of personal data, it can be very difficult to assemble that data: it requires collecting personal data from many users, trying to broadly cover the spectrum of possible inputs in order to produce robust models.
5. **Collecting labeled ground truth:** Related to collecting the training data, the developer must have labels for that training data in order to construct models. However, unlike many other machine learning problems, the effort of providing these ground truth labels cannot necessarily be shifted to paid laborers (e.g. crowd workers). Instead with personal data, the user often must label their own data because they are the only person that knows what the label is For example, in the tie strength and sharing models of chapter 3, the only person that could possibly answer is the user.

The transformation step is again a complex step that will have implications for how data will be applied in the last step and also for how easy it will be to maintain the code and implement changes in the future.

4.3.3 Using the Data

With the data transformed, the final step of the process is to actually apply the newly transformed data to the application. Again, this seems like it should be straightforward, but again there is the potential for complexity.

1. **Integrating the data into the application:** Figuring out how to present the data to the user requires consideration. Will the user be able to understand the transformed data? Do they need to understand it? Will users feel that a particular transformation is sensitive or invasive in some way? If the transformation involved some uncertainty, how is that uncertainty handled by the application and/or represented to the user? Does the system simply display the data to the user, or does it personalize or automate some behavior based on the data?
2. **Handling incorrect inferences:** How does the application handle incorrect inferences? Does it allow the user to correct them? Are these changes stored? Are the changes used to retrain the model?
3. **Offering transparency and control to the user:** How is the resulting data used in the application? Does the user have the ability to know how their data is being used? Is the data being shared with third parties? Is there a data retention policy? Can the user change, hide, or remove data? Can the user change how the application behavior that is associated with the underlying data?

4.4 The Conceptual Framework

Together, the continuum of personal data and the steps that are required to incorporate personal data in an application, which have been described above, form the components of a conceptual framework (Figure 14). This framework captures and makes explicit the otherwise implicit realities of applying personal data to an application. The framework serves as a boundary object to support reflection and discourse on the process of working with personal data. One thing that this framework makes particularly salient is the amount of effort that is currently required for a developer to incorporate personal data into an application.

In some ways, this framework looks similar to a more general data analytics pipeline, however many of the specifics are notably distinct between data analytics in general, and personal data in specific. For example, the pipeline described by Fisher, DeLine, Czerwinski, and Drucker (2012) define five steps that describe the process of working with big data: acquire data, choose architecture, shape data into architecture, code/debug, and reflect. The first three of these steps map to the first step described here (collecting the data), code/debug corresponds to the second step of transforming the data, and reflect is one way of completing the third step of applying the data. However, where these steps correspond abstractly, the specifics of these two processes have important differences that make them distinct.

Acquiring a big data dataset involves identifying an existing dataset, (e.g. from an online repository). This is a static step, it happens once. When developing with personal data, the data that is being acquired is specific to each user. Thus, the data is not acquired all at once, it is instead acquired at application runtime for each individual user, and is typically continuously collected over time. Individual users can either grant or deny access to their data. Furthermore, collecting data oftentimes requires collecting hand-labeled ground truth from users, which is completely outside of the requirements for a big data pipeline. Other aspects of this step are more similar, such as linking together data from different data sources and working with different schemas.

Coding and debugging with big data is largely focused on issues of scale (e.g. writing parallelizable code and abstracting away the cloud). There are other tradeoffs in this step as well, such as the tradeoff between doing manual operations on the data versus scripting. When developing a personal data application, the challenges are very different. Developers need to consider what transformation is taking place on the data. They do not have the ability to access each user's data while developing the transformation. Instead, they need to prepare for contingencies ahead of time (e.g. how will the transformation behave with small amounts of data, large amounts of data, sparse data, dense data, etc.). Another important dimension of personal data is that the same behavior can mean different things for different users. Will the developer support user-specific transformations (e.g. personalized models)?

Finally, the step of reflecting with big data is a discrete step that is iterative with the step of coding and debugging in which analysts reflect on transformed big data in order to build insights in that data. By contrast, personal data is applied directly in an application that faces the end user, who is the subject of that data. Even in the case that the data is being used to support the user in reflecting on her own data, there are differences in this step (e.g. this data is personally meaningful, the user can identify errors and fill in holes in the data). Beyond reflection, personal data can be used in applications to enable new applications or support personalization, which goes beyond the structure of a traditional data analytics pipeline. The personal nature of this data has implications for the way that users relate to the data.

Overall, personal data has many differences from a traditional analytics data pipeline, and two main differences stick out in particular. First, with traditional data analytics it is possible to follow the process with a series of manual steps: the start-to-end pipeline does not need to be fully automated. Second, the individually meaningful nature of personal data has the potential to impact all steps of the personal data framework.

4.5 Design Goals

I have synthesized the insights and challenges throughout this document to establish a set of design goals aimed at improving the state of personal data. These goals are a combination of insights gathered from the broader landscape of personal data work within the research community from chapter 2, the case study of my own experience developing with personal data from chapter 3, and reflecting on the current state of personal data and the process of working with it from chapter 4.

4.5.1 Minimize redundant effort required of developers

This design goal is a very broad goal that should be further broken down to highlight the many different places that redundant effort is currently required:

- Authenticating to multiple APIs
- Working with non-standard data formats
- Gathering or collecting the data
- Cleaning the data
- Linking data together so that it is easy to query
- Developing and testing useful abstractions or inference models to transform the data so that it can be used in an application

These are all tasks that could be simplified or completely eliminated from the responsibility of an individual application developer. This will make the development process easier and significantly lower the bar for innovating in this space. End users will benefit from more services, better services, and less effort required fixing their own data. This design goal is directly inspired by chapter 3: the entire project would have been much more straightforward without all of the complexity involved in bringing the data together.

Developing inferences can be a major barrier for developers. Even if the developer is skilled at applying machine learning (e.g. extracting features, selecting an algorithm, tuning parameters) these tasks still require time and effort (Patel et al., 2010). Furthermore, collecting training data and ground truth labels can be an even more difficult challenge. Essentially, the to address this goal the process of developing a reliable model should be made separate from the process of deploying that model in a target application.

One component of addressing this design goal is to have a set of inferences that are general enough that they can be used across multiple applications. For example, there are a number of ways that tie strength could be applied in different ways across a variety of applications.

4.5.2 Organize data by individual, not by service

Today, data is siloed in each application or service. This makes it easy to build applications that are based on their own data, but makes it much more difficult to offer an integrated and consistent user experience across multiple applications. This process should be quick and easy for developers.

It is easy to see why personal data today is organized by application or service: there is value for a service in having this data across all of its users, and it is the simplest thing to do. Even if a service wanted to make it easy for users to store their data on the level of the individual, there is no infrastructure for this today. Where would this data be stored? Who would protect the data? Who would pay the costs associated with storing this data? This concept has been proposed before in (Want et al., 2002), but the infrastructure for this is simply not there today. This is a major challenge across many of the research domains described in chapter 2, and the research projects in chapter 3 offer specific insights into how much of a challenge and a frustration this siloed approach is for developers and researchers. This also addresses the need cited in (Karger & Jones, 2006), that personal information must be defragmented (i.e. unified and linked together) in order for individuals to be able to realize the full potential of their data.

4.5.3 Support connections within the data

It should be easy to access data that is related to a particular piece of data. It should be easy to jump between related pieces of data. For example, a piece of information such as the most recent phone can have connections to other items that are overlapping in time such as a calendar appointment, or where the user was when she made the call. It can also be related to whom the call was with. There are many cases where these rich interconnections within the data are particularly useful. In the case of the tie strength model in chapter, rich interconnections in the data would simplify the process of calculating the features for the model. Having well connected data dramatically simplifies applications is also useful for episodic memory queries (e.g. "who did I call last time I was in San Francisco?"), personal-informatics-style data exploration (e.g. "do I spend more time on the phone when I'm at home or travelling"), specifying complex rules (e.g. in end-user programming environments), and for applications such as Autobiographical Authentication (Das, Hayashi, & Hong, 2013).

4.5.4 Limit unnecessary disclosure

One way of minimizing opportunities for personal data to be exploited is to follow the privacy maxim of limiting unnecessary disclosure (Romanosky, Acquisti, Hong, Cranor, & Friedman, 2006). In the ideal world, the only data that an application would access would be the data that it needed to access. For example, if an application only needs to know how many phone calls the user has made over a certain period of time, the application should definitely not have access to the individual phone call logs, only access to the count of logs over a specified time period. If the system can guarantee that only a specific set of data was accessed, it will be better able to support the user in controlling and limiting data access.

4.5.5 Offer users transparency

Offer as much transparency as possible when it comes to what personal data is used, specifically how that data is used, what (if anything) is stored, and what is shared or transmitted. One way of accomplishing this is by giving examples that demonstrate what can be done. Today, many privacy policies include language that is so general that it hardly communicates anything at all. Part of this design goal is to be as specific and clear as possible.

4.5.6 Offer users choices and control, while specifying reasonable defaults

Hand-in-hand with transparency, systems that handle personal data should also offer users choices and control.

In many cases, personal data is a component of the economics that enable a particular service to operate (often through revenue generated from behavioral advertising). In these cases, a user's privacy decision might affect the viability of that service. In these cases, a combination of transparency (e.g. letting the user know that they are able to offer free or subsidized service because of access to this data) and choices (e.g. the user could pay for the service instead of providing their data) offers more flexibility to the concerned consumer.

There is also room to innovate directly in the space of privacy and sharing mechanisms that are provided. Chapter 2 offers several examples of this in the form of expressing preferences that depend on "in common" information (Wiese, Kelley, et al., 2011), or easier ways of partitioning the target audience (Sleeper et al., 2013).

A broad and growing literature makes it clear that designing privacy and sharing controls is incredibly difficult, and in many cases people do not even understand what privacy settings mean (Kelley et al., 2012; Liu, Gummadi, Krishnamurthy, & Mislove, 2011). This is far from a solved problem. Thus, it is not enough to offer control, services should also specify reasonable defaults.

Finally, another component of this design goal is to offer users the ability to improve the service they are receiving (e.g. by providing ground truth or resolving duplicates).

5 Phenom: A Service for Unified Personal Data

Approaching the ecosystem of personal data from a user-centered perspective represents a significant shift from how personal data is handled today. Chapter 4 offered insights into how personal data is handled today, some of the issues with the current state of personal data, and design goals for improving this state. Achieving these goals is a long-term agenda that will require buy-in from many different stakeholders. There are many opportunities for improving the ecosystem of personal data. There are also so many prospects to employ personal data to improve the way that users interact with their technology.

This chapter describes the design and implementation of Phenom, a prototype service for managing personal data. Phenom incorporates several key ideas that represent a significant advance in the way that personal data is handled. While many issues remain to be addressed before the vision set forth in chapter 4 is achieved, Phenom is a proof of concept that represents an important step towards this goal.

The name Phenom comes from phenomenology, a field of study which "set out to explore how people experience the world – how we progress from sense-impressions of the world to understandings and meanings" (Dourish, 2001).

5.1 System Architecture

The main philosophy behind Phenom is that personal data is managed centrally in a single application-agnostic service, rather than in each independent application or service. This approach offers several key benefits:

1. A single API for an application developer to work with. The developer only has to authenticate the user to a single API, and the developer only has to work with a single format for representing the data.
2. Linking data together, correcting bad data or mistakes, and removing duplicates can all be done only once and the results will be reflected everywhere.
3. Inferences and models can be developed and improved centrally and the benefits can be had by all applications.
4. Operations on personal data can happen within the service, making it easier to constrain what data a client application has access to.
5. A user can specify privacy preferences in a centralized service with a familiar user interface, rather than in each client application.

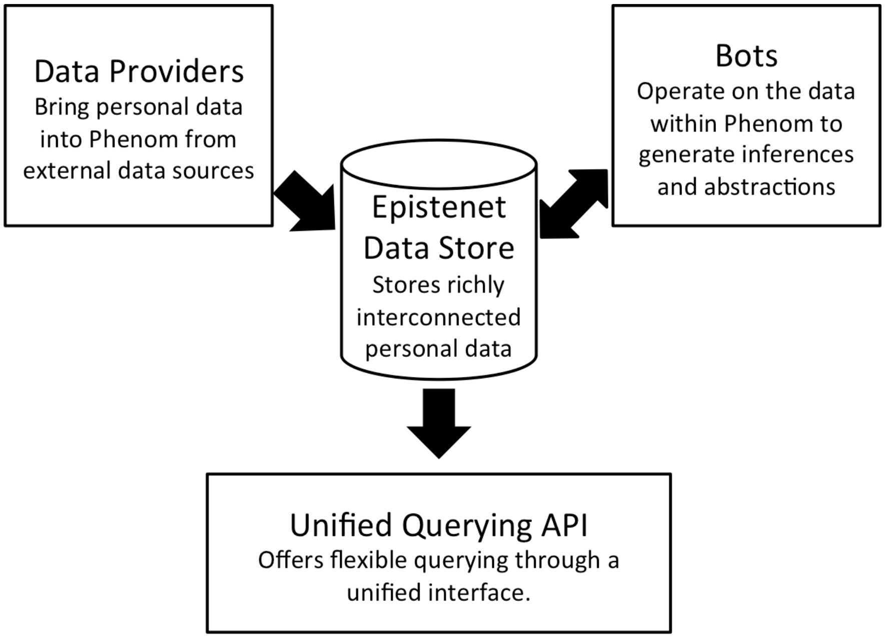

Data Providers
Bring personal data into Phenom from external data sources
Bots
Operate on the data within Phenom to generate inferences and abstractions
Epistenet Data Store
Stores richly interconnected personal data
Unified Querying API
Offers flexible querying through a unified interface.

Figure 15: A system diagram for Phenom illustrating its different components. The Epistenet Data Store serves as a semantic knowledge base of personal data. Data providers bring personal data in from external data sources. Bots operate on the data contained within the datastore to generate inferences and abstractions. A unified querying API provides application developers with a single query interface to access the richly interconnected personal data from the datastore.

At a high level, the Phenom architecture is composed of several key components:

- **Data providers** are responsible for connecting to a data source, retrieving new data from that source, and storing it in the internal datastore.
- The **data store** contains the rich interconnected data that has been brought in from the providers. The datastore contains objects of many different types, with attributes that can include references to other objects. Finally, object types are defined in a semantic tree, where the children of a type contain all of the attributes of its parent (but may contain additional attributes as well). The data store also contains inference data and ground truth data.

- **Bots** perform operations on the data in the semantic data store, for example simple "housekeeping" operations, model-based inferences, heuristic-based inferences, etc.
- The **API** offers the flexibility of SQL-like queries to the personal data store that is particularly focused on the needs associated with querying personal data, such as connecting across multiple data types and working with timestamps and aggregates. The Java API offers a simplified abstraction on the much more complicated structure of the underlying datastore while preserving query flexibility.

The remainder of this section offers a more in-depth discussion of these components that together form Phenom.

5.1.1 Epistenet: A Semantic Data Store

Epistenet, the semantic data store component is a core component to Phenom [XXX tech report]. Epistenet is a system that I co-developed with Sauvik Das, who lead the development of this component.

One major challenge when developing with personal data is that data from different sources are completely separate. For example different data sources use different schemas.

To address this issue, Epistenet offers a unified internal format for storing personal data with a source-agnostic schema, support for connections between different objects, and a hierarchical ontology that specifies subsumption relationships between different data types. To enable this source-agnostic schema and rich interconnections between data, every piece of personal data in Epistenet is represented as an object with some number of attributes that are associated with that object.

For example, a `PhoneCall` is one type of personal data captured by Phenom. Epistenet represents the `PhoneCall` object with the following attributes: `Direction` (incoming or outgoing), `Duration`, and `AlterAddress` (phone number). The underlying data schema of this implementation is very flexible. Data is stored in an SQLite database where `EpistenetObjects` are stored in one table and all `Attributes` are stored in a table that contains the name of the Attribute, a reference to the `EpistenetObject` that it is associated with, and the value. This simple database schema offers a flexible framework for objects to be represented within Epistenet.

Another issue for working with personal data is that data from different sources may be interconnected with each other in many ways, but because they are coming from different sources, those interconnections are difficult to leverage. Sometimes these interconnections are across semantically disjoint data (e.g. a user's presence at a physical location, and a cell phone call might be connected through their timestamp. The types of data are completely different). In other cases types of data have a more direct semantic relationship (e.g. phonecalls and SMS messages are both types of communication).

Epistenet also has infrastructure to support these kinds of connections. Each `EpistenetObject` is associated with an `OntologyClass` that identifies its type. `PhoneCall` is one example of an `OntologyClass`, so every `EpistenetObject` that represents a log of a phone call is associated with the `PhoneCall` ontology class. Ontology classes are key to a very important feature of Epistenet: the ability to maintain connections between data that is semantically related. Continuing the phone call example, a related type of data is a text message. Text messages and phone calls are both types of communication, and because of this similarity they share some attributes in common, such as `Direction` and `AlterAddress`, but not others like `Duration`. To capture this semantic relationship, Epistenet has a hierarchical representation of ontology classes. In the example, both the `PhoneCall` and `SMSMessage` ontology classes are children of the `Communication` ontology class (see Figure 16). The `Communication` ontology class defines the common attributes of `Direction` and `AlterAddress` and the `PhoneCall` and `SMSMessage` ontology classes inherit those attributes, and can define their own additional attributes as well.

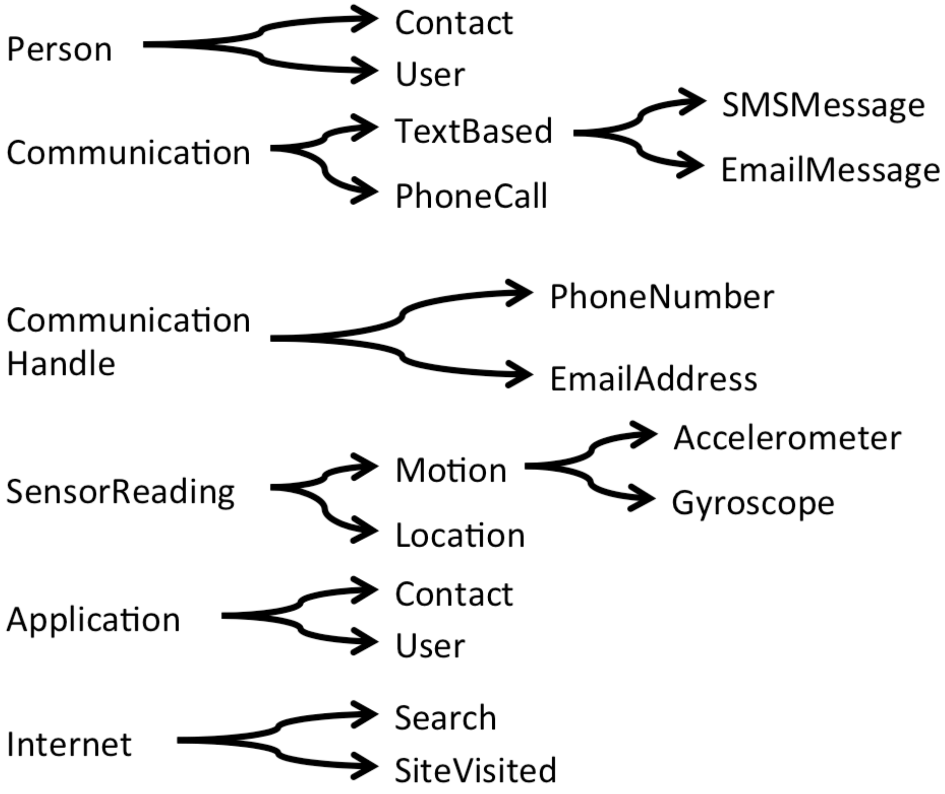


Figure 16: An example of an ontology in Epistenet. Direction edges in this graph refer to “subsumptive” relationships. So, a PhoneCall is a type of Communication. Attributes of a parent ontology class are also contained in the descendants of that ontology class.

As a result, a query to Epistenet for objects of a particular `OntologyClass` can specify whether the objects that are returned should only be the objects that are concretely associated with that `OntologyClass` (e.g. only objects that are explicitly identified as `Communication`, not `PhoneCall` and `SMSMessage` which are children of the `Communication` ontology class), or if it should also include objects that have a concrete type of a child `OntologyClass` (e.g. a query for the `Communication` ontology class would also return objects that were defined as both `PhoneCalls` or `SMSMessages`). Epistenet refers to these relationships as *identity* (i.e. only objects that are the concrete type specified in the query) and *subsumption* (i.e. all objects in the ontology subtree of the specified type).

This semantic linkage is powerful. Inserting a `PhoneCall` object into the datastore means that automatically through the ontology relationship it is also represented as a `Communication` object, and tied the object to a `CommunicationHandle` (through the `AlterAddress`), which is linked to a `Contact` ontology class object, which is subsumed by the `Person` ontology class. This structure allows for very rich flexible queries, enabling a single query to automatically incorporate the data from different but semantically related data types. For example, it would be very simple to query for a list of the 10 contacts that a user had communicated with most recently across all communication media (Section 5.1.3 describes the specifics of executing queries such as these using the API).

Defining a new Ontology Class

The following steps document the process of defining a new ontology class within Epistenet:

1. Determine where in the Ontology the new class should be added. For example, if we are adding a provider for phone call logs, the place to put the `PhoneCall` Ontology Class would be `Communication` → `PhoneCall`. An Ontology Class could also be at the root.
2. Declare the new ontology class in the `ontology.config` file. This includes specifying the name, and a versioning number (here it's 2). This file also specifies the ontology:

```
  SMSMessage,1
+ Phonecall,2
  …


  …
  Communication,Textbased,1
  Textbased,Email,1
+ Communication,Phonecall,2
```

3. Declare a new class that includes the attributes that will be associated with the `Phonecall` ontology class in in the `ontologyclasses` namespace:

```
public class Phonecall extends Communication {
   public static final Attribute DURATION =
      new Attribute("Duration", AttributeValueType.INTEGER);

   public static Attribute[] getAttributes() {
      return ArrayUtils.addAll(Communication.getAttributes(),
         new Attribute[]{Duration});
   }
}
```

4. Declare Gmail in the OntologyClass.java enum. The number just needs to be unique within the enum:

```
            Phonecall("Phonecall", 6) {
                @Override
                public Attribute[] attributes() {
                    return Phonecall.getAttributes();
            }
        },
```

After completing these steps, a new `Phonecall` ontology class will exist in Epistenet with all of the attributes associated with the `Communication` ontology class, in addition to a "Duration" attribute.

Reference Attributes

The example above offers a view into the general process for defining new `OntologyClass` types, but there are a few more details that are involved when defining some kinds of attributes. Many attributes are similar to "Duration" in the example above: they represent a basic value such as Integer, Double, Timestamp, or String. However, some attributes actually reference an Epistenet object. These are `ReferenceAttributes`. In the phone call example, one example of a `ReferenceAttribute` is the phone number. The reason that this attribute did not appear in the description above is that it was inherited from the `Communication` ontology class.

In fact, this is even more complicated because at the level of the `Communication` ontology class, the identifier isn't necessarily a phone number. It could also be an email address, or a screen name. Thus, the definition for `Communication` includes this:

```
    public static final ReferencesAttribute ALTER_ADDRESS =
            new ReferencesAttribute("AlterAddress",
                    OntologyClass.CommunicationHandle);
```

This "AlterAddress" `ReferencesAttribute` in `Communication` references a `CommunicationHandle` ontology class which has ontology class children `PhoneNumber` and `EmailAddress`. This preserves the overall structure of the `Communication` ontology class (i.e. communication happens with other people), but also maintains differences between different kinds of handles (i.e. phone numbers and email addresses).

The `Communication` ontology class also includes a reference to the `Person` ontology class. However, in this case it is not possible to use a `ReferenceAttribute`, because the communication is only indirectly tied to a person through the communication handle. Instead, the `IndirectReferencesAttribute` is a sort of ghost reference that references the target ontology class (i.e. `Person`), through the attribute of an intermediate ontology class to which this ontology class has a reference (i.e. the `Person` attribute of the `AlterAddress`, which is a direct reference from `Communication`).

```
    public static final IndirectReferencesAttribute PERSON =
        new IndirectReferencesAttribute(
            "Person",
            Communication.ALTER_ADDRESS,
        CommunicationHandle.PERSON);
```

One issue with ReferencesAttribute and IndirectReferencesAttribute is that they are unidirectional. In this case, the PhoneCall ontology class has a ReferencesAttribute for AlterAddress and Person, but Person does not have a ReferencesAttribute to PhoneCall. ReverseReferencesAttribute solves this problem. ReverseReferencesAttribute uses the existing attribute connection to make the relationship bi-directional. For example, the following two attributes are defined in the Person ontology class:

```
public static final ReverseReferencesAttribute PHONE_CALLS =
   new ReverseReferencesAttribute(
      "PhoneCalls", OntologyClass.Phonecall, Phonecall.PERSON);
```

This attribute is a sort of convenience attribute. No additional data is stored in Epistenet, but when querying the Phenom API, ReverseReferencesAttribute behaves the same as ReferencesAttribute by reversing the direction of the original ReferencesAttribute.

Together, the structure of Epistenet makes it very easy to interact with the data. For example, perhaps when a phonecall is initially recorded, Phenom does not know to whom the phone number belongs. With this approach, when the connection is made between the phone number and the person, all of the data is instantly updated for free.

5.1.2 Data Providers

Today, working with personal data means any application that wants to use the data from a data source needs to individually connect to all of the datasources to access that data. This process it often painful and involves a fair amount of boilerplate code. Phenom removes this responsibility for each developer by doing this only once.

Data providers are the component of Phenom that brings in raw personal data from any external source—for example, system content providers such as SMS logs, hardware sensors such as the accelerometer, and third-party applications such as “What’s App”. The call log, for example, is a data provider that contributes objects of the “PhoneCall” ontology class, while a web browser would contribute objects of the “SiteVisit” ontology class.

Data providers are responsible for connecting to the external data source, mapping the data from that data source to an ontology class within Phenom, and avoiding creating duplicate data. Data providers are polled at configurable intervals so that the can aggregate new data. While providers themselves do not offer much novelty to Phenom, they are an essential component for Phenom.

Defining a New Provider

Creating a data provider involves only a small amount of overhead beyond the boilerplate code that is required to query the data source and extract data from that source.

1. Add a line to providers.config with a name for the provider the number of milliseconds between polling times, and the Java classname for the provider:

```
  …
  sms_logs,14400000,SMSLogsProvider
+ call_logs,14400000,CallLogsProvider
  …
```

2. Implement the `CallLogsProvider` class in the `providers` namespace. The key aspect of implementing the provider is implementing the `poll` method.

```
public class CallLogsProvider extends Provider {
   public static final long PERSISTENCE = Long.MAX_VALUE;
   public static final String PERMISSION =
      "android.permission.READ_CALL_LOG";
   public static final String PROVIDER_NAME = "call_logs";

   public void poll() {
…
```

Within this method, the main steps are to

a. Poll the data source for new data:

```
EpistenetAdapter adapter = this.getAdapter();
long lastUpdated =
   adapter.getLastUpdatedTimeForProvider(
      this.getProviderName());
Cursor c = this.mContext.getContentResolver().query(
   CallLog.Calls.CONTENT_URI,
      new String[] { Calls.DATE, Calls.NUMBER,
         Calls.DURATION, Calls.TYPE, Calls.CACHED_NAME},
            Calls.DATE + " > " + String.valueOf(lastUpdated),
   null, null);
```

b. Cycle through the new data to create new objects:

```
if (c != null && c.getCount() > 0) {
   while c.moveToNext()) {
      long oid = adapter.createObject(PERSISTENCE);
```

c. Associate all of the relevant attributes with each object:

```
adapter.createAttribute( Phonecall.DURATION,
   c.getString(c.getColumnIndex(Calls.DURATION)),oid);
adapter.createAttribute( Phonecall.DIRECTION,
   this.getStringifiedType(
      c.getInt(c.getColumnIndex(Calls.TYPE))), oid);

adapter.createAttribute( Phonecall.ALTER_NAME,
    UtilityFuncs.coalesceString(c.getString(
       c.getColumnIndex(Calls.CACHED_NAME)),"Unknown"),oid
    );

// Code to create phone number object if necessary
long[] numberCreated =
   adapter.createObjectIfDoesNotExist(
      new String[]{ PhoneNumber.HANDLE.getSelectName()},
      new String[]{UtilityFuncs.formatPhoneNumber(
         c.getString(c.getColumnIndex(Calls.NUMBER)))}, -1);

if(numberCreated[0] > 0){
   adapter.createAttribute(PhoneNumber.HANDLE,
      UtilityFuncs.formatPhoneNumber(
         c.getString(c.getColumnIndex(Calls.NUMBER))),
         numberCreated[1]);

   adapter.createObjectOntologyLink(numberCreated[1],
      adapter.getIDsForOntologyClassNames(
         OntologyClass.PhoneNumber.className()).get(
            OntologyClass.PhoneNumber.className()));
}

adapter.createAttribute( Phonecall.ALTER_ADDRESS,
   String.valueOf(numberCreated[1]), oid);
```

d. Finally, add in the meta attribute, link the object to the rest of the ontology, and release resources:

```
long metaAttribute = this.createMetaAttribute(oid,
   c.getLong(c.getColumnIndex(Calls.DATE)));

this.createObjectOntologyLinks(oid);
}}
this.closeCursor(c);
}
```

Omitting a few accessors and utility methods, this is all that's required to create the provider for phone logs. The flexibility of this approach allows for much more complex providers to be implemented if necessary.

5.1.3 API for Querying Unified Personal Data

For Phenom to be effective, it needs to enable developers to access the rich interconnected personal data that is contained within in a simple and flexible manner. This requires designing and deploying an API that will be support the ontological structure and flexible attribute schema supported by the rest of Phenom, and providing a single unified environment for accessing the results of an arbitrary ontology class.

The Phenom API provides a unified interface for accessing the personal data that is stored within Phenom. The Phenom API is accessible to client applications through a lightweight Android Library Project. Client applications can use the library to query the API. The library binds to the Phenom service, which runs in a separate process on the phone. Results are then returned to the client application through a callback mechanism.

Specifying a query

Specifying a query to Phenom requires defining one or more `Filter` objects. A Filter can be very simple. For example, the following Filter specifies that the duration field should be returned for all phone calls:

```
Filter phonecallFilter =
   new Filter(OntologyClass.Phonecall).projection(Phonecall.DURATION);
```

While filters can also be more complex than this, the basic idea is the same. To create a `Filter`, the `OntologyClass` that specifies the type the `Filter` should return. After this, the Filter object behaves like a builder. Options for the filter include:

- Simple Constraints on the `OntologyClass`'s attributes (e.g. lessThan, greaterThan, equal, notEqual, inSet, notInSet, inRange)
- Limit the number of results returned

- Specify the sort order based on an attribute
- Specify an SQL-style "group by" on an Attribute
- Specify the projection of attributes to be returned. Attributes not explicitly included here will not be
- Constrain through join: essentially allowing for a compound query to be specified based on a `ReferencesAttribute` that connects this `OntologyClass` with a different `OntologyClass`.

In all of these cases, when an Attribute is required as a parameter, any attribute can be used. In particular, Phenom provides support for three additional non-concrete attribute types not previously discussed: `AggregateAttribute`, `TimepartAttribute`, and `ReferencesAggregateAttribute`.

`AggregateAttributes` perform the same behavior as SQL aggregates (i.e. sum, average, count, min, max, and group concat). An AggregateAttribute must be based on a concrete attribute, and can be obtained by calling the `asAggregate()` function of `Attribute`. For example:

```
Phonecall.DURATION.asAggregate(AggregateType.SUM);
```

The resulting `AggregateAttribute` can be used in any place where a concrete attribute would normally be used. It is important to note that the "group by" for a filter should be specified, otherwise the `ID` attribute is used as the attribute by default.

`TimepartAttributes` make it easy to extract information from a timestamp and use it within the query. For example this is the query for extracting the year number and month number from a timestamp and returning it in a single attribute:

```
PlaceVisit.TIMESTAMP.asTimePart(TimePart.YEAR, TimePart.MONTH)
```

It is also possible to get an `AggregateAttribute` of a `TimepartAttribute`, which allows for easy querying of aggregated statistics based on time. For example, the following attribute would give the most recent month and year of a `PlaceVisit`:

```
PlaceVisit.TIMESTAMP.asTimePart(TimePart.YEAR, TimePart.MONTH)
   .asAggregate(AggregateType.MAX));
```

Finally, ReferencesAggregateAttribute makes it easy to get aggregate information about the attributes of an object referenced by a ReferencesAttribute. For example

Together, this simple query interface easily enables a set of rich queries to be made to Phenom. For example, the following query returns statistics about the 10 places where the user has spent the largest amount of time, including:

- Latitude and longitude
- the total amount of time spent there
- the average length of a stay
- and the most recent year and month that the user was there

```
Filter placeVisitsFilter = new Filter(OntologyClass.PlaceVisit)
   .projection(
      PlaceVisit.DURATION.asAggregate(AggregateType.SUM),
      PlaceVisit.DURATION.asAggregate(AggregateType.AVERAGE),
      PlaceVisit.TIMESTAMP.asTimePart(TimePart.YEAR, TimePart.MONTH)
         .asAggregate(AggregateType.MAX))
  .orderBy(PlaceVisit.DURATION.asAggregate(AggregateType.SUM), false);

Filter placesFilter = new Filter(OntologyClass.Place)
   .projection(Place.LATITUDE, Place.LONGITUDE)
   .constrainThroughJoin(placeVisitsFilter, Place.VISITS)
   .limit(10);
```

This is a prime example of a query that would be much more difficult, or even impossible, to execute without Phenom.

Handling a Query Result

Queries to Phenom are executed asynchronously, so handling a result from Phenom involves implementing a callback. It is easy to implement this callback as an anonymous function similar to the way that click handlers are often implemented:

```
mApiClient.sendQuery(placesFilter, new PhenomCallback() {
   @Override
   public void onSuccess(ArrayList<PhenomObject> objs) {
      processResults(objs);
});
```

Queries are returned as a list of `PhenomObjects`, which offers a basic structure for representing query results. Essentially, each `PhenomObject` represents an object of the `OntologyClass` specified in the query. Attributes of the ontology class that were specified in the `Filter`'s projection can be accessed by calling the `get` method for the corresponding type of the attribute.

5.1.4 Bots

The raw personal data gathered and stored within Phenom is useful in its own right, but often the real value of this aggregated personal data comes from additional processing or inferences that are done on top of the raw data. In some cases it makes sense for individual developers to do this additional processing, but in many cased multiple developers can make use of the same processing work. For example, as discussed in chapter 3, tie strength can be useful to a variety of applications.

Bots are the component that offers this functionality to Phenom. Bots carry out worker functionality on the Epistenet semantic data store, following a blackboard architecture where the Epistenet datastore is the blackboard. Bots are somewhat similar to Providers in that they are polled on a fixed schedule and they do work on the contents of the semantic data store. However, instead of inserting new data into the datastore, bots operate on the existing data. This can include maintenance tasks such as removing duplicated data or identifying connections across multiple kinds of data.

However, the more exiting use of Bots is to offer the ability to generate inferences and abstractions based on the existing data within Epistenet. For example, the "home_labeler" bot uses some basic heuristics to label places that the user calls home or has called home in the past, and the "strong_tie" bot uses communication behavior to infer some of a user's strong ties and label those contacts as strong ties in Epistenet. In these examples, there is an `Attribute` associated with the `Place` and `Contact` ontology classes respectively for each of these inferences. When a bot has made an inference, it simply adds or updates the corresponding attribute.

Defining a new Bot

There are only a few steps required to create a new Bot.

1. Add a line to bots.config with a name for the bot, the number of milliseconds between polling times, the Java classname for the bot, and the version number of the config file for which this bot was added:

```
  …
  significant_places,86400000,SignificantPlaceBot,1
+ tie_strength_bot,86400000,TieStrengthBot,2
```

2. Implement the `TieStrengthBot` class in the `bots` namespace. The key aspect of implementing the bot is implementing the `poll` method.

```
public class TieStrengthBot extends Bot {
    private static final String BOT_NAME = "tie_strength_bot";

    public TieStrengthBot(Context c){ super(c); }

    @Override
    public String getBotName() { return BOT_NAME; }

    @Override
    public String getPermission() {
        return "phenom.permissions.tie_strength";
    }

    @Override
    public void poll() {
…
```

Implementing this method depends on the specific functionality of the bot. In the case of the tie strength bot, the steps are to:

a. Query Epistenet for the relevant data:

```
// A few aliases for readability
ReferencesAggregateAttribute smsCount =
   Person.SMS_MESSAGES.getReferencesAggregate(
      SMSMessage.ID.asAggregate(AggregateType.COUNT));

ReferencesAggregateAttribute callCount =
   Person.PHONE_CALLS.getReferencesAggregate(
      Phonecall.ID.asAggregate(AggregateType.COUNT));

ReferencesAggregateAttribute callDuration =
   Person.PHONE_CALLS.getReferencesAggregate(
      Phonecall.DURATION.asAggregate(AggregateType.SUM));

Filter personList = new Filter(OntologyClass.Person)
   .projection(Person.ID, Person.NAME,
      smsCount, callCount, callDuration);

ArrayList<PhenomObject> allPeople =
   getAdapter().doPhenomQuery(personList);
```

b. Calculate tie strength based on the specified heuristics:

```
int maxDur = 0;
int maxCallCt = 0;
int maxSMSCt = 0;

for (PhenomObject person : allPeople) {
   maxSMSCt = Math.max(maxSMSCt, person.getInt(smsCount,0));
   maxCallCt = Math.max(maxCallCt, person.getInt(callCount,0));
   maxDur = Math.max(maxDur, person.getInt(callDuration,0));
}

for (PhenomObject person : allPeople) {
   double closeness = ((person.getInt(callDuration,0) / maxDur) +
      (person.getInt(callCount,0) / maxCallCt) +
      (person.getInt(smsCount,0) / maxSMSCt))/3;
   getAdapter().createOrUpdateAttribute(Person.TIE_STRENGTH,
      Double.toString(closeness), person.getLong(Person.ID));
}
```

In this case, creating a bot was as simple as that, there are no other steps. Of course, bots can be much more complex, for example actively retraining a model based on new data labels from the user.

5.2 Evaluation: Example Applications and Queries

This section offers two examples of applications that Phenom makes particularly simple, where previously they would have been much more complicated, perhaps impossible. These examples offer a basic evaluation that demonstrates the value offered by Phenom.

5.2.1 Bootstrapping Users' Interests from Location Data

The first example is an approach that is intended to solve the "cold start" problem that happens when a user begins using a new service that is trying to personalize content within the application (e.g. a personalized news reading application). One innovative approach to solving this problem is to try to use the user's location history to identify significant places that the user has been to. Specifically, a location history can be used to identify unique places that a user has visited, how recently, and how frequently the user has been there. With this information, it is possible to look up additional information about a location, like what type of a location it is, and any identifying characteristics of the location (e.g. a user that frequents a rock gym is likely interested in climbing). This data can then be used to generate an interest profile for a user. Even if the results are only partially correct, this approach is still better than completely random data, or no data at all.

Phenom Implementation

One of the bots implemented in Phenom is a SignificantPlaces bot, which uses a few different heuristics to identify places that are significant to the user based on her location history.

With the significant places bot implemented, this particular application is fairly straightforward in Phenom:

```
mApiClient = new ApiClient(this);
Filter placesFilter = new Filter(OntologyClass.Place)
   .projection(Place.LATITUDE, Place.LONGITUDE)
   .notEqual(Place.SIGNIFICANT_PLACE, "NULL");

mApiClient.sendQuery(placesFilter, new PhenomCallback() {
   @Override
   public void onSuccess(ArrayList<PhenomObject> objs) {
      getTagsFromFlickr(objs);
   }
});
```

Upon receiving the callback from Phenom, the application can cycle through the significant locations and query a third-party API for tags that are associated with those locations. In this example, The application connects to Flickr to retrieve the annotated photo tags from geotagged photos, but an implementation might also use data from Foursquare, Yelp, or Google Maps. Next, the application does TF-IDF with the words from the tags, and the result is a word vector that offers some clues to the user's interests that should be better than a random selection of articles.

This is a great example of the value offered by Phenom: there were very few steps involved in getting the data needed in a usable format. Phenom obviates the need to create the custom code to gather and store location data. Furthermore it makes it easier to retrieve location data based on different qualities of the data points (e.g. a window of time, a particular city, etc.). This offers a similar kind of abstraction to that which happens within modern GUI toolkits. These toolkits offer developers a lot of support for developing the graphical interface portions of an application. While each developer still needs to write the business logic and functionality of an application, they do not need to be concerned with the specific implementation of the GUI components (e.g. standard appearance of widgets, event stream, etc.). Similarly, Phenom obviates the boilerplate code that each developer would need to write in order to raise the abstraction level to a point where developers can focus on the business logic and functionality that is specific to their application.

Non-Phenom Implementation

Without Phenom, the first step to implementing this example is to access enough of a user's location history that it would be possible to identify the user's significant places. Possibilities include:

1. Asking the user to upload their location history (e.g. from Google Location History, which provides users with that data but does not offer an API)

2. Collecting the data from the API of a service that the user already uses (e.g. from Moves, or from Foursquare)

3. Collecting the user's location automatically within the application over a period of time until enough data has been collected that significant places are salient

Each of these options has drawbacks. Options 1 and 2 are service-dependent in a way that excludes users who do not use those services. Option 3 includes all users, but involves running on the user's device for long enough to bootstrap with enough data that significant places could be determined. Developers who are using Phenom are not exposed to this challenge because the data has already been brought together through the use of data providers, which can cover all three of these alternatives in a way that supports reuse across applications. However, without Phenom a developer has to cobble together a solution that is likely to exclude more users from the feature.

In this case, the goal was to eliminate the cold-start problem, so option 3 does not work. From a development perspective option 2 seems easier than option 1, though this does have the drawback of only collecting location check-ins, rather than all location data. First is pseudocode for accessing a user's check-ins:

```
Intent intent =
   FoursquareOAuth.getConnectIntent(context, CLIENT_ID);

startActivityForResult(intent, REQUEST_CODE_FSQ_CONNECT);
…
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent
data) {
   switch (requestCode) {
      case REQUEST_CODE_FSQ_CONNECT:
         AuthCodeResponse codeResponse =
          FoursquareOAuth.getAuthCodeFromResult(resultCode, data);
            Intent intent =
               FSOauth.getTokenExchangeIntent(context, CLIENT_ID,
                       CLIENT_SECRET, authCode);

               startActivityForResult(intent,
                       REQUEST_CODE_FSQ_TOKEN_EXCHANGE);
            break;
      case REQUEST_CODE_FSQ_TOKEN_EXCHANGE:
         AccessTokenResponse tokenResponse =
         FSOauth.getTokenFromResult(resultCode, data);
           checkins = retrieveCheckinData(resultCode.getAccessToken());
           break;
    }
}

…
private Checkin[] retrieveCheckinData(String accessToken){
   FoursquareApi api = new FoursquareApi(
      "ClientID", "ClientSecret", "CallbackURL",
      accessToken, new IOHandler());

   Result<CheckinGroup> result =
      api.usersCheckins(null, 1000, 0, Long.MIN_VALUE, Long.MAX_VALUE);

   return result.getResult().getItems();
}
```

At this point, we have access to the user's check-ins. The next step is to process the check-ins in a way that surfaces "significant places". Where developers that are using Phenom can make use of the existing "significant places" bot, here the developer would need to determine those significant places independently. For simplicity here, significant places can be the user's most frequent places. We might also want other information to be included here, such as the places where the user has spent the longest duration. However because we are using check-ins this information is not available. Pseudocode for this follows:

```
HashMap<Location, Integer> visitCount = new HashMap();
for(Checkin c : checkins){
   Integer count = frequency.get(c.getLocation());
   if(count == null)
      count = 0;

   frequency.put(c.getLocation, ++count);
}
visitCount.sortByValue(); //Implemented elsewhere
getTagsFromFlickr(visitCount);
```

Comparing Implementations

Even with this relatively basic task, these two implementations demonstrate several ways that Phenom offers value compared to the non-Phenom implementation. The most obvious difference between these two examples is the number of lines of code written for each example: notably fewer lines of code for Phenom. This is possible because the code for gathering and storing locations, as well as for calculating significant places, is code that many applications would be able to use across a variety of applications. Even more value for the developer comes from the modularity behind Phenom. Specifically, the Phenom-based implementation above will instantly be able to take advantage of any improvements made to earlier parts of the process without changing any lines of code (e.g. collecting data from more sources, automatically collecting location data even before this application was installed, an improved algorithm for detecting significant places, or user-provided ground truth on which places are or are not significant). This means that the developer could deploy her application and not need to make any changes in order to receive these benefits.

By contrast, for the non-Phenom implementation, the developer had to make choices on which data to include in the process. Adding another data source to the existing data source means more coding for the developer, both for accessing the data, but also for integrating it. Adding two new data sources is twice as much work. Furthermore, the non-Phenom implementation is unlikely to receive corrections to significant place labels from the user. If the developer wanted this information, she would need to implement a mechanism for the user to provide it. However, even with such an implementation, the likelihood of a user providing feedback for use in a single application seems low.

One drawback to the existing implementation of Phenom is that the codebase (i.e. for providers, bots, and the schema) is managed centrally: there isn't an immediate mechanism for a developer to add a new data provider, or to contribute her own bot. The most immediate way to address this is to run Phenom as an open source project, where individual developers could submit pull requests for changes that they would like to make.

5.2.2 Ordering Contacts Based On Tie Strength

The next example again demonstrates something that would require many more steps to complete without the assistance of Phenom. In this example, we will make use of the `TieStrengthBot` described earlier in this chapter.

```
Filter contactTieStrengthFilter = new Filter(OntologyClass.Person)
   .projection(Person.NAME, Person.TIE_STRENGTH)
   .orderBy(Person.TIE_STRENGTH, false);

mApiClient.sendQuery(contactTieStrengthFilter, new PhenomCallback() {
   @Override
   public void onSuccess(ArrayList<PhenomObject> objs) {
      setContactOrder(objs);
   }
});
```

After retrieving the ordered list of contacts, the application can use that information to determine which contacts to show more prominently. One interesting example where this could be applied might be in an Email application: the email inbox might first group emails by day, but within each day it could show emails first from people that the user is closer to, and then show other emails below.

Non-Phenom Implementation

Without Phenom, the first step to implementing this example is to get programmatic access to the user's call and SMS logs. For this example, we're trying to calculate the number of phone calls in the call log, number of SMS messages in the SMS log, and the total duration of calls in the call log. There are two main approaches for doing this, and each has tradeoffs.

One alternative is to take a more SQL-centric approach to calculating the communication statistics. This involves making SQL group-by queries that groups the communication log tables by each contact and uses SQL aggregates (i.e. `COUNT()` and `SUM(duration)`). This solution might typically require the fewest lines of code, but because of the structure of the Android Content Providers, this process has several problems. First, there is no way to join between the CallLogs provider and the Contacts provider. Doing a GROUP BY on the phone number column is tempting, but the implementation is such that the same phone number might be represented by different strings, even on the same device (e.g. dashes, parentheses, leading country code). In this case, the best solution is that the CallLogs provider does have a cached URI for each contact, which can be used in the GROUP BY clause. However, this information is not guaranteed to be updated as contact records change. Finally, the Android Content Provider API does not support GROUP BY anyway, so it turns out that this approach is simply not possible

The other approach is to calculate those statistics in the Java code of the application. This approach requires writing much more code, but will also be more precise and reliable. Furthermore, if the developer wanted to add some other data that was not already in an SQLite database or Android content provider (e.g. if querying a REST API), then these calculations would need to be done in code.

In spite of this, we will need to pursue the second approach because the first is simply not possible with the current implementation of Android.

```
HashMap<String, Integer> callCount = new HashMap<>();
HashMap<String, Integer> callDuration = new HashMap<>();
HashMap<String, Integer> smsCount = new HashMap<>();

int maxCallCount = 0;
int maxCallDuration = 0;
int maxSMSCount = 0;

Cursor callCursor = this.mContext.getContentResolver().query(
   CallLog.Calls.CONTENT_URI,
   new String[] {
      Calls.DATE,
      Calls.NUMBER,
      Calls.DURATION,
      Calls.CACHED_NAME,
      Calls.CACHED_LOOKUP_URI
   }, null, null, null);

while( callCursor != null && callCursor.moveToNext()){
   String lookupURI = callCursor.getString(
                      callCursor.getColumnIndex(Calls.CACHED_LOOKUP_URI));
   int count = 0;

   if (callCount.get(lookupURI) != null)
      count = callCount.get(lookupURI);

   callCount.put(lookupURI, ++count);

   maxCallCount = Math.max(maxCallCount, count);

   int duration = 0;

   if (callDuration.get(lookupURI) != null)
      duration = callDuration.get(lookupURI);

   duration += callCursor.getInt(
                  callCursor.getColumnIndex(Calls.DURATION));

   callDuration.put(lookupURI, duration);

   maxCallDuration = Math.max(maxCallDuration, duration);
}

callCursor.close();

Cursor smsInboxCursor = this.mContext.getContentResolver().query(
   Sms.Inbox.CONTENT_URI,
   new String[] {
      Sms.DATE,
      Sms.ADDRESS,
   }, null, null, null);

Cursor smsSentCursor = this.mContext.getContentResolver().query(
   Sms.Sent.CONTENT_URI,
   new String[] {
      Sms.DATE,
      Sms.ADDRESS,
   }, null, null, null);

for(Cursor c : new Cursor [] {smsInboxCursor, smsSentCursor}){
   while(c != null && c.moveToNext()){
      String phoneNumber = c.getString(
                      c.getColumnIndex(Sms.ADDRESS));

        /* Note that the method below doesn’t exist so must be implemented
         * and requires a call to the contacts provider
         */
      String lookupURI = getLookupUriForNumber(phoneNumber);

      int count = 0;

      if (smsCount.get(lookupURI) != null)
         count = smsCount.get(lookupURI);

      smsCount.put(lookupURI, ++count);
      maxSmsCount = Math.max(maxSmsCount, count);
   }
}
smsInboxCursor.close();
smsSentCursor.close();
```

So the code above produces the call counts, total call duration, and SMS counts for each contact. There are a couple of things to note in the inconsistency between the APIs for the calls and SMSs. First, note that SMSs are split into different tables, depending on whether they are incoming, outgoing, drafts, etc. Thus, the developer needs to know to query both the inbox and the sent SMSs. Additionally, the SMS Content Provider does not provide the cached lookup URI, so that information has to be retrieved manually in the code from the Contacts Content Provider.

The next step is to calculate a tie strength score for each contact.

```
HashMap<String, Double> tieStrength = new HashMap<>();

for(Entry<String, Integer> e : callCount.entrySet()){
   String lookupURI = e.getKey();
   int callCount = e.getValue();
   int callDuration = callDuration.get(lookupURI);
   int smsCount = 0;

   if((int val = smsCount.remove(lookupURI)) != null)
      smsCount = val;

   double tieStrengthVal = (callCount/maxCallCount/3) +
                           (callDuration/maxCallDuration/3) +
                           (smsCount/maxSmsCount/3);


   tieStrength.put(lookupURI, tieStrengthVal);

}

// For the remaining SMS counts, where a contact didn’t have any calls

for(Entry<String, Integer> e : smsCount.entrySet()){
   String lookupURI = e.getKey();
   int smsCount = e.getValue();

   double tieStrengthVal = (smsCount/maxSmsCount/3);
   tieStrength.put(lookupURI, tieStrengthVal);
}
```

The last step is to sort the HashMap by its values, so that the highest tie strength values are at the top of the list. That code is omitted here.

Comparing Implementations

Again, as with the previous implementation example, it is clear that the Phenom implementation is much simpler for an application developer. The Phenom solution is more robust as well: it does not rely on the cached contact information or the phone's contact list. This also means that adding in additional communication data (e.g. emails, social network, or instant messaging) would be easier with Phenom than in the custom implementation. Finally, the Phenom implementation can automatically and for free take advantage of any ground truth data provided by the user (i.e. fixing incorrectly labeled contacts whose real tie strength does not match that which was calculated).

The reason that this approach works for Phenom is because there are many applications that might be able to make use of communication metadata, and of tie strength (e.g. contact ordering, notification prioritization, personal informatics). Additionally, these are both potentially useful as input to even higher-level inferences (e.g. mental health, social support, busyness). Thus, the code that supports this in Phenom is valuable because it can be reused across a variety of applications, eliminating the need for any of those developers to redo the common steps of these processes.

5.3 Discussion

Phenom is a proof of concept system that demonstrates the possibility and the power of an integrated service for managing personal data on the level of the individual rather than on the level of a company or data source.

The architecture of Phenom described in this implementation organizes the personal data process into a modular set of reusable components that are flexible enough to store arbitrary types of personal data, support the linkages between personal data regardless of whether they are from the same or different sources, generate inferences and abstractions on the data, and provide access to that data through a unified API. As a result, individual applications do not need to solve the issues and challenges associated with storing personal data, those responsibilities can be delegated to Phenom and solved once.

5.3.1 Reflecting On Design Goals

Section 4.5 laid out an ambitious set of design goals targeted at addressing the issues and challenges that are associated with the current ecosystem of personal data. While no single developer, system, or approach could possibly address the entirety of these goals singlehandedly, the implementation of Phenom that is described in this chapter represents an important step towards reaching these goals. Phenom speaks directly to some of these goals, and indirectly to others. Each of those goals is discussed in turn below.

Minimize redundant effort required of developers

Phenom dramatically reduces the net effort that is required of developers in order to make use of personal data, and each of the components of Phenom contribute to this. Through the abstraction of Data Providers, Phenom simplifies the process of working with multiple APIs and the only developer who needs to be concerned with the structure of the API of the data source is the developer who implements the data provider for that data source. The Epistenet data store supports rich interconnection between data, both homogeneous through the semantic network of `OntologyClasses`, and also heterogeneous through the use of `ReferenceAttributes`. The combination of these two types of interconnection is especially powerful. Finally, the API offers even more value to developers by simplifying operations that would otherwise be complicated and would require a deeper understanding of the underlying implementation.

Phenom's Bots offer the ability to support the reuse of machine learning by enabling the modular deployment of models that can generate inferences and abstractions based on the contents of Epistenet. This represents a good first step, but more can be done to further streamline the process of developing machine learning models. Part of that opportunity comes from better mechanisms for collecting ground truth and retraining models. Another opportunity is to provide better support for the actual process of coming up with an initial model. With the current design of Phenom this remains a challenge because Phenom does not offer developers who are implementing bots any way to access the personal data of individuals, even if a user is willing to offer their data.

The real value of Phenom on this design goal is visible through the two examples highlighted in section 5.2. The amount of code required, the complexity, and the potential to make errors was dramatically better with Phenom than without it. The Phenom solutions are easier, more robust, and more flexible to future additions and improvements to the process.

Organize data by individual, not by service

Phenom addresses this goal very directly: in Phenom, the top level of organization for personal data is the user, not the service or data source that the data came from.

Support connections within the data

Again, Phenom directly engages the goal of supporting connections within the data. The main limitations of supporting connections within the data now likes in the hierarchical organization of the ontology, and in the choice of `ReferencesAttributes` to associate with a particular `OntologyClass`.

Limit unnecessary disclosure

Phenom's API easily supports many queries that would have previously required the developer to have access to copious amounts of raw personal data. Not only does this capability help to minimize redundant effort by developers, but it also lays the groundwork for a system that offers users strong guarantees on how much data is being accessed by developers. While Phenom does not fully implement that system, it is now conceivable to do.

Offer users transparency, and offer users choices and control, while specifying reasonable defaults

These final two personal data design goals remain mostly untouched by Phenom, and are ripe for implementation and further work.

5.3.2 Next Steps

Phenom is a big idea and it represents a major shift in the approach to handling personal data. However, as the previous section suggested, there are a number of important aspects of Phenom that will need further development in order to realize its full potential.

Privacy

Without question, the topic of privacy is the aspect of Phenom requires the most attention. However, developing a strong approach to privacy here is a large topic that will require significant additional work.

One approach to handling privacy in Phenom is to simply continue to enforce the Android permissions framework that is already in use on the platform. This implementation would involve tracking the permissions that were required to obtain all of the data that was used in the specification of a certain query, and ensuring that the client application has declared all of those permissions in its own `AndroidManifest.xml` file. This approach is in some ways the most obvious and probably the simplest to implement as well, however there are several problems with this approach. First, querying for something like tie strength, for example, would require Android's permissions for contacts, call logs, and SMS logs, even though neither call logs nor SMS logs are directly accessible by the developer, and it would be very difficult to infer much meaning from the value produced by the tie strength bot (except to infer that the user has shared no communication with a particular contact). This approach is suboptimal because it does not allow for stronger guarantees on what data an application is or not accessing, which is an important aspect of the design goals. The next problem is that some data that Phenom aggregates or will be able to aggregate in the future does not have an Android permission (and did not come from Android in the first place). In these cases, a different permission approach would be necessary because developers would not have access to a permission that they should declare for those data types.

Another approach for handing privacy controls in Phenom is to define custom permissions for new types of data (e.g. a permission for tie strength, a permission for accessing aggregated statistics on calls, etc.). This is approach is a permutation of the previous approach that at least offers some solutions to the aforementioned issues. While this approach is more plausible, it suffers from an unfortunate tradeoff. In order for this approach to guarantee minimum access, it will likely result in an explosion of new permissions for every possible permutation of different combinations of data that might be accessed. This will be unwieldy for developers, and certainly will make the development process more complex. However, even more troubling is that users are likely to be overwhelmed or confused by the explosion of new preferences. This could result in the average user paying less attention to privacy preferences. Studies have already shown that users often do not understand the existing permissions (Kelley et al., 2012).

Finally, a more progressive approach would be to rethink the permissions system more holistically. One idea in this directions is the idea of tiered permissions. The idea behind tiered permissions is that some kinds of data are considered to be less sensitive than others, and so permission to access this data should be presented differently to the user. For example, something as simple as how recently the user made a phone call, how many contacts are in the contact list, or the average number of text messages the user sends per month are all likely to be perceived as less sensitive, so these items might appear at a lower tier. By contrast, the exact location of the user's home and work, her entire call log, or the amount of money the user has in her bank account might be considered more sensitive items and belong in the top tier. There are probably different kinds of data that belong in between these two as well. For example, tie strength seems like it might be in between the two extremes. The idea with this approach is that lower tier items would require less confirmation from the user in order to access, while higher tier things might be especially prominent to encourage the user to be cautious.

This tiered approach feels promising, but also introduces its own challenges. For example, deciding what belongs in each tier requires non-trivial effort. Furthermore, adding additional bots or data sources is going to require even more decisions to be made. Finally, there is the question of this approach to the vulnerability of an inference attack. For example, the tiered permissions model may determine that simply knowing whether or not the user is currently at home is less sensitive than the exact location of the user's home. However, if the application can gain access to the user's current location in some other way, then the developer still has access to the more sensitive information. In this particular example, perhaps Phenom would check to be sure that the application has not declared permissions to access Android's location APIs. However, there are many different combinations of inference attacks that might occur, and it seems intractable to be able to protect against all of them. Even the combination of different data that are accessed within Phenom might change something that was otherwise not very sensitive into something that was very sensitive. For example, it has been demonstrated that having access to an individual's date of birth and place of birth (both fairly innocuous facts on their own), can be exploited to guess the considerably sensitive information of the individual's social security number (Acquisti & Gross, 2009). Ultimately, the challenges of creating usable interfaces for managing privacy and security are so difficult that an entire research area has developed to understand and address these challenges. This topic is a rich space for future work.

Ground Truth and Mediation

Beyond privacy, there are a number of opportunities to push Phenom forward. First is providing users with opportunities for correcting incorrect inferences and providing ground truth data to help improve inference mechanisms. It is conceivable that individuals would be willing to provide better labels for their own data if in exchange they receive better service. Existing examples of this include features like Netflix asking users to rate more movies so that they get better recommendations, and Gmail Priority Inbox asking users to select which items should be moved to the Priority Inbox and which items should be removed. There are many opportunities for individual applications to encourage this type of labeling and Phenom should provide a process for integrating with that. Also along these lines, as discussed in the previous section, there are opportunities to further simplify the process of developing machine learning models and improving those models after they have been deployed.

Externally Defined Providers

Next, it would be very useful for Phenom to accept incoming data from external data providers. This feature would allow client applications that otherwise do not want to expend resources to provide and maintain an API to still contribute that data to Phenom and thus offer access and control of that data to the user. This will lead to other important challenges to consider. For example, what if the data that an application wants to contribute to Phenom does not fit into the existing ontology or requires an additional attribute? The way that Phenom is implemented today, those decisions are made statically at compile time. However, in the future it is possible that the ontology definition and the attributes associated with a particular ontology class could be dynamically defined. Such an implementation would need to have a centralized component for handling the definition of the ontology. Otherwise, a decentralized version would mean that a developer could never depend on what ontology is implemented on a particular device, which is problematic from a development perspective.

Architecture

The topic of a centralized component for storing a dynamically changing ontology also leads to a broader discussion of the particular architecture that Phenom is implemented in today. Phenom is quite decentralized in its current implementation, with an instance of Phenom running on the phone of each user. This has a variety of tradeoffs:

- Individuals may feel more secure that their data is physically in their control on their own device. The reverse perspective is that a smartphone is much easier to physically steal than if the data was stored in the cloud.

- There is no centralized cost for owning and maintaining servers, including the processing power, storage capacity, and electricity costs. This means that it might be easier to spark adoption of Phenom because there is no cost barrier to starting to use it. The reverse perspective here is that resources on a smartphone are certainly limited: storage space, processing power, and battery. If Phenom really became popular, its impact on the resources of the device might become more salient to the user.

- Because Phenom is decentralized, there is no real support for non-phone applications to gain access to Phenom. This could become an issue, in particular if part of the value of Phenom is to offer a consistent user experience across all of the applications that an individual uses.

If we collectively adopted a computing architecture more akin to one that would support the proposed Personal Server (Want et al., 2002), but given the widespread success of mobile data and cloud computing, the idea of changing our computing infrastructure to support this idea of a Personal Server seems unlikely.

With a centralized architecture, the issues and concerns would be reversed. A third option to consider is the potential to support a hybrid architecture, with some components of Phenom centralized, and others decentralized. Such an approach might begin to offer the benefits of each approach while minimizing the drawbacks. One example of this idea of a hybrid decentralized platform is the social network platform Diaspora[18]. In Diaspora, the idea was that any individual could host their own server (called a pod), and that pods could connect with each other, but physical control of the server and the personal data is decentralized. Ultimately, a hybrid architecture would represent a massive undertaking, but may also offer the most promise for deploying the ideas behind Phenom out into the real world.

5.4 Related Work

Aspects of the approach that Phenom takes to handling personal data are related to a variety of project in the space of HCI and mobile computing. While Chapter 2 gave a much broader overview of various work related to personal data, this section is mainly focused on systems that may appear similar or related to Phenom.

The Context Toolkit (A. Dey et al., 2001) is a software framework for making software context-aware. In the context toolkit, data is collected from sensors by *context widgets* that separate the data that was collected from the specific complexity of

[18] https://diasporafoundation.org/

how it was collected, *interpreters* raise the level of abstraction of the data within each sensor, *aggregators* bring together related contextual information together from different sensors, *services* trigger actions based on the data, and *discoverers* maintain a registry of what capabilities exist in the framework. Phenom was inspired in part by The Context Toolkit. The most obvious difference between these two systems is that The Context Toolkit is designed to bridge the gap between very low-level, sensor-based personal data. By contrast, Phenom is not designed to handle very low level sensor data but is much more focused on accepting the output of a system such as the Context Toolkit.

Following on from The Context Toolkit, a number of frameworks and tools have been developed that further expand the idea that underlies The Context Toolkit. In particular, following the creation and widespread adoption of the Android operating system, a handful of tools have emerged that are focused on offering a unified framework for interacting with a phone's contextual data, whose definition has in some instances been expanded beyond hardware sensors to include data from "software sensors" and even humans. These systems include the AWARE framework (Ferreira, Kostakos, & Dey, 2015), ohmage (Ramanathan et al., 2012), and the Funf Open Sensing Framework (Aharony et al., 2011). While the specific details of the implementations of these systems do vary, the basic structure is fairly similar across all of these systems. All three of these systems have a strong focus on collecting data in the context of a study: they all include a backend server component and tools for researchers to collect and analyze data from participants. In addition to these features, all three systems do offer a library that contains the core components for developers to integrate the framework into the development of their own applications.

These systems do share some aspects of similarity with Phenom: they all run on Android, they all collect personal data, and in some cases (particularly with AWARE), there is some effort to raise the level of abstraction of the data beyond the level it was collected at. However, Phenom stands distinct from these systems. Perhaps most distinct is the combination of Phenom's semantic data store and Phenom's API. None of the three systems mentioned above support linking and interconnection of data across different data types. Instead, they all expose the underlying personal data through a very thin API layer. Phenom's API offers the ability to easily specify complex cross-data-type queries. Furthermore, the ontological hierarchy in Epistenet offers additional power and flexibility in working with the data. Finally, Phenom's framing as a service for managing the breadth of personal data is distinct from those presented in the above systems, where the focus is more directly on the information that is available on the phone.

One idea that was proposed is that the phone's operating system is what should be responsible for collecting and making inferences from contextual data (Chu, Kansal, Liu, & Zhao, 2011). This offers a different perspective on collecting personal data from a smartphone. For example, operating system-level support for collecting context could provide unified support for collecting user behavior within applications (Fernandes, Riva, & Nath, 2015). This is a perfect example of the kind of data that

Phenom would be perfect for collecting: information about what users do when they are in applications could dramatically improve the amount of data from which we can make personal inferences in Phenom. Again, the level of inference described in this work is at the lower levels of contextual inference, where Phenom is positioned to be making higher-level inferences.

A number of personal data stores have been proposed over the years, with various architectures, access mechanisms, and privacy controls (Bell, 2001; Cáceres, Cox, Lim, Shakimov, & Varshavsky, 2009; de Montjoye, Shmueli, Wang, & Pentland, 2014; “Higgins Personal Data Service,” n.d.; Hong & Landay, 2004; Mun et al., 2010; Want et al., 2002). The motivations behind these systems echo each other: offering users ownership and control over her personal data, strongly emphasizing privacy. Echoing the points above, Phenom’s unique approach to storing, interconnecting, and querying the data makes it distinct from these other approaches. Furthermore, Phenom’s bots offer additional functionality for making inferences and abstractions internally in the system. The related work that offers something most similar is openPDS (de Montjoye et al., 2014). openPDS includes a component called SafeAnswers. SafeAnswers essentially offers functionality complimentary to bots. However, in the SafeAnswers model, individual developers are responsible for writing the code that will be run in the system, and the only data that is released to the developer is the answer to the question. By contrast, Phenom’s Bots are intended to be highly-reusable, application-agnostic modules. Furthermore, because the output of bots is also stored in the semantic data store, Bot output can be easily and flexibly combined with other parts of the user’s data, a functionality not supported by the SafeAnswers architecture.

Recently, both Google and Apple have released platforms (Fit (“Google Fit,” n.d.) and HealthKit (“Apple HealthKit,” n.d.) respectively), which share some aspects with Phenom: they are intended to collect fitness and health data from arbitrary applications, store that data in a data-centric format rather than a source-centric one, and then make that data available to other applications with the user’s permission. The approach taken here is in some ways closer to Phenom’s ontology-class-driven semantic approach to organizing data. However, these systems are restricted to health data, they lack the ability to generate inferences within the system, and they do not provide the same interconnected API querying facilities that Phenom offers.

6 Conclusion

Personal data today is abundant, and there remains enormous potential for it to grow both in the breadth of sources captured and in the duration of time captured. Applications that make use of this data are limited only by our own creativity. But between the vast amounts of personal data and the functional applications that are enabled by the data lies an orthogonal set of challenges. The ecosystem of personal data was not purposefully designed with a goal of unlocking the full potential of a collected and quantified world. In fact, it seems that nobody at all has approached personal data from a holistic perspective.

This dissertation explores a holistic view of personal data. A broad survey of computer science in chapter 2 research reveals multiple domains where an integrated approach to personal data is the key to advancing the state of the art in that discipline. The case study in chapter 3 demonstrates the practical challenges and issues that transform the simple steps of a research project into a resource-intensive distraction from the main goal of the work. Chapter 4 explores the ecosystem of personal data: What does it look like today? What is wrong with it? What would improve it? It introduces a conceptual framework for thinking about the process of working with personal data consisting of a continuum of abstraction levels of personal data, and three steps necessary for working with it. Using the frame of unified personal data, simplifies many of the challenges involved in this process. Chapter 5 demonstrates a proof-of-concept service for unified personal data that offers a single user-centric data store of richly interconnected personal data.

This dissertation offers the following technical and design contributions to HCI:

1. A proposal for unified personal data; a reframing of many HCI challenges, human needs, and technical opportunities that can all be advanced by more

holistically viewing all of the individual data amassing around people as their personal data that should work for them.

2. The notion of personal data as a continuum, and a conceptual framework that unpacks the implicit process involved in working with personal data.
3. A set of design goals for improving the ecosystem of personal data.
4. The design of Phenom: a service that supports software development with personal data. Phenom modularizes the collection, interconnection, processing, and querying of personal data to solve a key set of challenges involved in developing applications that use personal data.
5. The implementation of a proof of concept of Phenom which demonstrates its viability and utility as a personal data service.

Personal data is only in its beginnings as a research domain. If researchers from many disciplines are going to continue to employ personal data to make research advances in their own disciplines, it is imperative that we establish this multi-disciplinary domain.

The possibility of a world where unified personal data can be used to enable powerful and complex applications is very real, however many important and interconnected questions remain in personal data research. What economic model will enable companies to maintain their value and competitive advantage while also enabling end-users fair access to their data? What software architecture offers the best compromise of across concerns? What access mechanisms will offer an effective balance between privacy and utility?

Even beyond research, as a society we will need to answer a set of questions that we might not be ready for. Who "owns" my personal data? Is ownership even the most applicable concept? Does an individual have a right to access their own data? A right to demand that it is collected? A right to demand that it is deleted? A right to stop it from being deleted? In the context of these questions, Phenom is a software artifact that offers the ability to engage these questions, explore potential solutions, and continue to evolve the ecosystem of personal data.

7 References

Abbar, S., Bouzeghoub, M., & Lopez, S. (2009). Context-aware recommender systems: A service-oriented approach. In *VLDB PersDB workshop* (pp. 1–6).

Ackerman, J. M., Kenrick, D. T., & Schaller, M. (2007). Is friendship akin to kinship? *Evolution and Human Behavior*, *28*(5).

Ackerman, M. S., Cranor, L. F., & Reagle, J. (1999). Privacy in e-commerce: examining user scenarios and privacy preferences. In *Proceedings of the 1st ACM conference on Electronic commerce* (pp. 1–8).

Acquisti, A., & Gross, R. (2009). Predicting Social Security numbers from public data. *Proceedings of the National Academy of Sciences of the United States of America*, *106*(27), 10975–80. doi:10.1073/pnas.0904891106

Adar, E., Karger, D., & Stein, L. A. (1999). Haystack: Per-user Information Environments. In *Proceedings of the Eighth International Conference on Information and Knowledge Management* (pp. 413–422). New York, NY, USA: ACM. doi:10.1145/319950.323231

Adomavicius, G., & Tuzhilin, A. (2011). Context-aware recommender systems. In *Recommender systems handbook* (pp. 217–253). Springer.

Aharony, N., Pan, W., Ip, C., Khayal, I., & Pentland, A. (2011). Social fMRI: Investigating and shaping social mechanisms in the real world. *Pervasive and Mobile Computing*, *7*(6), 643–659. doi:http://dx.doi.org/10.1016/j.pmcj.2011.09.004

Allen, J. F. (1979). *A Plan-based Approach to Speech Act Recognition*.

American Psychiatric Association. (2013). *Diagnostic and Statistical Manual of Mental Disorders (DSM-5)*. American Psychiatric Pub.

Apple HealthKit. (n.d.). Retrieved August 10, 2015, from https://developer.apple.com/healthkit/

Assad, M., Carmichael, D., Kay, J., & Kummerfeld, B. (2007). PersonisAD: Distributed, active, scrutable model framework for context-aware services. *Pervasive Computing*, 55–72.

Baldauf, M., Dustdar, S., & Rosenberg, F. (2007). A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, *2*(4), 263–277.

Barkhuus, L., Brown, B., Bell, M., Sherwood, S., Hall, M., & Chalmers, M. (2008). From awareness to repartee: sharing location within social groups. *CHI '08: Proceeding of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems*. Retrieved from http://portal.acm.org/citation.cfm?id=1357054.1357134

Belk, R. (2010). Sharing. *Journal of Consumer Research*, *36*(5), 715–734. doi:10.1086/612649

Belk, R. W. (1988). Possessions and the Extended Self. *The Journal of Consumer Research*, *15*(2), 139–168.

Bell, G. (2001). A personal digital store. *Communications of the ACM*, *44*(1), 86–91.

Bellotti, V., Dalal, B., Good, N., Flynn, P., & Bobrow, D. (2004). What a to-do: studies of task management towards the design of a personal task list manager. In *In Proceedings of the SIGCHI conference on Human factors in computing systems*. Retrieved from http://portal.acm.org/citation.cfm?id=985785

Bernstein, M., Van Kleek, M., Karger, D., & Schraefel, M. (2008). Information scraps: How and why information eludes our personal information management tools. *ACM Transactions on Information Systems (TOIS)*, *26*(4).

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, *46*, 109–132.

Brandimarte, L., & Acquisti, A. (2012). The Economics of Privacy. In *The Oxford Handbook of the Digital Economy*. Oxford University Press.

Brandt, J., Weiss, N., & Klemmer, S. (2007). txt 4 l8r: lowering the burden for diary studies under mobile conditions. *CHI '07: CHI '07 Extended Abstracts on Human Factors in Computing Systems*.

Brown, B., Taylor, A., Izadi, S., Sellen, A., Kaye, J., & Eardley, R. (2007). Location family values: A field trial of the whereabouts clock. *Ubiquitous Computing (Ubicomp '07)*.

Browne, G., Berry, E., Kapur, N., Hodges, S., Smyth, G., Watson, P., & Wood, K. (2011). SenseCam improves memory for recent events and quality of life in a patient with memory retrieval difficulties. *Memory*, *19*(7), 713–722.

Burke, M. (2011). *Reading, Writing, Relationships: The Impact of Social Network Sites on Relationships and Well-Being*. Carnegie Mellon University.

Burke, M., & Kraut, R. (2013). Using Facebook after losing a job: Differential benefits of strong and weak ties. In *Proceedings of the 2013 conference on Computer supported cooperative work* (pp.

1419–1430).

Burton, R. R., & Brown, J. S. (1979). An investigation of computer coaching for informal learning activities. *International Journal of Man-Machine Studies*, *11*(1), 5–24. doi:http://dx.doi.org/10.1016/S0020-7373(79)80003-6

Bush, V. (1945, July). As we may think. *The Atlantic Monthly*. doi:http://dx.doi.org/10.1145/227181.227186

Cáceres, R., Cox, L., Lim, H., Shakimov, A., & Varshavsky, A. (2009). Virtual individual servers as privacy-preserving proxies for mobile devices. In *Proceedings of the 1st ACM workshop on Networking, systems, and applications for mobile handhelds* (pp. 37–42).

Cadiz, J., Venolia, G., & Jancke, G. (2002). Designing and deploying an information awareness interface. In *In Proceedings of the 2002 ACM conference on Computer supported cooperative work*.

Chang, K. S.-P., Myers, B. A., Cahill, G. M., Simanta, S., Morris, E., & Lewis, G. (2013). Improving Structured Data Entry on Mobile Devices. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (pp. 75–84). New York, NY, USA: ACM. doi:10.1145/2501988.2502043

Chen, G., & Kotz, D. (2000). *A survey of context-aware mobile computing research*.

Chu, D., Kansal, A., Liu, J., & Zhao, F. (2011). Mobile Apps: It's Time to Move Up to CondOS. In *Proceedings of the 13th USENIX conference on Hot topics in operating systems* (p. 16).

Cohen, P. R., & Perrault, C. R. (1979). Elements of a plan-based theory of speech acts. *Cognitive Science*, *3*(3), 177–212. doi:10.1016/S0364-0213(79)80006-3

Consolvo, S., McDonald, D. W., Toscos, T., Chen, M. Y., Froehlich, J., Harrison, B., … Landay, J. a. (2008). Activity sensing in the wild: a field trial of ubifit garden. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)* (pp. 1797–1806). doi:10.1145/1357054.1357335

Conti, M., Passarella, A., & Pezzoni, F. (2011, June 20). A model for the generation of social network graphs. *2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*. IEEE. doi:10.1109/WoWMoM.2011.5986141

Cummings, J. N., Lee, J. B., & Kraut, R. (2006). Communication technology and friends during the transition from high school to college. In *Computers, phones, and the Internet: Domesticating information technology*.

Danezis, G. (2009). Inferring Privacy Policies for Social Networking Services. In *Conference on Computer and Communications Security* (pp. 5–10).

Das, S., Hayashi, E., & Hong, J. I. (2013). Exploring capturable everyday memory for autobiographical authentication. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing - UbiComp '13* (p. 211). New York, New York, USA: ACM Press. doi:10.1145/2493432.2493453

Davidoff, S., Lee, M. K., Dey, A. K., & Zimmerman, J. (2007). Rapidly exploring application design through speed dating. *LNCS*. Retrieved from http://www.springerlink.com/index/w174666l525j2741.pdf

Davidoff, S., Ziebart, B. D., Zimmerman, J., & Dey, A. K. (2011). Learning patterns of pick-ups and drop-offs to support busy family coordination. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1175–1184).

de Montjoye, Y.-A., Shmueli, E., Wang, S. S., & Pentland, A. S. (2014). openPDS: protecting the privacy of metadata through SafeAnswers. *PloS One*, *9*(7), e98790. doi:10.1371/journal.pone.0098790

Dey, A. K. (2001). Understanding and Using Context. *Personal Ubiquitous Comput.*, *5*(1), 4–7. doi:10.1007/s007790170019

Dey, A., Salber, D., & Abowd, G. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, *12*(2), 97–166.

Dim, E., Kuflik, T., & Reinhartz-Berger, I. (2015). When User Modeling Intersects Software Engineering: The Info-bead User Modeling Approach. *User Modeling and User-Adapted Interaction*, *25*(3), 189–229. doi:10.1007/s11257-015-9159-1

Dorst, K. (2011). The core of "design thinking"and its application. *Design Studies*, *32*(6), 521–532.

Doryab, A., Min, J. K., Wiese, J., Zimmerman, J., & Hong, J. I. (2014). Detection of behavior change in people with depression. In *AAAI Workshops Workshops at the Twenty-Eighth AAAI Conference on Artificial Intelligence*.

Dourish, P. (2001). Seeking a foundation for context-aware computing. *Human--Computer Interaction*, *16*(2-4), 229–241.

Dourish, P. (2004). What we talk about when we talk about context. *Personal Ubiquitous Comput.* doi:http://dx.doi.org/10.1007/s00779-003-0253-8

Ducheneaut, N., & Bellotti, V. (2001). E-mail as habitat: an exploration of embedded personal information management. *Interactions*, *8*(5), 30–38. doi:10.1145/382899.383305

Dumais, S., Cutrell, E., Cadiz, J. J., Jancke, G., Sarin, R., & Robbins, D. C. (2003). Stuff I've seen: a system for personal information retrieval and re-use. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval* (pp. 72–79).

Eagle, N., & Pentland, A. (2006). Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, *10*(4), 255–268.

Eagle, N., Pentland, A., & Lazer, D. (2009). Inferring Social Network Structure using Mobile Phone Data. *PNAS*, *106*(36).

Estrin, D. (2014). Small data, where n= me. *Communications of the ACM*, *57*(4), 32–34.

Fang, L., & LeFevre, K. (2010). Privacy wizards for social networking sites. In *Proceedings of the 19th* *international conference on World wide web* (pp. 351–360).

Farnham, S. D., & Churchill, E. F. (2011). Faceted identity, faceted lives. *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work - CSCW '11*, 359. doi:10.1145/1958824.1958880

Fernandes, E., Riva, O., & Nath, S. (2015). My OS ought to know me better: In-app behavioural analytics as an OS service. In *15th Workshop on Hot Topics in Operating Systems (HotOS XV)*.

Ferreira, D., Kostakos, V., & Dey, A. K. (2015). AWARE: Mobile Context Instrumentation Framework. *Frontiers in ICT*, *2*. doi:10.3389/fict.2015.00006

Fisher, D., DeLine, R., Czerwinski, M., & Drucker, S. (2012). Interactions with Big Data Analytics. *Interactions*, *19*(3), 50–59. doi:10.1145/2168931.2168943

Fogarty, J., Lai, J., & Christensen, J. (2004). Presence versus availability: the design and evaluation of a context-aware communication client. *International Journal of Human-Computer Studies*.

Freeman, E., & Fertig, S. (1995). Lifestreams: Organizing your electronic life. In *AAAI Fall Symposium: AI Applications in Knowledge Navigation and Retrieval* (pp. 38–44).

Freeman, E., & Gelernter, D. (1996). Lifestreams: A storage model for personal data. *ACM SIGMOD Record*, *25*(1), 80–86.

Friedkin, N. (1980). A test of structural features of Granovetter's strength of weak ties theory. *Social Networks*, *2*(4), 411–422.

Gemmell, J., Bell, G., & Lueder, R. (2006). MyLifeBits. *Communications of the ACM*, *49*(1), 88–95. doi:10.1145/1107458.1107460

Gemmell, J., Bell, G., Lueder, R., Drucker, S., & Wong, C. (2002). MyLifeBits: Fulfilling the Memex Vision. In *Proceedings of the Tenth ACM International Conference on Multimedia* (pp. 235–238). New York, NY, USA: ACM. doi:10.1145/641007.641053

Gilbert, E., & Karahalios, K. (2009). Predicting tie strength with social media. In *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 211–220).

Gliem, J. A., & Gliem, R. R. (2003). Calculating, Interpreting, And Reporting Cronbach's Alpha Reliability Coefficient For Likert-Type Scales.

Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *Commun. ACM*, *35*(12), 61–70. doi:10.1145/138859.138867

González, M., Hidalgo, C., & Barabási, A. (2008). Understanding individual human mobility patterns. *Nature*.

Google Fit. (n.d.). Retrieved August 10, 2015, from https://developers.google.com/fit/?hl=en

Granovetter, M. (1973). The strength of weak ties. *The American Journal of Sociology*,.

Gulotta, R., Odom, W., Forlizzi, J., & Faste, H. (2013). Digital Artifacts As Legacy: Exploring the

Lifespan and Value of Digital Data. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1813–1822). New York, NY, USA: ACM. doi:10.1145/2470654.2466240

Higgins Personal Data Service. (n.d.). Retrieved August 10, 2015, from http://www.eclipse.org/higgins/

Hill, R. A., & Dunbar, R. I. M. (2003). Social network size in humans. *Human Nature*, *14*, 53–72.

Hodges, S., Williams, L., Berry, E., Izadi, S., Srinivasan, J., Butler, A., … Wood, K. (2006). SenseCam: A retrospective memory aid. In *UbiComp 2006: Ubiquitous Computing* (pp. 177–193). Springer.

Hong, J., & Landay, J. (2004). An Architecture for Privacy-Sensitive Ubiquitous Computing. In *The Second International Conference on Mobile Systems, Applications, and Services (MobiSys 2004)* (pp. 177–189).

Hori, T., & Aizawa, K. (2003). Context-based Video Retrieval System for the Life-log Applications. In *Proceedings of the 5th ACM SIGMM International Workshop on Multimedia Information Retrieval* (pp. 31–38). New York, NY, USA: ACM. doi:10.1145/973264.973270

Hsieh, G., Tang, K. P., Low, W. Y., & Hong, J. I. (2007). Field Deployment of IMBuddy: A Study of Privacy Control and Feedback Mechanisms for Contextual Instant Messengers. In *The Ninth International Conference on Ubiquitous Computing (Ubicomp 2007).*

Iachello, G., & Hong, J. (2007). End-User Privacy in Human-Computer Interaction. *FNT in Human-Computer Interaction*. doi:10.1561/1100000004

Jones, S., & O'Neill, E. (2010). Feasibility of Structural Network Clustering for Group-Based Privacy Control in Social Networks. In *Symposium on Usable Privacy and Security (SOUPS 2010).*

Jones, W. (2007). Personal information management. *Annual Review of Information Science and Technology*, *41*(1), 453–504.

Karger, D. R., & Jones, W. (2006). Data unification in personal information management. *Communications of the ACM*, *49*(1), 77–82.

Kay, J., & Kummerfeld, B. (2010). *PortMe: personal lifelong user modelling portal.* School of Information Technologies, University of Sydney [Sydney].

Kaye, J. "Jofish," Vertesi, J., Avery, S., Dafoe, A., David, S., Onaga, L., … Pinch, T. (2006). To Have and to Hold: Exploring the Personal Archive. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 275–284). New York, NY, USA: ACM. doi:10.1145/1124772.1124814

Kelley, P. G., Bresee, J., Cranor, L. F., & Reeder, R. W. (2009). A nutrition label for privacy. In *Proceedings of the 5th Symposium on Usable Privacy and Security* (p. 4).

Kelley, P. G., Consolvo, S., Cranor, L. F., Jung, J., Sadeh, N., & Wetherall, D. (2012). A conundrum of permissions: installing applications on an android smartphone. In *Financial*

Cryptography and Data Security (pp. 68–79). Springer.

Kelley, P.G., Brewer, R., Mayer, Y., Cranor, L.F., Sadeh, N. (2011). An Investigation into Facebook Friend Grouping. *In Proceedings of INTERACT 2011.*

Kiukkonen, N., Blom, J., Dousse, O., Gatica-Perez, D., & Laurila, J. (2010). Towards rich mobile phone datasets: Lausanne data collection campaign. *Proc. ICPS, Berlin*.

Klemperer, P., Liang, Y., Mazurek, M., Sleeper, M., Ur, B., Bauer, L., … Reiter, M. (2012). Tag, You Can See It!: Using Tags for Access Control in Photo Sharing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 377–386). New York, NY, USA: ACM. doi:10.1145/2207676.2207728

Kobsa, A. (2001). Generic user modeling systems. *User Modeling and User-Adapted Interaction*, *11*(1-2), 49–63.

Konstan, J. A., & Riedl, J. (2012). Recommender systems: from algorithms to user experience. *User Modeling and User-Adapted Interaction*, *22*(1-2), 101–123.

Lamming, M., Brown, P., Carter, K., Eldridge, M., Flynn, M., Louie, G., … Sellen, A. (1994). The design of a human memory prosthesis. *The Computer Journal*, *37*(3), 153–163.

Lazer, D., Pentland, A., Adamic, L., Aral, S., Barabási, A.-L., Brewer, D., … Van Alstyne, M. (2009). Computational Social Science. *Science*, *323*(5915), 721–723. doi:10.1126/science.1167742

Lee, M. L., & Dey, A. K. (2008). Lifelogging memory appliance for people with episodic memory impairment. In *Proceedings of the 10th international conference on Ubiquitous computing* (pp. 44–53).

Li, I., Dey, A., & Forlizzi, J. (2010). A stage-based model of personal informatics systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 557–566).

Lin, J., Amini, S., Hong, J. I., Sadeh, N., Lindqvist, J., & Zhang, J. (2012). Expectation and purpose: understanding users' mental models of mobile app privacy through crowdsourcing. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing* (pp. 501–510).

Lin, N., & Dean, A. (1984). Social support and depression. *Social Psychiatry and Psychiatric Epidemiology*, *19*(2), 83–91.

Lindqvist, J., Cranshaw, J., Wiese, J., Hong, J., & Zimmerman, J. (2011). I'M the Mayor of My House: Examining Why People Use Foursquare - a Social-driven Location Sharing Application. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 2409–2418). New York, NY, USA: ACM. doi:10.1145/1978942.1979295

Liu, Y., Gummadi, K. P., Krishnamurthy, B., & Mislove, A. (2011). Analyzing Facebook Privacy Settings: User Expectations vs. Reality. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference* (pp. 61–70). New York, NY, USA: ACM. doi:10.1145/2068816.2068823

Marcu, G., Dey, A. K., & Kiesler, S. (2012). Parent-driven Use of Wearable Cameras for Autism

Support: A Field Study with Families. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing* (pp. 401–410). New York, NY, USA: ACM. doi:10.1145/2370216.2370277

Marin, A., & Hampton, K. N. (2007). Simplifying the Personal Network Name Generator. *Field Methods*.

Matyas, C., & Schlieder, C. (2009). A spatial user similarity measure for geographic recommender systems. In *GeoSpatial Semantics* (pp. 122–139). Springer.

McCarty, C. (2002). Structure in personal networks. *Journal of Social Structure*, *3*(1).

McEwan, G., & Greenberg, S. (2005). Supporting social worlds with the community bar. In *Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work*.

Mesch, G. S. (2009). Social context and communication channels choice among adolescents. Computers. *Human Behavior*, *25*, 244–251.

Min, J.-K., Doryab, A., Wiese, J., Amini, S., Zimmerman, J., & Hong, J. I. (2014). Toss “n” turn: smartphone as sleep and sleep quality detector. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI ’14* (pp. 477–486). New York, New York, USA: ACM Press. doi:10.1145/2556288.2557220

Min, J.-K., Wiese, J., Hong, J. I., & Zimmerman, J. (2013). Mining smartphone data to classify life-facets of social relationships. In *In Proc. CSCW ’13*. New York, New York, USA. doi:10.1145/2441776.2441810

Miritello, G., Moro, E., Lara, R., Martínez-López, R., Belchamber, J., Roberts, S. G. B., & Dunbar, R. I. M. (2013). Time as a limited resource: Communication strategy in mobile phone networks. *Social Networks*, *35*(1), 89–95. doi:10.1016/j.socnet.2013.01.003

Mun, M., Hao, S., Mishra, N., Shilton, K., Burke, J., Estrin, D., … Govindan, R. (2010). Personal data vaults: a locus of control for personal data streams. In *Proceedings of the 6th International COnference* (p. 17).

Nguyen, T. T., Nguyen, D. T., Iqbal, S. T., & Ofek, E. (2015). The Known Stranger: Supporting Conversations Between Strangers with Personalized Topic Suggestions. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (pp. 555–564). New York, NY, USA: ACM. doi:10.1145/2702123.2702411

Odom, W., Zimmerman, J., & Forlizzi, J. (2010). Virtual possessions. In *Proceedings of the 8th ACM Conference on Designing Interactive Systems - DIS ’10* (p. 368). New York, New York, USA: ACM Press. doi:10.1145/1858171.1858240

Odom, W., Zimmerman, J., & Forlizzi, J. (2011). Teenagers and Their Virtual Possessions: Design Opportunities and Issues. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1491–1500). New York, NY, USA: ACM. doi:10.1145/1978942.1979161

Odom, W., Zimmerman, J., & Forlizzi, J. (2014). Placelessness, Spacelessness, and Formlessness: Experiential Qualities of Virtual Possessions. In *Proceedings of the 2014 Conference on Designing*

Interactive Systems (pp. 985–994). New York, NY, USA: ACM. doi:10.1145/2598510.2598577

Oku, K., Kotera, R., & Sumiya, K. (2010). Geographical recommender system based on interaction between map operation and category selection. In *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems* (pp. 71–74).

Olson, J., Grudin, J., & Horvitz, E. (2005). A study of preferences for sharing and privacy. *Conference on Human Factors in Computing Systems*. Retrieved from http://portal.acm.org/citation.cfm?id=1057073

Onnela, J.-P., Saramäki, J., Hyvönen, J., Szabó, G., Lazer, D., Kaski, K., … Barabási, A.-L. (2007). Structure and tie strengths in mobile communication networks. *Proceedings of the National Academy of Sciences of the United States of America*, *104*(18), 7332–6. doi:10.1073/pnas.0610245104

Oulasvirta, A., Raento, M., & Tiitta, S. (2005). ContextContacts: re-designing SmartPhone's contact book to support mobile awareness and collaboration. *Proceedings of the 7th International Conference on Human Computer Interaction with Mobile Devices & Services - MobileHCI '05*.

Ozenc, F. K., & Farnham, S. D. (2011). Life " Modes " in Social Media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

Patel, K., Bancroft, N., Drucker, S. M., Fogarty, J., Ko, A. J., & Landay, J. (2010). Gestalt: integrated support for implementation and analysis in machine learning. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology* (pp. 37–46).

Pentland, A. (2009). Reality mining of mobile communications: Toward a new deal on data. *The Global Information Technology Report 2008–2009*, 1981.

Perrault, C. R., Allen, J. F., & Cohen, P. R. (1978). Speech Acts As a Basis for Understanding Dialogue Coherence. In *Proceedings of the 1978 Workshop on Theoretical Issues in Natural Language Processing* (pp. 125–132). Stroudsburg, PA, USA: Association for Computational Linguistics. doi:10.3115/980262.980282

Pousman, Z., Stasko, J. T., & Mateas, M. (2007). Casual information visualization: Depictions of data in everyday life. *Visualization and Computer Graphics, IEEE Transactions on*, *13*(6), 1145–1152.

Ramanathan, N., Alquaddoomi, F., Falaki, H., George, D., Hsieh, C., Jenkins, J., … Estrin, D. (2012). ohmage: An open mobile system for activity and experience sampling. In *2012 6th International Conference on Pervasive Computing Technologies for Healthcare* (pp. 203–204).

Rich, E. (1979a). Building and exploiting user models. In *Proceedings of the 6th international joint conference on Artificial intelligence-Volume 2* (pp. 720–722).

Rich, E. (1979b). User modeling via stereotypes*. *Cognitive Science*, *3*(4), 329–354.

Ricken, S. T., Schuler, R. P., Grandhi, S. A., & Jones, Q. (2010). TellUsWho: Guided Social Network Data Collection. In *2010 43rd Hawaii International Conference on System Sciences* (pp. 1–10). IEEE. doi:10.1109/HICSS.2010.365

Rittel, H. J., & Webber, M. (1973). Dilemmas in a general theory of planning. *Policy Sciences*, *4*(2), 155–169. doi:10.1007/BF01405730

Roberts, S. G. B., & Dunbar, R. I. M. (2011). Communication in social networks: Effects of kinship, network size, and emotional closeness. *Personal Relationships*.

Romanosky, S., Acquisti, A., Hong, J. I., Cranor, L. F., & Friedman, B. (2006). Privacy Patterns for Online Interactions. In *The 11th European Conference on Pattern Languages of Programs (Europlop 2006)*.

Saeb, S., Zhang, M., Karr, C. J., Schueller, S. M., Corden, M. E., Kording, K. P., & Mohr, D. C. (2015). Mobile Phone Sensor Correlates of Depressive Symptom Severity in Daily-Life Behavior: An Exploratory Study. *Journal of Medical Internet Research*, *17*(7), e175. doi:10.2196/jmir.4273

Schilit, B., Adams, N., & Want, R. (1994). Context-aware computing applications. *Mobile Computing Systems and Applications*.

Schön, D. A. (1983). *The reflective practitioner: How professionals think in action* (Vol. 5126). Basic books.

Sellen, A. J., & Whittaker, S. (2010). Beyond Total Capture: A Constructive Critique of Lifelogging. *Commun. ACM*, *53*(5), 70–77. doi:10.1145/1735223.1735243

Simon, H. A. (1969). The sciences of the artificial. *Cambridge, MA*.

Sleeper, M., Balebako, R., Das, S., McConahy, A. L., Wiese, J., & Cranor, L. F. (2013). The post that wasn't: exploring self-censorship on facebook. In *Proceedings of the 2013 conference on Computer supported cooperative work* (pp. 793–802). ACM.

Spencer, L., & Pahl, R. E. (2006). *Rethinking friendship: hidden solidarities today*. Princeton University Press.

Starner, T. E., Snoeck, C. M., Wong, B. A., & McGuire, R. M. (2004). Use of mobile appointment scheduling devices. In *CHI'04 Extended Abstracts on Human Factors in Computing Systems* (pp. 1501–1504).

Tang, J., Yankelovich, N., Begole, J., Kleek, M., Li, F., & Bhalodia, J. (2001). ConNexus to awarenex: extending awareness to mobile users. *ACM Conference on Human Factors in Computing Systems (CHI2001), CHI Letters 3(1)*.

Tang, K. P., Lin, J., Hong, J. I., Siewiorek, D. P., & Sadeh, N. (2010). Rethinking Location Sharing: Exploring the Implications of Social-Driven vs. Purpose-Driven Location Sharing.

Tolmie, P., Pycock, J., Diggins, T., Maclean, A., & Karsenty, A. (2002). Unremarkable computing. *Proceedings*. doi:http://dx.doi.org/10.1145/503376.503448

Wang, D., Pedreschi, D., Song, C., Giannotti, F., & Barabasi, A.-L. (2011). Human mobility, social ties, and link prediction. In *In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (p. 1100). doi:10.1145/2020408.2020581

Wang, R., Chen, F., Chen, Z., Li, T., Harari, G., Tignor, S., … Campbell, A. T. (2014).

StudentLife: assessing mental health, academic performance and behavioral trends of college students using smartphones. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (pp. 3–14).

Want, R., Hopper, A., Falcão, V., & Gibbons, J. (1992). The active badge location system. *ACM Transactions on Information Systems*. doi:10.1145/128756.128759

Want, R., Pering, T., Danneels, G., Kumar, M., Sundar, M., & Light, J. (2002). The personal server: Changing the way we think about ubiquitous computing. In *Ubicomp 2002: Ubiquitous Computing* (pp. 194–209). Springer.

Weiser, M. (1991). The computer for the 21st century. *Scientific American*, *265*(3), 94–104. doi:http://dx.doi.org/10.1145/329124.329126

Weka 3: Data Mining Software in Java. (n.d.).

Westin, A. (2001). Opinion surveys: What consumers have to say about information privacy. *Prepared Witness Testimony, The House Committee on Energy and Commerce*.

Whittaker, S., Jones, Q., & Terveen, L. (2002). Contact management. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work - CSCW '02* (p. 216). New York, New York, USA: ACM Press. doi:10.1145/587078.587109

Wiese, J., Biehl, J. T., Turner, T., van Melle, W., & Girgensohn, A. (2011). Beyond "Yesterday"s Tomorrow': Towards the Design of Awareness Technologies for the Contemporary Worker. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services* (pp. 455–464). New York, NY, USA: ACM. doi:10.1145/2037373.2037441

Wiese, J., Hong, J. I., & Zimmerman, J. (2014). Challenges and opportunities in data mining contact lists for inferring relationships. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*.

Wiese, J., Kelley, P. G., Cranor, L. F., Dabbish, L., Hong, J. I., & Zimmerman, J. (2011). Are you close with me? are you nearby?: investigating social groups, closeness, and willingness to share. In *In Proceedings of the 13th international conference on Ubiquitous computing - UbiComp '11*. New York, New York, USA. doi:10.1145/2030112.2030140

Wiese, J., Min, J.-K., Hong, J. I., & Zimmerman, J. (2015). "You Never Call, You Never Write": Call and SMS Logs Do Not Always Indicate Tie Strength. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing* (pp. 765–774). New York, NY, USA: ACM. doi:10.1145/2675133.2675143

Wiese, J., Saponas, T. S., & Brush, A. J. B. (2013). Phoneprioception: Enabling Mobile Phones to Infer Where They Are Kept. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 2157–2166). New York, NY, USA: ACM. doi:10.1145/2470654.2481296

Yang, J., Yessenov, K., & Solar-Lezama, A. (2012). A language for automatically enforcing privacy policies. In *ACM SIGPLAN Notices* (Vol. 47, pp. 85–96).

Zhou, W. X., Sornette, D., Hill, R. A., & Dunbar, R. I. M. (2005). Discrete hierarchical organization of social group sizes. In *In Proceedings of the Royal Society of London B: Biological Sciences* (Vol. 272).